Networked Control System: Stability, Robustness and Controllability with respect to Packet Dropouts and Scheduling

Lješnjanin Merid

Submitted in partial fulfillment of the requirements of the degree of

Doctor of Philosophy
(with coursework component)

Department of Electrical and Electronic Engineering
THE UNIVERSITY OF MELBOURNE

June 2015

Copyright © 2015 Lješnjanin Merid

All rights reserved. No part of the publication may be reproduced in any form by print, photoprint, microfilm or any other means without written permission from the author.

Abstract

Control systems which use a *communication network* as a *communication medium* are the focus of this thesis. These systems are widely recognized as *Networked Control Systems (NCSs)*. In particular, we consider NCSs in which the corresponding network induces two communication issues. One of them is a packet dropout(s) while the other is scheduling. To mitigate the undesirable effects of packet dropouts and scheduling, such as instability or deteriorated performance, we use a protocol and controller co-design. More precisely, we use a Model Predictive Control (MPC) framework and the flexible nature of NCS architecture which allows for distributed computation.

Considering a specific NCS architecture affected with packet dropouts and/or scheduling we focus on stability, robustness and controllability properties of the corresponding NCS.

In particular, we begin by considering stability property of a NCS in which the corresponding network is located between the controller output and the plant input. The network is prone to packet dropouts and it induces scheduling of its communication resource. To address these communication constraints we employ a protocol and controller co-design and show stability, in particular, Uniform Global Asymptotic Stability (UGAS) of the corresponding NCS state. We use two approaches to establish this result. One approach consists of finding an appropriate Lyapunov function while the other approach uses a cascade idea.

Following this is an investigation of the same NCS architecture with the difference that it is governed not with a standard MPC controller but an Economic MPC controller. Here we combine several recently established results for an Economic MPC in a way so that our result can be applied off the shelf to establish UGAS of the corresponding NCS

state.

We then proceed by considering robustness properties of the same NCS architecture. In particular, we consider the case where, additionally, the plant is affected with exogenous disturbances. Here we exploit a concept of nonlinear gains to establish the corresponding result, namely, partial nonlinear gain ℓ_2 stability. We also establish partial linear gain ℓ_2 stability, recover and strengthen a result from the literature for the case when there is no disturbance, provide an alternative robustness characterization for the case when there is no scheduling and finally, by using stronger assumptions, we establish Input-to-State Stability (ISS) of the corresponding NCS state.

The last theoretical contribution is controllability of a NCS where the network only induces the scheduling of its communication resource. Namely, we first provide an interesting and novel model which is followed by splitting attention to a NCS with a nonlinear and a NCS with a linear plant. In the former case, we establish general results while in the latter case we extend a result from the corresponding literature and use it to establish our controllability result.

Finally, we finish with the implementation of the obtained results within a Hardware-in-the-loop (HIL) simulation. We verify the expectations from theoretical stability and robustness results. Finally, we encounter several issues while carrying out the implementation which will be used as a motivation for further research.

Declaration

This is to certify that

1. the thesis comprises only my original work towards the PhD,
2. due acknowledgement has been made in the text to all other material used,
3. the thesis is less than 100,000 words in length, exclusive of tables, maps, bibliographies and appendices.

Lješnjanin Merid, June 2015

This page intentionally left blank.

Acknowledgements

I would like to use this opportunity to thank my supervisor Dragan Nešić for his time, help, support and above all, his patience. I am truly grateful for each advice and suggestion but more importantly for the established friendship.

A special thanks is due to Daniel E. Quevedo with whom I worked on most of my Ph.D. projects and whose suggestions lead to many improvements.

I would also like to thank my committee members, Ying Tan and Wei Wang, for all useful advice.

A special thanks also goes to Michael Cantoni, Peter Farrell and Ying Tan for providing me with the opportunity to work with the undergraduate students in their respective subjects.

Also, I would like to thank all the students and staff at the University of Melbourne for making such a stimulating and friendly environment.

Finally, expressing in words my gratitude for love, understanding and support from my family and friends is simply impossible.

This page intentionally left blank.

To you!

This page intentionally left blank.

Contents

1 Mathematical Preliminaries and Notation **1**

2 Introduction **7**
2.1 Goal . . . 12
2.2 Motivation . . . 12
2.3 Methodology . . . 15
2.4 Literature review . . . 19
2.4.1 Networked Control Systems . . . 19
2.4.2 Model Predictive Control . . . 24
2.4.3 Packetized MPC . . . 27
2.5 Overview and contributions . . . 31
2.6 Publications . . . 34

3 Stability with respect to Packet Dropouts and Scheduling **37**
3.1 NCS Architecture . . . 40
3.1.1 Plant . . . 40
3.1.2 Network . . . 42
3.1.3 Buffer . . . 43
3.1.4 Controller . . . 49
3.2 Stability analysis . . . 53
3.2.1 Assumptions . . . 53
3.2.2 Results . . . 56
3.3 Proofs . . . 63

4 Stability with respect to Packet Dropouts and Scheduling - Economic MPC **75**
4.1 UGAS-Economic MPC . . . 77
4.2 Proofs . . . 83

5 Robustness with respect to exogenous disturbances **87**
5.1 NCS architecture . . . 89
5.1.1 Plant . . . 89
5.1.2 Network . . . 90
5.1.3 Buffer . . . 91
5.1.4 Controller . . . 91
5.2 Robustness Analysis . . . 94

5.2.1 Assumptions . . . 96
5.2.2 Results . . . 99
5.2.3 Simulations . . . 111
5.3 Proofs . . . 115

6 Controllability with respect to Scheduling 129
6.1 NCS architecture . . . 130
6.1.1 Plant . . . 131
6.1.2 Network protocols . . . 132
6.1.3 Processing devices . . . 134
6.1.4 NCS architecture with dynamic devices . . . 136
6.1.5 NCS architecture with static devices . . . 136
6.2 Results . . . 137
6.2.1 Controllability: nonlinear plants . . . 139
6.2.2 Controllability: linear plants . . . 143
6.3 Proofs . . . 147

7 Implementation 151
7.1 NCS architecture . . . 153
7.1.1 Plant . . . 154
7.1.2 Network . . . 154
7.1.3 Buffer . . . 157
7.1.4 Controller . . . 158
7.2 Assumptions . . . 160
7.2.1 Stability – UGAS . . . 160
7.2.2 Robustness – Partial linear gain ℓ_2 stability . . . 161
7.3 Implementation . . . 162
7.4 Results . . . 164

8 Conclusion and future work 171
8.1 NCS architecture . . . 172
8.2 Network . . . 172
8.3 Analysis and design . . . 173
8.4 Implementation . . . 173

A Implementation details 175
A.1 Hardware-in-the-loop simulation . . . 175
A.2 Control Area Network . . . 176
A.3 Simulation setup . . . 183
A.3.1 Considered plant . . . 184
A.3.2 Designing CAN structure . . . 184
A.3.3 Physical realization of CAN structure . . . 189
A.3.4 Designing MPC controller(s) . . . 189
A.3.5 NCS model . . . 192
A.3.6 ControlDesk Next Generation® . . . 193
A.4 Specific settings of RTICANMM blocks . . . 195

List of Figures

2.1 Control systems. 7
2.2 A "traditional" interconnection versus usage of a network; Σ_i, $i \in \{1, \ldots, 8\}$ denotes a system while u and y denotes its corresponding input and output, respectively. 8
2.3 A wireless NCS; e.g., with Bluetooth network. 10
2.4 General NCS with extra devices. 15
2.5 Schematic methodology for addressing packet dropouts and scheduling. . 19
2.6 An NCS architecture for addressing unbounded delay. 28
2.7 An NCS architecture for addressing the issues of packet dropouts and scheduling. 29
2.8 An NCS architecture for addressing the issues of packet dropouts. 30
2.9 An NCS architecture for addressing robustness with respect to the packet dropouts and scheduling. 33
2.10 An A NCS architecture for addressing controllability. 33

3.1 An NCS architecture considered for stability. 37
3.2 Illustration of the function of a buffer. 38
3.3 Illustration of what we refer to as inter-sample behavior; k denotes current discrete time. 57

5.1 A NCS architecture for investigating robustness with respect to packet dropouts and scheduling. 89
5.2 Dynamic and static scheduling comparison; buffer set to zero value; dropout probability 0.2, 1 – channel 1 (e.g., $u_{\mathfrak{p}_1}$) while 2 – channel 2 (e.g., $u_{\mathfrak{p}_2}$). . . . 113
5.3 Dynamic and static scheduling comparison; buffer set to the "last value"; dropout probability 0.2, 1 – channel 1 (e.g., $u_{\mathfrak{p}_1}$) while 2 – channel 2 (e.g., $u_{\mathfrak{p}_2}$).113
5.4 Dynamic and static scheduling comparison; buffer set to zero value; dropout probability 0.6, 1 – channel 1 (e.g., $u_{\mathfrak{p}_1}$) while 2 – channel 2 (e.g., $u_{\mathfrak{p}_2}$). . . . 114
5.5 Dynamic and static scheduling comparison; buffer set to the "last value"; dropout probability 0.6, 1 – channel 1 (e.g., $u_{\mathfrak{p}_1}$) while 2 – channel 2 (e.g., $u_{\mathfrak{p}_2}$).114

6.1 A NCS architecture for addressing controllability. 131
6.2 A NCS architecture for addressing controllability - a simplified representation. 132
6.3 Conceptual abstraction of static devices for a plant with three inputs. . . . 134
6.4 A convention for "transitioning" form hybrid time to discrete time. 137

6.5 A block diagram abstraction of the concept of the realization of control sequences over a network. 138
6.6 Illustration of tricking TOD with APC_0 for a plant with three inputs. 141
6.7 Case 1: $u^\delta(0) = x_{\mu^d_{\mathfrak{p}}}(0)$; Case 2: $u^\delta(0) \neq x_{\mu^d_{\mathfrak{p}}}(0)$ but when $u_{\mathfrak{p}}(k) = u^\delta(k)$ we have $\phi_{f_{\mathfrak{p}}}(k, x_{\mathfrak{p}}, \{u^\delta\}_0^\infty) = \phi_{f_{\mathfrak{p}}}(k, x_{\mathfrak{p}}, \{\tilde{u}^\delta\}_0^\infty)$ where $k \geq m$; Case 3: $u^\delta(0) \neq x_{\mu^d_{\mathfrak{p}}}(0)$ but when $u_{\mathfrak{p}}(k) = u^\delta(k)$ we have $\phi_{f_{\mathfrak{p}}}(k, x_{\mathfrak{p}}, \{u^\delta\}_0^\infty) \neq \phi_{f_{\mathfrak{p}}}(k, x_{\mathfrak{p}}, \{\tilde{u}^\delta\}_0^\infty)$ 142

7.1 Implementation as a Hardware-In-the-Loop (HIL) simulation. 151
7.2 Implemented NCS architecture. 153
7.3 Demonstrating quantization issues on CAN bus; here we use only one byte to transmit the values of sine signal since we have to use one byte to transmit the values of each control element from the sequence of the optimally predicted controls. 155
7.4 Demonstrating delay on CAN bus. 156
7.5 The effects of generated packet dropouts. 157
7.6 Conceptual illustration of the considered NCS in Simulink®. 164
7.7 Stability: scheduling of input $u_{\mathfrak{p}_1}$ 165
7.8 Stability: scheduling of input $u_{\mathfrak{p}_2}$ 166
7.9 Stability: Euclidean norm of plant state 166
7.10 Stability: indication of packet dropouts through updating contents of buffer located before input $u_{\mathfrak{p}_2}$ 167
7.11 Stability: Euclidean norm of plant state due to packet dropouts 167
7.12 Robustness: Euclidean norm of plant disturbance 168
7.13 Robustness: indication of packet dropouts through updating contents of buffer located before input $u_{\mathfrak{p}_2}$ 168
7.14 Robustness: Euclidean norm of plant state 169

A.1 Serial bus networking; node $n.$ is an abstraction for a communication participant, e.g., a plant. 177
A.2 OSI 7 layer communication model. 178
A.3 Basic communication principles in OSI 7 layer communication model; PCI - Protocol Control Information, PDU - Protocol Data Unit. 179
A.4 Three Layer model. 179
A.5 CAN data framing. 181
A.6 Principle of CAN bus access; ITM - Intermission. 182
A.7 CAN structure designed in Vector Informatik® CANdb++ editor 185
A.8 dSPACE® Simulator - back view. 189
A.9 Physical connections resembling the designed CAN connection structure. 190
A.10 Simulink® model of the implementation of the corresponding NCS. 192
A.11 Other custom made Simulink® sub-models. 193
A.12 dSPACE® RTICANMM Simulink® blockset. 193
A.13 dSPACE® ControlDesk New Generation®. 194
A.14 Generating communication errors and/or misbehaviors which are effectively packet dropouts. 195
A.15 Specific settings of RTICANMM blocks 196

A.16 Specific settings of RTICANMM blocks 196
A.17 Specific settings of RTICANMM blocks 197
A.18 Specific settings of RTICANMM blocks 197
A.19 Specific settings of RTICANMM blocks 197
A.20 Specific settings of RTICANMM blocks 198
A.21 Specific settings of RTICANMM blocks 198
A.22 Specific settings of RTICANMM blocks 198
A.23 Specific settings of RTICANMM blocks 199
A.24 Specific settings of RTICANMM blocks 199
A.25 Specific settings of RTICANMM blocks 199
A.26 Specific settings of RTICANMM blocks 200
A.27 Specific settings of RTICANMM blocks 200

This page intentionally left blank.

List of Tables

1.1 Notation for basic sets. 1
1.2 Symbols for parts of NCS architecture. 4
1.3 Symbols for dynamical description where $\diamond \in \{\mathfrak{c}, \mathfrak{n}, \mathfrak{b}, \mathfrak{p}\}$. 4

2.1 Typical references for network-induced issues. 25

6.1 Communication sequence matrix polynomials for different periodic sequences and roots of the corresponding polynomials for a second ordered system; permutation of added elements, adding $(1, 0)$ instead of $(0, 1)$, or adding both where possible, does not generate new polynomials. 149

This page intentionally left blank.

Chapter 1

Mathematical Preliminaries and Notation

We will present a collection of mathematical preliminaries and notations used throughout the thesis. We start with notation used for basic sets given in Table 1.1.

Set	Symbol	Definition
Natural numbers	$\mathbb{N}$	Consult [1]
Integer numbers	$\mathbb{Z}$	Consult [1]
Real numbers	$\mathbb{R}$	Consult [1]
Complex numbers	$\mathbb{C}$	Consult [1]
"Dummy"	$\mathbb{D}$	$\mathbb{D} \in \{\mathbb{N}, \mathbb{Z}, \mathbb{R}, \mathbb{C}\}$
Extended "dummy"	$\bar{\mathbb{D}}$	$\mathbb{D} \cup \{-\infty, \infty\}$
Elements bounded by a specific element c	$\mathbb{D}_{\diamond c}$	$\{\nu \in \mathbb{D} : \nu \diamond c, c \in \mathbb{D}, \diamond \in \{\leq, <, >, \geq\}\}$
Natural numbers including zero	$\mathbb{N}_0$	$\mathbb{Z}_{\geq 0}$

Table 1.1: Notation for basic sets.

We will use symbol $^\top$ for transposition and quite often we will use tuple notation to denote a column vector, that is

$$(\nu_1, \ldots, \nu_i) := \left[\nu_1^\top \ldots \nu_i^\top\right]^\top = \begin{bmatrix} \nu_1 \\ \vdots \\ \nu_i \end{bmatrix}, \; i \in \mathbb{N}, \tag{1.1}$$

where v_j are vectors for each $j \in \{1, \ldots, i\}$.

The origin element of the ith dimensional space of real numbers is denoted by 0^i, more precisely

$$0^i := (0, \ldots, 0) \in \mathbb{R}^i,\ i \in \mathbb{N}. \tag{1.2}$$

The $i \times i$ identity matrix is denoted via

$$I_{i\times i} = I_i := \operatorname{diag}(1, \ldots, 1) := \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{bmatrix},\ i \in \mathbb{N}, \tag{1.3}$$

while $i \times i$ zero matrix is denoted via

$$0_{i\times i} = 0_i := \operatorname{diag}(0, \ldots, 0) := \begin{bmatrix} 0 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & 0 \end{bmatrix},\ i \in \mathbb{N}. \tag{1.4}$$

We use the following notation for a sequence of elements

$$\left.\begin{aligned} v_j^k := \{v(i)\}_{i=j}^k,\ \text{ if } j \leq k, \\ \{\},\ \text{ if } j > k, \end{aligned}\right\} \tag{1.5}$$

where $j \in \bar{\mathbb{Z}}$ and $k \in \bar{\mathbb{Z}}$.

We define a set of infinite sequences initialized at index 0 whose elements take value from some set $\mathcal{V}$ as

$$\mathcal{S}^{\mathcal{V}} := \{v_0^\infty : v(i) \in \mathcal{V},\ \forall i \in \mathbb{N}_0\}. \tag{1.6}$$

We use the following definitions of some functions.

Definition 1.1 (Positive definite function)**.** *A function $\rho : \mathbb{R}^n \to \mathbb{R}_{\geq 0}$ is said to be positive definite ($\rho \in \mathcal{PD}$) with respect to $v = c$ if it is continuous, $\rho(c) = 0$ and $\rho(v) > 0$ for all $v \neq c$.*

Definition 1.2 (Class - $\mathcal{K}$ function). *A function $\alpha : \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$ is said to be of class-$\mathcal{K}$ ($\alpha \in \mathcal{K}$) if it is continuous, $\alpha(0) = 0$, and strictly increasing.*

Definition 1.3 (Class - $\mathcal{K}_\infty$ function). *A function $\alpha : \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$ is said to be of class-$\mathcal{K}_\infty$ ($\alpha \in \mathcal{K}_\infty$) if $\alpha \in \mathcal{K}$ and, in addition, $\lim_{i\to\infty} \alpha(i) = \infty$.*

Definition 1.4 (Class - $\mathcal{L}$ function). *A function $\sigma : \mathbb{R}_{\geq 0} \to \mathbb{R}_{>0}$ is said to be of class-$\mathcal{L}$ ($\sigma \in \mathcal{L}$), if it is continuous, monotonically decreasing to zero for nonzero argument and it is zero at zero.*

Definition 1.5 (Class - $\mathcal{KL}$ function). *A function $\beta : \mathbb{R}_{\geq 0} \times \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$ is said to be of class-$\mathcal{KL}$ ($\beta \in \mathcal{KL}$) if it is class-$\mathcal{K}$ in its first argument and class-$\mathcal{L}$ in its second argument; that is, $\beta \in \mathcal{KL}$ if for each fixed $j \in \mathbb{R}_{\geq 0}$, $\beta(\cdot, j) \in \mathcal{K}$ and for each fixed $i \in \mathbb{R}_{\geq 0}$, $\beta(i, \cdot) \in \mathcal{L}$.*

Definition 1.6 (UIB function [2]). *A class-$\mathcal{KL}$ function $\beta(i, j)$ is called Uniformly Incrementally Bounded (UIB) if there exists a number $P > 0$ such that $\beta(i, j) \leq P\beta(i, j+1)$ for each $i \geq 0$ and each $j \in \mathbb{N}$.*

We will use the fact that for any class-$\mathcal{K}_\infty$(class-$\mathcal{K}$) function α and any non-negative real numbers c_1 and c_2 it holds that

$$\frac{1}{2}\alpha(c_1) + \frac{1}{2}\alpha(c_2) \leq \alpha(c_1 + c_2) \leq \alpha(2c_1) + \alpha(2c_2). \tag{1.7}$$

Using the former inequality it is easy to show the following ones

$$\left.\begin{array}{c} -\alpha(c_1 + c_2) \leq -\frac{\alpha(c_1)}{2} + \frac{\alpha(c_2)}{2}, \\ \alpha\left(\sum_{l=i}^{j} c_l\right) \leq \sum_{l=i}^{j} \alpha(2^{j-i} c_l),\ i \leq j, \\ \sum_{j=j_1}^{j_2} \alpha\left(\sum_{l=i}^{j} c_l\right) \leq (j_2 - j_1 + 1) \sum_{l=i}^{j_2} \alpha(2^{j_2-i} c_l),\ i \leq j \leq j_1 \leq j_2, \end{array}\right\} \tag{1.8}$$

where all elements from $\{i, j, j_1, j_2\}$ are non-negative numbers. We have dedicated some lowercase Fraktur letters[1] to denote certain parts of NCS architecture, see Table 1.2.

Further, for dynamical description we use symbols presented in Table. 1.3.

[1]The reason for this is that same symbol in standard letters will be used for dimension; for instance, a dimension for plant output vector $y_{\mathfrak{p}}$ will be denoted with p, e.g., $y_{\mathfrak{p}} \in \mathbb{R}^p$.

NCS element	Lowercase Fraktur letter
Controller	$\mathfrak{c}$
Network	$\mathfrak{n}$
Buffer	$\mathfrak{b}$
Plant	$\mathfrak{p}$
Address	$\mathfrak{a}$
Data	$\mathfrak{d}$

Table 1.2: Symbols for parts of NCS architecture.

	Symbol
Dynamical system	$\Sigma_\diamond$
State	$x_\diamond$
State mapping	$f_\diamond$
Input	$u_\diamond$
Input mapping	$v_\diamond$
Disturbance	$w_\diamond$
Disturbance mapping	$\varpi_\diamond$
Output	$y_\diamond$
Output mapping	$h_\diamond$

Table 1.3: Symbols for dynamical description where $\diamond \in \{\mathfrak{c}, \mathfrak{n}, \mathfrak{b}, \mathfrak{p}\}$.

Solution mapping

Let $\diamond \in \{\mathfrak{p}, \mathfrak{c}, \mathfrak{n}, \mathfrak{b}\}$ and consider the corresponding dynamical system in discrete-time

$$\Sigma_\diamond : x_\diamond(k+1) = f_\diamond(x_\diamond(k), u_\diamond(k), w_\diamond(k)),\ k \in \mathbb{N}_0, \tag{1.9}$$

where $x_\diamond \in \mathbb{R}^{n_\diamond}$ is the state, $u_\diamond \in \mathbb{R}^{m_\diamond}$ is the input and $w_\diamond \in \mathbb{R}^{q_\diamond}$ is the exogenous disturbance of $\diamond$ with $n_\diamond, m_\diamond$ and $q_\diamond$ belonging to natural numbers. We denote the solution of $\Sigma_\diamond$, $j-i$ steps into the future, starting at initial condition $x_\diamond$ at time instant $i \leq j$, under the influence of control input sequence $\{u_\diamond(k)\}_{k=i}^{j-1}$ and disturbance sequence $\{w_\diamond(k)\}_{k=i}^{j-1}$ via $\phi_{f_\diamond}\big(j-i, x_\diamond, \{u_\diamond(k)\}_{k=i}^{j-1}, \{w_\diamond(k)\}_{k=i}^{j-1}\big)$. Note that $\phi_{f_\diamond}\big(0, x_\diamond, \{\}, \{\}\big) = x_\diamond = x_\diamond(0)$. Also, note that we do not use a frequently used notation for an initial condition $x_\diamond$, such as writing a number "0" or a lower case letter "o" at the subscript place, due to the fact we have

decided to use the subscript place to denote objects ($\diamond$) and/or index of a corresponding vector element.

Whenever appropriate we will use the following succinct notation for Eq. 1.9 (and other similar equations)

$$\Sigma_\diamond : x_\diamond^+ = f_\diamond(x_\diamond, u_\diamond, w_\diamond). \tag{1.10}$$

Finally, if a system is defined as augmentation of several variables, e.g.,

$$\Sigma_x : x^+ := \begin{bmatrix} x_\mathfrak{p}^+ \\ x_\mathfrak{b}^+ \end{bmatrix} = \begin{bmatrix} f_\mathfrak{p}(x_\mathfrak{p}, u_\mathfrak{p}, w_\mathfrak{p}) \\ f_\mathfrak{b}(x_\mathfrak{b}, u_\mathfrak{p}, w_\mathfrak{p}) \end{bmatrix} =: f(x, u_\mathfrak{p}, w_\mathfrak{p}) \tag{1.11}$$

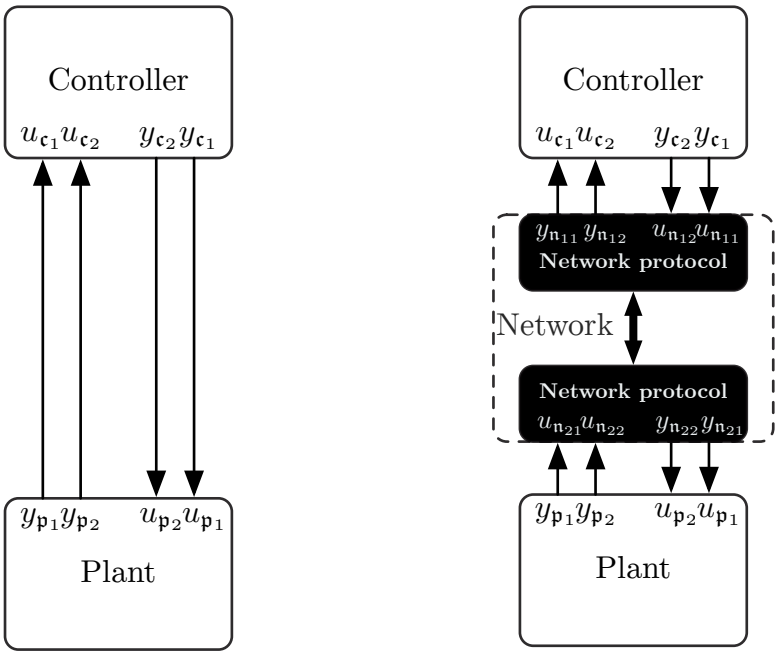
then, to extract, say plant component from a solution mapping of system Σ_x, that is $\phi_f(\cdot)$, we will write $\phi_{f_\mathfrak{p}}(\cdot)$.

This page intentionally left blank.

Chapter 2

Introduction

The focus of this thesis are the so-called *Networked Control Systems (NCSs)*. More precisely, these are a special class of *control systems* in which a *communication network* is used as a *communication medium* to send and/or receive the corresponding control-related signals.



(a) "Traditional".

(b) Networked (wired).

Figure 2.1: Control systems.

In order to provide more detail about NCSs, let us consider a "traditional" control sys-

tem[1], see Fig. 2.1a. As illustrated, the chosen "traditional" control system is very simple. Both, the plant and the controller, have two inputs and two outputs, but more importantly, for each corresponding pair of inputs and outputs there is a *dedicated point-to-point connection*. Equally important is to notice that this dedicated point-to-point connection ensures that the output of the plant is *equal* to the input of the controller and vice versa, i.e., $u_{\mathfrak{p}.} = y_{\mathfrak{c}.}$ and $u_{\mathfrak{c}.} = y_{\mathfrak{p}.}$.

On the other hand, an alternative to interconnecting the corresponding inputs and outputs is to use a *communication network* as depicted in Fig. 2.1b, which schematically illustrates an NCS diagram. In this case, we do not have dedicated point-to-point connections and thus the output of the plant is *not* (necessarily) *equal* to the input of the controller and vice versa; i.e., it is very likely that $u_{\mathfrak{p}.} = y_{\mathfrak{n}_2.} \neq u_{\mathfrak{n}_1.} = y_{\mathfrak{c}.}$ and $u_{\mathfrak{c}.} = y_{\mathfrak{n}_1.} \neq u_{\mathfrak{n}_2.} = y_{\mathfrak{p}.}$.

A figure that illustrates the difference between "traditional" control systems and NCSs even more is depicted in Fig. 2.2. Moreover, this figure sets a nice stage for making the basic comparison between the two systems, which in turn will additionally provide more detail about NCSs, as desired.

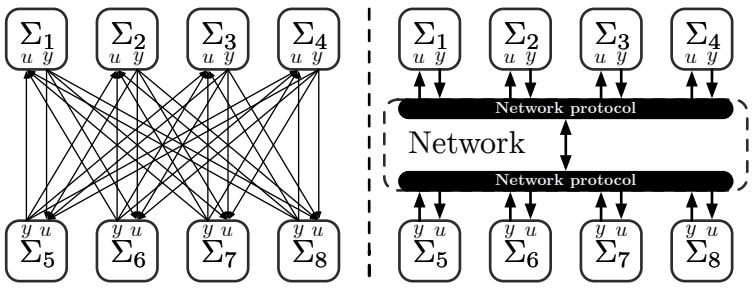


Figure 2.2: A "traditional" interconnection versus usage of a network; Σ_i, $i \in \{1, \ldots, 8\}$ denotes a system while u and y denotes its corresponding input and output, respectively.

Let us first focus on a "traditional" type of interconnection, i.e., a non-networked type of interconnection. As illustrated in the previous figures, for each pair of inputs and outputs one needs a dedicated channel, i.e., a dedicated point-to-point connection. In small and simple control systems this approach of interconnecting the inputs and out-

[1]We note that the concept of a ("traditional") control system is directly related to the concept of a system and the concept of control. Since both concepts are very broad, the usage of generic and general definitions might potentially distract the reader. For this reason, and for the sake of rigor, we rely on the corresponding notions and definitions documented in control literature, e.g., see [3–8].

puts makes a lot of sense. However, in larger systems or systems with many inputs and outputs this type of interconnection increases the number of dedicated channels which in turn increases the price, volume and weight of the corresponding system. Moreover, installation of the system, its maintenance, troubleshooting and/or addition of new elements to the existing system are all relatively complex and expensive tasks; especially in large-scale "traditional" control systems.

On the other hand, its networked counterpart, i.e., NCS (see Fig 2.2), does not have these issues. Moreover, the above-mentioned issues for "traditional" control systems can be seen as some of the advantages of NCSs. More precisely, as illustrated in the previous figures, the number of dedicated channels decreases or, as illustrated in Fig. 2.3, it despairs in a case of wireless network. This results in a lower price, volume and weight, making NCS paradigm ideal for transporting systems such as cars, plains and spacecrafts and certain applications such as vehicle platooning (which in particular is not realizable with wired network). Moreover, installation, maintenance, troubleshooting and/or addition of new elements all become much simpler and less expensive tasks; this, together with a reduction in price, resulted in a wide adaptation of networks in large-scale systems such as power and/or manufacturing plants.

On the other hand, when it comes to design and analysis, the "traditional" control systems are comparatively older and thus have much larger set of tools for analysis and design. Correspondingly, there are not as many tools which one can use for design and analysis of NCSs. Now, one would hope that the established tools for analysis and design of "traditional" control system should be applicable (perhaps with minor modifications) to NCSs. Unfortunately, more often than not, due to intrinsic network communication phenomena this is not the case and a lot of work needs to be done to make the corresponding tools applicable in the NCS setting. There are quite a few of intrinsic network communication phenomena including:

- **Packet dropouts**

  Most networks operate in a packet-based fashion, i.e., they communicate the corresponding data via data packets. Usually, a packet dropout occurs due to the packet collisions, traffic congestion and/or failed transmissions. Moreover, if the delay of

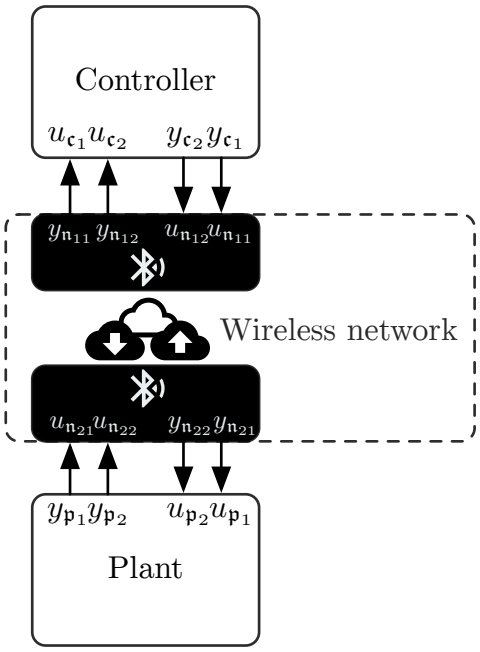


Figure 2.3: A wireless NCS; e.g., with Bluetooth network.

the package reception is too long (or even infinite), the packet is considered to be dropped; that is, the information is lost. Also, note that network induced delays more than one sampling/transmission interval can lead to a packet disorder, which corresponds to mixing packet indices of the packets transmitted over a network, see [9]. However, due to *past packets rejection logic*, the older packets will be discarded if the most recent packet has arrived. Thus, due to implementation of this logic, the packet disorder can be viewed as the packet dropout in the analysis and design. However, it should be noted that the discard is due to the implemented logic, which essentially differs from the real packet losses.

- **Scheduling**

  Another intrinsic network communication phenomenon is scheduling of its communication resource, e.g., the network bandwidth. Scheduling usually occurs due to the available resources (the finite bandwidth) not matching the demand for transmission. One of the main reasons for this mismatch between the demand and the

supply, is the sharing of network resources among different systems; note that the ability to do so is one of the main advantages for exploitation of the network as a communication medium in control applications, e.g., see Fig. 2.2.

- **Delays**
  The sources of delay can roughly be classified into four groups: processing delay - time routers take to process the packet header, queuing delay - time the packet spends in routing queues, transmission delay - time it takes to push the packet's bits onto the link, and propagation delay - time for a signal to reach its destination.

- **Quantization**
  Similarly as for scheduling, the quantization occurs due to finite bandwidth of the network making data transmission with infinite precision unrealistic. Namely, due to finite bandwidth, packets have finite length, and hence we need to quantize.

- **Time-varying packet transmission and/or sampling intervals**
  Due to limited processing resources and insufficient accuracy of the local clocks, time instants at which the network transmits and samples data packets is inaccurate. This results in time-varying packet transmission and/or sampling intervals.

Indeed, there are much more intrinsic network communication phenomena such as clock asynchronization among local and remote nodes and network security and safety. Each of the above-mentioned network induced communication issues, alone or in combination with another communication issue(s) can potentially deteriorate the performance of the corresponding NCS or even destabilize it, e.g., see [10–14] and references therein. Addressing the effects of these issues constitutes most of the research within the NCS community. In this thesis we focus on two network induced communication issues, namely, *packet dropouts* and *scheduling*. The motivation for our focus is provided in the sequel. We start by first stating precisely and succinctly the goal of the thesis.

2.1 Goal

The main goal of the thesis, is to provide a better and deeper understanding of NCSs affected with packet dropouts and scheduling. Our approach to achieving this goal is by concentrating on the following control system properties:

- Stability,
- Robustness,
- Controllability.

2.2 Motivation

We motivate NCSs from three perspectives. First, we draw the attention to the economical and practical improvements over "traditional" control systems. Then, we focus on the conceptual value, namely, the fact that NCS constitute an abstraction and architecture for the convergence of control, computation and communication which is predicted to be the next big leap in (information) technology. Finally, we end the motivation section by concentrating on theoretical values which come from addressing fundamental communication constraints in control systems.

Economical and practical improvements

Due to the rapid technological advancements in communication technology in the last few decades, communication networks have become ever more omnipresent; they are everywhere, in our pockets, in our cars, in our homes, on our streets, etc. Indeed, a resource which is only omnipresent does not necessarily imply we should adapt to it, especially when it comes to control applications which are so sensitive to real-time communication of control-related signals. Extra incentives are necessary and present communication networks have a sufficient number of incentives, spanning both, economical and practical fields, to cause a fundamental change in design of control systems. As we have

mentioned earlier, see also Fig. 2.2, the amount of wiring drastically reduces, or even disappears in the case of a wireless network (see Fig. 2.3), resulting in the lower price of the overall system and reduced volume and weight. While the reduction of the price is especially important for large-scale systems such as power and/or manufacturing plants, reduction in volume and weight (and price) is extremely important for transportation systems such as cars, airplanes and spacecrafts. Furthermore, the complexity of the overall system reduces not only through wiring, see Fig. 2.2 once again, but also through the installation of the corresponding system, its maintenance, troubleshooting and the ease of the addition of new elements to the existing system. Last but not the least, the current networks are sufficiently fast and reliable to be used in real-time applications, i.e., control applications, e.g., application of Control Area Network (CAN) and FlexRay network in cars, aircrafts and spacecrafts.

An abstraction and architecture for the next technological leap

Further, with the increasing trend of technological enhancements, the level of the interaction between the technology and our physical environment is set to be even greater in the near future. At the moment, some of the examples that capture this interaction are as follows: cars, aircrafts, spacecrafts, vehicle platooning, cooperative control of unmanned aerial vehicles, tele-operated haptic systems, smart homes, large manufacturing/power plant systems, chemical plants, water distribution networks, distributed power generation networks, intelligent highways, mobile sensor networks, remote surgery, smart phones and networked city services. In the seminal report [15], the *convergence* of *communication*, *computation* and *control* was predicted to be the next phase in (information) technology. The immediate follow-up question was/is: "What is the right abstraction and architecture for this convergence?" The answer to the last question is complex to say the least, and vast literature on the topic to date confirms this. One abstraction and architecture that encapsulates the mentioned convergence is an NCS, which we are going to focus on. Recent report [16] confirms the predictions on the importance and application of NCSs from [15], and puts them again as one of the important directions for the field of control.

Fundamental communication challenges in control applications

Unfortunately, as indicated earlier, the network does not come only with incentives but with (*motivating*) challenges as well. These challenges come from the intrinsic network communication phenomena such as delays, quantization, packet dropouts, scheduling and time-varying transmission intervals. These network imposed communication issues can not only degrade the performance of a corresponding (networked) control system but can lead to instability. Of course, some of them are easier to mitigate than the others and some are important only within certain networks and applications; for instance, the quantization issue can be quite important in a case when CAN bus is used due to only 8 bytes for useful data while in a case of FlexRay network it can be usually ignored due to a much larger useful data field (254 bytes). Considering all known communication issues that a network introduces, at the same time, usually, amounts to an extremely complex system. Designing and/or analyzing the corresponding system, if possible at all, leads to very challenging analysis and design problems. Another, a more systematic and more fruitful approach is *the divide and conquer* approach, where one considers only few issues and investigates them at a deeper level. In our case this translates in concentration on two network induced issues (even this will result in a hard problem). Namely, we focus on the issue of packet dropout and the issue of scheduling; in what follows we motivate why we consider specifically these two.

In control applications, the unavailability of current control value and/or measurement will degrade the performance of the corresponding system and may even cause the instability. Due to the nature of how NCSs communicate the corresponding control data, this scenario is very likely to happen due to a packet dropout. Indeed, networks are designed to have few dropouts on average, and in "non-control" communication applications, the usual practice is to resend the data when a dropout occurs. However, this usually is insufficient for the purposes of control. Moreover, dropouts are *inevitable*, see for instance [17–20] on how the throughput is affected by packet dropouts due to excessive delays, failed transmissions, congestions and/or collisions. Hence, packet dropouts *can not* be avoided and have to be *properly* addressed for control purposes. This makes them very important and, thus, we have paid a special attention to this issue in this thesis.

Depending on the network load, the scheduling issue might not be avoided in the same manner as the issue of packet dropouts. Moreover, this issue can cause packet dropouts if not properly addressed. However, besides potentially causing packet dropouts, the scheduling issue on its own, is also very interesting and important since it tackles the question: "How much network resources do I really need to control a system?" Moreover, it induces the computation related questions, e.g.: "How much computation on controller side can free network resources for other processes and systems?"

In summary, NCSs offer many improvements over "traditional" control systems and from the point of view of abstraction and architecture they encapsulate what is believed to be the next technological leap in information technology. They do impose some challenging analysis and design problems caused primarily with intrinsic network communication phenomena. Out of many network induced communication issues, we focused on packet dropouts and scheduling and in the sequel we provide a methodology we used to mitigate their undesirable effects.

2.3 Methodology

One degree of freedom one can use to address the networked induced communication issues in an NCS is its *flexible architecture*. Namely, as pointed out in the previous section, the addition of *new elements (devices)* to an existing NCS is quite easy. Correspondingly, this can enable *distributed computation* which can be used to address the aforementioned issues; see Fig. 2.4 for a general NCS including these devices.

Figure 2.4: General NCS with extra devices.

In this thesis, we focus only on packet dropouts and scheduling. Both issues will most-likely deteriorate the performance of the system or even cause the instability. The available approach for dealing with packet dropouts in non-control communication ap-

plications, i.e., *resend the data*, usually is not the best idea for control applications. Namely, this is due to the fact that the *old* measurements or control values are typically not useful and it is better to discard lost data and transmit the *new* data. On the other hand, scheduling must be properly addressed since it can lead to congestion which can cause delays and packet dropouts and thus harm the system.

As indicated above, augmenting an NCS architecture with *devices* that enable *distributed computation* can be used to address packet dropouts and scheduling. A natural question is which device(s) to add and where to place it (them).

In order to provide *an* answer to this question let us first briefly consider what happens when a packet dropout occurs; for simplicity, let us focus on the scenario where a controller sends a packet to the plant input. The information is lost and the corresponding plant input lacks the control packet. An ad hoc fix can be to apply last received value or some fixed value (e.g., a zero value), and perhaps for some applications this might do the job. But then, what if we have consecutive packet dropouts. Would the previous approach still work and if so for how large classes of systems. Another approach, that appears to be more natural in this setting, is to use *predictions*. Namely, let us imagine if we had a sequence of predicted control values over a finite horizon which we could store in a device (placed before the plant input) which would apply these predictions in the case of a packet dropout(s). A simple device that enables this sort of simple computation, and which we will use, is a buffer, e.g., a parallel-in-serial-out shift register. Scheduling, on the other hand, is addressed by having a buffer in front of each plant input.

Using devices (buffers) which enable distributed computation is one part of methodology for addressing packet dropouts and scheduling. The other part is using a control paradigm that can generate control predictions over a finite horizon which *accounts* for packet dropouts and scheduling. In our case this will be the Model Predictive Control (MPC) paradigm. Unfortunately, inclusion of a network in most cases makes many well developed control tools for analysis and design inapplicable. Luckily, certain tools remain applicable with relatively small changes to account for the network part and MPC is one of them.

Roughly speaking, MPC is a control framework which involves solving on-line open-

loop finite-horizon optimization problem at each sampling time instant. The current measurement of the state of the plant is used as the initial state in the predictive model used in the optimization. The result of the optimization is a sequence of optimally predicted control values where the length of the sequence corresponds to a finite prediction horizon. Finally, the usual practice (in non-networked applications) is to apply the first element of this control sequence and repeat the process ad infinitum.

More precisely, the *model* of the *plant* is usually given in discrete-time as

$$\Sigma_{\mathfrak{p}}^{\mathfrak{m}} : \tilde{x}_{\mathfrak{p}}(k+1) = f_{\mathfrak{p}}(\tilde{x}_{\mathfrak{p}}(k), \tilde{u}_{\mathfrak{p}}(k))$$

where: $\tilde{x}_{\mathfrak{p}} \in \mathbb{R}^n$ is the state and $\tilde{u}_{\mathfrak{p}} \in \mathbb{R}^m$ is the input of the *plant model*, $\mathbb{R}$ is the set of real numbers, n and m are, respectively, the numbers of plant states and inputs, k belongs to nonnegative integers and finally, $f_{\mathfrak{p}} : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$. Now, at each sampling time instant the plant model is initialized by the measurement of the plant state $x_{\mathfrak{p}}(k)$ and is used for obtaining the corresponding predictions which are penalized via a cost function given as

$$\begin{aligned} &J\big(x_{\mathfrak{p}}(k), \{\tilde{u}_{\mathfrak{p}}(k+i)\}_{i=0}^{\mathfrak{h}-1}\big) \\ &:= \sum_{i=0}^{\mathfrak{h}-1} l\big(\phi_{f_{\mathfrak{p}}}\big(k+i, x_{\mathfrak{p}}(k), \{\tilde{u}_{\mathfrak{p}}(k+i)\}_{i=0}^{\mathfrak{h}-1}\big), \tilde{u}_{\mathfrak{p}}(i)\big) + g\big(\phi_{f_{\mathfrak{p}}}\big(k+\mathfrak{h}, x_{\mathfrak{p}}(k), \{\tilde{u}_{\mathfrak{p}}(k+i)\}_{i=0}^{\mathfrak{h}-1}\big)\big) \end{aligned}$$

where:

- $x_{\mathfrak{p}}(k)$ is the plant measurement at time instant k,
- $\{\tilde{u}_{\mathfrak{p}}(k+i)\}_{i=0}^{\mathfrak{h}-1} := \{\tilde{u}_{\mathfrak{p}}(k), \ldots, \tilde{u}_{\mathfrak{p}}(k+\mathfrak{h}-1\}$ is a sequence of predicted control values over a finite prediction horizon $\mathfrak{h}$,
- $\mathfrak{h}$ is an integer number greater than one which represents a finite prediction *horizon*,
- $\phi_{f_{\mathfrak{p}}}(\cdot)$ is a solution mapping for plant model under the influence of $\{\tilde{u}_{\mathfrak{p}}(\cdot+i)\}_{i=0}^{\mathfrak{h}-1}$,
- $l : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}$ is a *stage cost* function,
- $g : \mathbb{R}^n \to \mathbb{R}$ is a *terminal cost* function.

This cost function is minimized with respect to $\{\tilde{u}_{\mathfrak{p}}(k+i)\}_{i=0}^{\mathfrak{h}-1}$ (and usually some constraints which we omit for simplicity) resulting in an *optimal value* function as

$$V\big(x_{\mathfrak{p}}(k)\big) := \min_{\{\tilde{u}_{\mathfrak{p}}(k+i)\}_{i=0}^{\mathfrak{h}-1}} J\big(x_{\mathfrak{p}}(k), \{\tilde{u}_{\mathfrak{p}}(k+i)\}_{i=0}^{\mathfrak{h}-1}\big).$$

Finally, from the optimal value function one extracts the sequence of optimal control values via

$$\{u_{\mathfrak{p}}^{*}(k+i)\}_{i=0}^{\mathfrak{h}-1} := \operatorname*{argmin}_{\{\tilde{u}_{\mathfrak{p}}(k+i)\}_{i=0}^{\mathfrak{h}-1}} V\big(x_{\mathfrak{p}}(k)\big),$$

and usually applies *only* the first value from the sequence which defines the *implicit* MPC control law as

$$\kappa(x_{\mathfrak{p}}(k)) := u_{\mathfrak{p}}^{*}(k).$$

In our case, the optimization problem will *account* for packet dropouts and scheduling, and we will use the *whole* sequence of optimally predicted control values. Namely, we will send it over a network to the corresponding buffer, and thus address packet dropouts and scheduling. Notice that because we are sending a sequence of control values instead of only a control value, we might potentially put a higher burden on the corresponding network and perhaps even increase the likelihood of a packet dropout. More precisely, we would need to use more data to transmit a sequence of control values. However, many modern data-like networks, such as Ethernet, have large payloads which can accommodate the need for extra data. Indeed, sending only a control value might require less network resources but as mentioned above dealing with packet dropouts becomes much harder and it can often lead to instability. Also notice that in some applications sending only a control value will use the same packet as sending a sequence of control values since data packets are of fixed size, see [21]. Now, in comparison to Fig. 2.4, our methodology, conceptually, can be schematically depicted as in Fig. 2.5. Important part to notice from Fig. 2.5 is that we assumed that the controller has direct access to measurements of plant state. This approach is taken since accurate plant measurements are crucial

in making predictions in MPC framework; e.g., see [107] for in-depth explanations.

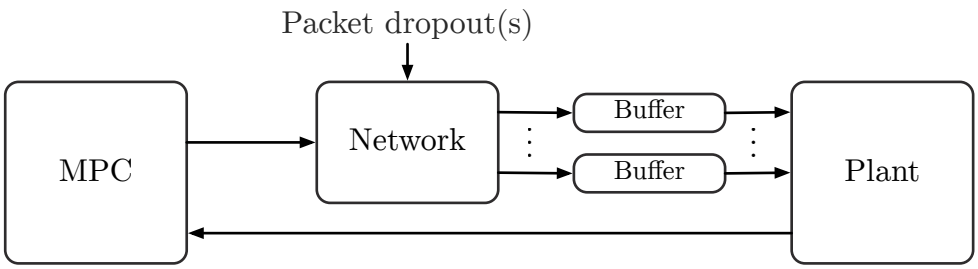


Figure 2.5: Schematic methodology for addressing packet dropouts and scheduling.

2.4 Literature review

We consider NCSs and we use an MPC framework to mitigate the undesirable effects of packet dropouts and scheduling. Both topics, NCSs and MPC, are vibrant research areas on their own. Hence, we first overview each separately and then review papers that considered MPC for NCS, i.e., *packetized* MPC.

2.4.1 Networked Control Systems

The literature on NCSs is vast, e.g., see the books [22–24], survey papers [10–14] and special issues [25–28]. Usually, the literature on NCS revolves around networks, or more precisely, around network induced issues. This enables one to compile a table[2] of typical references for the corresponding issues, see Table 2.1. Besides the latter table, additionally, we will provide the basic overview of the most addressed network induced issues. However, largely we will focus on the issue of packet dropouts and scheduling since they are considered explicitly in this thesis; before we begin, we note that we have used the recent survey paper [14] to provide the basic overview of the most addressed network induced issues.

[2]This table is compiled by closely following [14].

Packet dropout

Roughly, most of the analysis and design for mitigating the effects of packet dropouts relies on two strategies which differ in how the current packet dropout affects the corresponding controller.

One strategy is packetized MPC and it is the strategy considered in this thesis. More precisely, one computes the predictions over a finite horizon and applies the predicted control values in a case of a packet dropout. To be able to apply the predicted control values one needs actuators with some computational power and memory. This is due to the fact that the corresponding predictions have to be stored and applied when dropout occurs. More precisely, one starts by assuming a finite upper bound on the number of consecutive packet dropouts. Then, an optimization problem over a finite horizon, which corresponds to the assumed upper bound on the number of consecutive packet dropouts, is solved. The result is a sequence of optimally predicted control values, which is sent over a network to a device which can store this sequence and apply the predicted values in the case of a packet dropout(s). The origin of this idea is considered to be [29] and some recent works that have used this idea are [30–32]. Related approaches in a sense that the current packet dropout dictates which controller is being applied, can be found in, for instance, [33–35]. In [33] and [35], the correlation between packet dropout and no packet dropout is modeled as a two-dimensional Markov chain, resulting with a Markov jump system. The corresponding design results with two controllers which are applied depending if packet dropout occurred or not. This approach can be replaced with a nondeterministic switched systems approach as described in [34].

Another strategy is to design a stabilizing controller which is valid as long as the number of consecutive packet dropouts is less than a given bound, e.g., see [36–42]. Note that here, regardless of how many consecutive packet dropouts there are, the same controller is still applied; that is, actual packet dropouts do not change a controller and the only requirement is that the number of consecutive packet dropouts is less than a given bound. Now, depending, for instance, on the dynamics of the plant and how one models packet dropouts, different tools and approaches are applied for the corresponding analysis and design. For instance, in [37], focusing on the stability analysis and controller design of

the corresponding NCS, authors considered linear plants and determined a stabilizing memoryless controller via delay-dependent approach. In [39], authors used a linear matrix inequality (LMI)-based procedure for designing a state-feedback controllers, which guarantee that the output of the closed-loop networked control system tracks the output of a given reference model well in the $\mathcal{H}_\infty$ sense. Further, in [40], the packet losses are modeled as a linear function of a stochastic variable satisfying Bernoulli random binary distribution, resulting with a stochastic description of the corresponding stability. Other approaches that also use probability tools can be found in [36,38,41–43].

Scheduling

A natural approach to address the issue of scheduling is to design and implement a scheduling protocol. One of the early references that investigates stability of NCSs, with respect to scheduling issue, by employing scheduling protocols, are [44–46]. There, two special cases of protocols were considered. Namely, they considered a static Round-Robin and a dynamic Try-Once-Discard protocol. The former protocol operates in a static fashion (token-ring-type scheduling), meaning that the corresponding schedule is predefined and nodes get access according to this schedule. Usually, this schedule is repeated ad infinitum. On the other hand, for the latter protocol one needs to define an error, usually as the difference between the current value of the sensor/actuator signal and its previous value. Then, usually, a maximum-error-first (MEF) scheduling policy is employed, implying that the node with the greatest weighted error from its last reported value will be granted access to network. For both protocols, bounds on maximally allowable transmission interval (MATI) are provided which ensure global exponential stability. Finally, note that an emulation approach was used for designing a stabilizing controller, i.e., it is designed a priori without the network included into the loop. Extensions and generalizations of these results are provided in [47–50] where it was also discovered that RR and TOD protocols are in fact special cases of Uniformly Globally Exponentially Stable (UGES) protocols. Further, stochastic protocols were considered in [51] while controllability questions with respect to the scheduling issues were addressed in, e.g., [52–56].

In the sequel, even though we do not address the issue of delays, quantization and

time-varying transmission interval in this thesis, due to their importance, we discuss briefly.

Delays

How the delay is modeled depends on the corresponding application and used network(s). Consequently, this dictates analysis and design of the corresponding NCSs. For instance, in Controller Area Network (CAN) protocol or in case when buffers are used at network nodes, delays can be considered constant. On the other hand, the delay induced by Ethernet is typically time-varying. Furthermore, some applications might allow for deterministic delay models while others require stochastic delay models. Additionally, in some applications only an upper bound on delay might be sufficient for the corresponding analysis and design.

The type of delay considered, coupled with the corresponding NCSs, gives rise to different control strategies and methodologies. Up until the year 2003, the survey paper [10] covers very nicely the typical methodologies in analysis and design of NCSs with respect to delays such as: LQR, hybrid system and perturbation approach, to name a few. Then, the survey paper [12] covers the period from 2003 to 2007 including delayed differential equations approach and switched systems approach among others. Worth mentioning is also the survey paper [11] which includes two Markov chain models. Finally, the recent survey paper [14] classifies two frameworks with respect to delays, namely, a robustness and an adaptation framework. The former framework corresponds to a case where control design is robust with respect to time-varying delays, e.q., see [57–60] for details. The latter framework exploits modern network attributes such as sending data in packets (which can be relatively large) and/or time-stamping of messages. Additionally, some references exploit actuators which posses some computational power and memory, for instance see [29, 33, 34, 61–66].

Quantization

Among many results documented, two approaches stand out. The first one revolves around memoryless feedback quantizers or also denoted as static quantization policies, [67–70]. In such policies, data quantization at time k depends on data at time k only, resulting in relatively simple structures for coding/decoding schemes. One of the first treatments of control systems with uniform quantized feedback is documented in [67]. Later, in [68], in order to stabilize a linear system, a bound on the number of quantization intervals was imposed. Further, in [69], authors showed that for a quadratically stabilizable discrete-time single-input single-output (SISO) linear time-invariant systems, the quantizer needs to be logarithmic. Following this work, authors of [70], gave a comprehensive study on feedback control systems with logarithmic quantizers by using the sector bound approach to quantized feedback control.

The second approach employs a class of dynamic quantizers which dynamically scale the quantization levels according to the location of the corresponding state, [71, 72]. Relaying on this approach, different scenarios were considered. The treatment of both, input and output quantization, is documented in [73]. Consideration of exogenous disturbances in the quantized control systems can be found in [74] while the nonlinear systems models are covered in [75]. In [76], authors provide a unified framework for the design of NCSs with simultaneous existence of the scheduling and the quantization issue.

Even though, recently, less attention is being given to this issue, justifying it with assumption on sufficiently large data packets which is true for certain networks, e.g., Ethernet and FlexRay. However, in case of CAN one still needs to think about this issue due to the fact that only 8 bytes can be used for data. However, regardless which network is used, this issue is still very important for its fundamental values.

Time-varying transmission interval

The time-varying sampling problem has been a major topic in classical sampled-data systems. For some relatively recent results, for instance, see [77–80].

With the emergence of NCSs, time-varying transmission interval problem has drawn

research attention once again. For instance, plant outputs are transmitted at instants that may vary significantly due to the network, e.g., due to congestion, which results in time-varying transmission intervals. To date, many results have been published, concentrating mainly on stability. For emulation type design of time triggered networks, for instance see [36, 81]. On the other hand for a model-based approach, e.g., see [82] and [83]. Other approaches, such as hybrid system approach, and a more detailed review of model-based approach can be found in survey paper [12].

Besides these five network induced issues there are many more, such as network security, for instance, which are becoming more popular. Finally, a lot of research is dedicated at considering combinations of the issues discussed above, and we have provided typical references dealing with these combinations in Table 2.1. Thus, we will stop the NCS literature review here and proceed with literature review for MPC.

2.4.2 Model Predictive Control

The available literature on MPC to date is vast. Good starting references are books [104–109], survey papers [110–121], papers [122, 123] and references therein.

Many properties and applications of MPC, such as stability, robustness, tracking, output feedback, soft constraints, adaptation and optimization algorithms, have been and are still being investigated. However, we will briefly concentrate only on the stability and robustness of MPC since these two properties are considered in this thesis. Furthermore, we will also provide a very brief review of the so-called Economic MPC since this also is to some extent covered in this thesis. Finally, note that other topics related to MPC are well documented and one can start by consulting the references presented in the first paragraph of this subsection.

Over the years, many approaches for establishing stability have been reported. However, as discovered in [119], a consensus has been reached by many researchers on the essential ingredients that ensure closed-loop stability when MPC is employed. This made stability for MPC a mature topic. These ingredients are: a terminal cost, a terminal constraint set and a local stabilizing controller. Then, choosing an optimal value function as a Lyapunov function and satisfying certain conditions on the ingredients listed above,

Issue	References
Delay	[29, 33, 34, 57–66]
Packet dropout	[31, 33–42]
Packet dropout & Delay	[84–89]
Time-varying transmission interval	[36, 82, 83]
Time-varying transmission interval & Delay	[90–92]
Time-varying transmission interval & Delay & Scheduling	[21]
Time-varying transmission interval & Delay & Packet dropout	[93, 94]
Scheduling	[45, 47–50, 95]
Scheduling & Delay	[96, 97]
Scheduling & Packet dropout	[51]
Scheduling & Delay & Packet dropout	[98, 99]
Quantization	[67–72]
Quantization & Delay	[100, 101]
Quantization & Packet dropout	[102, 103]
Quantization & Delay & Packet dropout	[39, 87]
Quantization & Scheduling	[76]

Table 2.1: Typical references for network-induced issues.

ensures the stability; for instance, see [104, 117, 124–134] for various approaches. One of the underlying requirements in many approaches, is that a terminal cost is a control Lyapunov function (CLF). A drawback of these approaches is the need for an a priori computation of a CLF, though, when the linearization is stabilizable, it is easy to find a CLF. An approach that does not require the terminal cost to be a local CLF is reported in [123], which, as authors state, can be seen as discrete-time counterparts of those in [135] for the class of systems investigated in [136].

Stability robustness of the systems governed by MPC has been relatively well investigated; e.g., see Section 4 in [119]. The simplest approach is to rely on inherent robustness as documented, for instance, in [132, 133, 137]. However, this is not always enough as reported in [122]. Another approach, referred to as an open-loop model predictive control,

is to include the uncertainty in both, cost minimization and constraint satisfaction; e.g., see [113, 127, 138–140]. However, this approach cannot contain the "spread" of predicted trajectories, leading to conservative or, even, infeasible solutions of the corresponding uncertain optimal control problem. To overcome these issues, a feedback model predictive control is investigated, for instance, in [113, 114, 139–144]. Note that both previous approaches provide a feedback control. However, whereas in open-loop MPC the decision variable is a sequence of control actions, in feedback MPC it is a policy which is a sequence of control laws. Determining a control policy is usually more difficult. Thus, a lot of research effort was/is directed towards simplifying the feedback optimal control problem. Some relatively recent results that (to some extent) overcome the corresponding disadvantages of both methods are reported, for instance, in [145, 146].

We finish the MPC literature review with a brief overview of Economic MPC. Let us start by noting that in a standard MPC, the corresponding stage cost function penalizes either the distance to a desired equilibrium (stabilization) or time-varying reference solution (tracking). On the other hand, one might wish to consider a more general stage cost function that potentially can encapsulate an economic criterion in its literal sense such as an economical cost (e.g., money) related to controlling a plant; for instance see [147, 148]. Nice examples on how standard and Economic MPC differ are provided in [147]. However, to be mathematically precise, let us borrow the discussion from [149]; see the second paragraph in the second section in [149]. For the standard MPC the following holds for the corresponding cost function $l : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}$

$$0 = l(0,0) \leq l(x,u) \text{ for any admissible } (x,u), \tag{2.1}$$

where x is the state and u is the input of the corresponding plant[3]. On the other hand, for the Economic MPC, the inequality (2.1) cannot be generally assumed and the following can happen

[3]We assume that the origin is the equilibrium of the corresponding system, i.e., we would apply the appropriate change of coordinates (if necessary).

$$\left.\begin{array}{r} 0 < l(0,0) \text{ or,} \\ l(0,0) > l(x,u) \text{ for some admissible } (x,u), \end{array}\right\} \tag{2.2}$$

where (x, u) do not correspond to any equilibrium point. Unfortunately, this fundamental difference between the standard and Economic MPC is sufficient that the extensive collection of results for the standard MPC stability analysis do not simply extend to Economic MPC. However, a noticeable progress has been made, see for instance [147–156]. Most of the first stability results are established with the aid of the terminal constraint. Usually, an optimal equilibrium is determined and the system is stabilized by forcing a state to be equal to this equilibrium at the end of the horizon, e.g, see [147, 148, 150, 152, 154, 155]. Recently, this has been relaxed by imposing a region constraint on the terminal state instead of a point constraint, and by adding a penalty on the terminal state to the regulator cost [153]. Moreover, the stability analysis tools, developed for terminal constraint economic MPC, were extended and it was established that strict dissipativity, introduced in [149], is sufficient for guaranteeing asymptotic stability of the closed-loop system. An approach that does not need any terminal constraint can be found in a very recent reference [156]. However, the trade-off for removing terminal constraint requires a more involved analysis with stronger assumptions and getting an approximate optimal performance instead of an exact as in [150, 152, 153].

We proceed with a more detailed review of references that are more directly related to this thesis.

2.4.3 Packetized MPC

Before we begin, let us recall that we focus on two network induced issues. Namely, we considered the issue of packet dropouts and the issue of scheduling. To address these issues we use an MPC framework. One aspect of this framework is that it enables one to incorporate the mentioned issues into the predictive model. This predictive model is used by an MPC controller to generate the corresponding sequence of optimally predicted control values over a finite horizon. Another aspect of MPC framework is the possibility to exploit (if the corresponding control system allows) distributed computa-

tion; recall that the flexible nature of NCSs allows for distributed computation. Namely, we use an extra device with computational capabilities and memory (buffer). This device is used to store the sequence of predicted control values and to apply the corresponding values in the case of packet dropouts. Moreover, if we add the issue of scheduling, this device is used to apply the corresponding predicted control values not only in the case of packet dropouts but also if the corresponding input was not updated due to scheduling. Similar approaches have been considered previously for addressing the two mentioned network induced issues. Thus, we proceed with a review of references that used similar approaches to establish stability, robustness and/or controllability of similar NCS architectures.

Stability with respect to packet dropouts and scheduling – Chapter 3

The idea of exploiting more than just the first element of the sequence of optimally predicted control values over a finite horizon was proposed in [29]. There, the considered network induced issue was an *unbounded delay* in a Internet TCP/IP (Transmission Control Protocol/Internet Protocol) communication links. The considered NCS architecture is depicted in Fig. 2.6 where the controller consists of an MPC controller which takes as an input human operator instructions and the output of a predictor. The piece to focus on is a buffer which stores the sequence of optimally predicted control values and applies them in a case of excessive delays (i.e., a packet dropout).

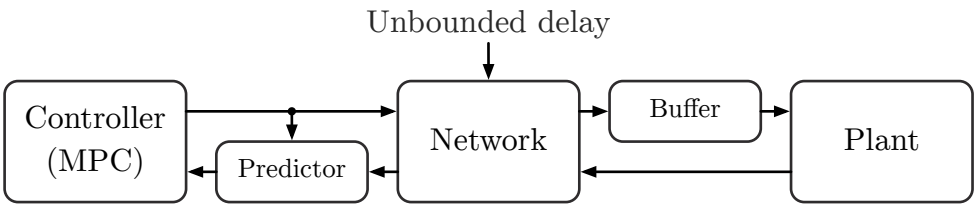


Figure 2.6: An NCS architecture for addressing unbounded delay.

This idea of using a buffer at plant input and an MPC controller was later pursued, for instance in [157]. To some extent it was extended in [30] to address not only the issue of packet dropouts but also the issue of scheduling. The corresponding NCS architecture

is depicted in Fig. 2.7.

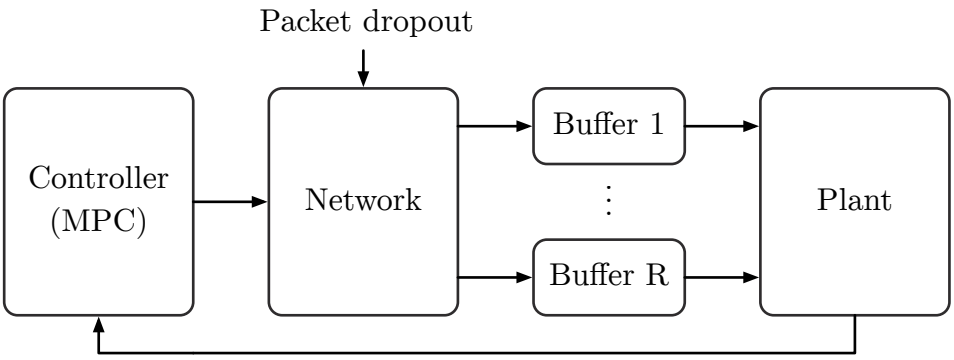


Figure 2.7: An NCS architecture for addressing the issues of packet dropouts and scheduling.

The network is affected by packet dropouts depicted as disturbance and is such that allows access to *only one* plant input at each time instant. Due to scheduling, there are now R buffers, one for each input. The buffers have a similar role as above. Namely, they are used to store the sequence of optimally predicted control values and apply them in case of a failed transmission, i.e., a packet dropout. However, due to scheduling, only one plant input can be addressed at each time instant. Hence, if a buffer is not being updated at the current time, it keeps using the values it has stored previously that originate from previous optimization. How one obtains the optimal node and the corresponding sequence of optimally predicted control values will be presented later in the thesis in detail, see Chapter 3. At this stage, important to mention is that the underlying requirement for showing the convergence of plant states was to assume the existence of a finite bound on the number of consecutive packet dropouts plus some modification of standard MPC stability assumptions; e.g., see [119] and/or Section 3 for standard stability assumptions for non-networked MPC

Stability with Economic MPC with respect to packet dropouts and scheduling – Chapter 4

As documented in, for instance, [147, 149–156], the existing stability analysis for a standard MPC, unfortunately, does not extend to Economic MPC. Thus, new tools had to be

developed for stability analysis for Economic MPC. These early tools were developed for the terminal constraint MPC formulation, in which the system is stabilized by forcing the state to the best equilibrium point at the end of the horizon. More details on these and some recent developments can be found in this collection of papers [147, 149–156]. We proceed with the focus on [153] where a relaxation of imposing a region constraint on the terminal state instead of a point constraint, and adding a penalty on the terminal state to the regulator cost, is established. Moreover, the stability analysis tools, developed for terminal constraint economic MPC, were extended. The enabling assumptions is the so-called *strict dissipativity* of a system introduced in [150]. This together with Lyapunov tools, proved to be sufficient for establishing stability of the corresponding system.

Robustness with respect to packet dropouts and scheduling – Chapter 5

To address robustness of the same NCS architecture the results from [31, 122, 158] played an important role. As documented in [122] in order to establish nominal robustness the optimal value function and feedback law have to be continuous in the interior of the feasibility region. Additionally, a simpler, but closely related, NCS architecture was considered in [31], see Fig. 2.8.

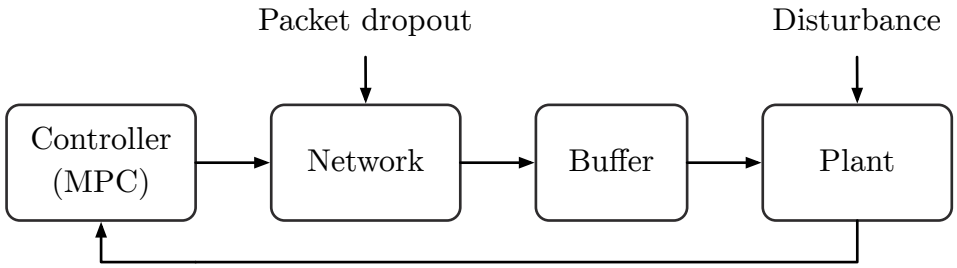


Figure 2.8: An NCS architecture for addressing the issues of packet dropouts.

There, only the issue of packet dropouts was considered. The key assumptions were the uniform finite upper bound on the number of consecutive packet dropouts and the continuity assumption on the optimal value function.

Controllability with respect to scheduling – Chapter 6

We have already seen that network-induced issues can affect performance and stability of the corresponding system. An interesting question is: "How does it affects controllability of the corresponding NCS?" If it destroys it, can it be recovered by an appropriate NCS structure?

The network-induced issue we focused on is scheduling. Some recent relevant references that address the same question are [52–55]. In [55] a NCS with a Single-Input-Single-Output (SISO) linear plant was considered and the effects of the so called *blind* periods in communications were investigated. The authors do not consider scheduling explicitly but the result can be seen as a special case of [56]. In [53] and [54], Multi-Hop Control Networks with Multiple-Input-Multiple-Output (MIMO) and SISO linear plants, respectively, were considered providing conditions for the controllability of the corresponding systems.

2.5 Overview and contributions

Chapter 3

We begin with a focus on stability of the corresponding NCS architecture with respect to packet dropouts and scheduling. More precisely, considering the same NCS architecture as in Fig. 2.7 we established Uniform Global Asymptotic Stability (UGAS) of the augmented state of the plant and buffer state [159]. Important to note is that the established property is much stronger than the property in [30] with slightly stronger assumptions. The enabling assumption is the lower bound on the stage cost function in terms of control, e.g., see [160]. Namely, this lower bound is a $\mathcal{K}_\infty$ function *only* of an absolute value of the control. Note that it is a standard stability-related assumption in MPC to have a lower bound on the stage cost function which is a $\mathcal{K}_\infty$ function of an absolute value of the plant state. Our additional assumption is an assumption which is just not frequently used.

To establish UGAS, advanced nonlinear system analysis techniques [2, 161, 162] were

used. We established the property via two approaches. One involves finding an appropriate Lyapunov function. The other one uses a cascade idea. The key fact that leads to this is the observation that the *overall* buffer (collection of all buffers, precisely defined later in the thesis) as a system is Input-to-State Stable with respect to the input being the arriving control sequence. Moreover, the overall buffer is UGAS uniformly in buffer states, e.g., ISS with zero gain, which enables one to consider the corresponding NCS as a cascade and use the corresponding stability analysis. The last two observations plus the fact that the plant trajectories converge (see [30]) plus our extra assumption and the mentioned advanced control techniques enabled us to establish the result. Additionally, the techniques used in the proofs are novel, providing an alternative approach for addressing stability properties for similar NCS architectures.

Chapter 4

This chapter focuses on stability of a NCS governed by Economic MPC with respect to packet dropouts and scheduling. Inspired by the results in [153], we first extend the definition of strict dissipativity introduced in [149, 153]. Then, we use the ideas documented in [153] to formulate the corresponding stage and terminal cost function so that the results we established for Standard MPC in Chapter 3 can be applied with mild changes in the assumption related to terminal control law. Namely, there is an extra term in the corresponding inequality. The novelty we introduce is an extension of some results documented in [153] and a stability result for a NCS governed with an Economic MPC.

Chapter 5

In this chapter we concentrate on robustness of a corresponding NCS architecture with respect to packet dropouts and scheduling.

Namely, we use the insights from [31] and the concept of nonlinear gains introduced in [158]. The contributions of this chapter are as follows. We capture robustness of the NCS architecture depicted in Fig. 2.9 via partial nonlinear gain ℓ_2 stability notion (defined precisely later in the thesis). Using appropriate assumptions lead to a more traditional

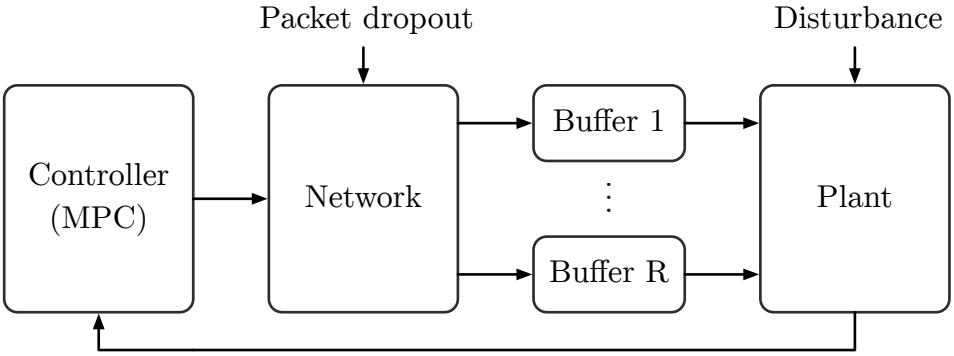


Figure 2.9: An NCS architecture for addressing robustness with respect to the packet dropouts and scheduling.

linear gain ℓ_2 stability notion. For one node case, i.e., no scheduling, we establish alternative robustness characterization of the NCS architecture considered in [31]. Assuming stronger assumptions we establish Input-to-State Stability of the aggregated state of the plant and buffer state. We also recover global asymptotic stability for the disturbance-free system. Additionally, via computer simulation, we demonstrate that the dynamic scheduling outperforms the static one.

Chapter 6

This chapter investigates the property of controllability of the corresponding NCS architecture with respect to scheduling, see [56].

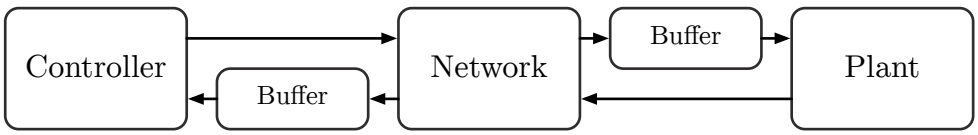


Figure 2.10: An A NCS architecture for addressing controllability.

The considered NCS architecture is depicted in Fig. 2.10. Contributions are as follows. First, we provide a novel model. Then, controllability results for a case where the plant is nonlinear were established, followed by their linear counterparts. For linear case the result from [52] was extended and applied in establishing the corresponding controllability

result from [56].

Chapter 7

Here we concentrate on the implementation of the considered framework and we focus on demonstrating our stability and robustness results. The implementation is carried out as a Hardware-in-the-loop (HIL) simulation. The real hardware is the network, more precisely, Control Area Network (CAN). This network is not designed with large data capabilities. In particular, it can dedicate at most 8 bytes for transmission of a message, which in our case, is a sequence of predicted control values). Correspondingly, we encountered another network induced issue which we did not consider, namely, the issue of quantization. Moreover, we have also encountered the issue of delay, but this issue was easily addressed within MPC. We note that our results are tailored for networks with better data capabilities (e.g., larger data fields for message transmissions), however, the implementation results are still satisfactory, though with deteriorated performance.

2.6 Publications

The following publications have resulted from results in this thesis.

Journal

- Lješnjanin Merid, Quevedo Daniel E. and Nešić Dragan, "Packetized MPC with dynamic scheduling constraints and bounded packet dropouts", *Automatica*, volume 50., number 3, pages 784 – 797, 2014
- Lješnjanin Merid, Nešić Dragan and Quevedo Daniel E., "Uniform Global Asymptotic Stability of Model Predictive Control Governed Networked Control Systems Affected with Packet Dropouts and Scheduling", submitted as a regular journal paper in *Automatica*, June 2015
- Lješnjanin Merid, Nešić Dragan and Quevedo Daniel E., "Implementation of Packetized Control over Control Area Network bus", to be submitted as a regular journal

paper in *IEEE Transactions on Industrial Electronics* in 2015

- Lješnjanin Merid, Nešić Dragan and Quevedo Daniel E., "Controllability of Discrete-time Networked Control System Affected by Packet Dropouts and Scheduling Issues", to be submitted as a regular journal paper in *Automatica* in 2015

Refereed conference

- Lješnjanin Merid, E. Quevedo Daniel and Nešić Dragan, "Robustness of networked control systems with multiple actuator-links and bounded packet dropouts" *52nd IEEE Annual Conference on Decision and Control*, pages 5963 – 5968, 2013

- Lješnjanin Merid, Quevedo Daniel E. and Nešić Dragan, "Controllability of Discrete-Time Networked Control Systems with Try Once Discard Protocol", *19th IFAC World Congress*, pages 3758 –3763, year 2014

- Lješnjanin Merid, Nešić Dragan and Quevedo Daniel E., "Uniform Global Asymptotic Stability of Model Predictive Control Governed Networked Control Systems Affected with Packet Dropouts and Scheduling" submitted to *54th IEEE Annual Conference on Decision and Control*, March 2015

This page intentionally left blank.

Chapter 3

Stability with respect to Packet Dropouts and Scheduling

The considered NCS architecture is depicted in Fig. 3.1. The piece to focus on is a communication network $\Sigma_{\mathfrak{n}}$ which is located between the controller $\Sigma_{\mathfrak{c}}$ output and plant $\Sigma_{\mathfrak{p}}$ inputs. The network is such that it allows access to only one plant input at each time instant which generates scheduling and, furthermore, it is affected with packet dropouts $w_{\mathfrak{n}}$. In order to establish stability with respect to scheduling and packet dropouts, we carry out a controller and protocol co-design; e.g., see [30].

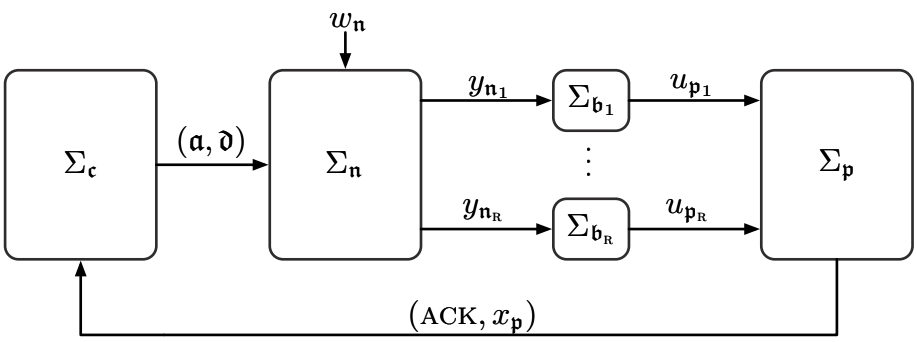


Figure 3.1: An NCS architecture considered for stability.

Succinctly, the considered controller and protocol co-design consists of exploiting the *flexible architecture* of NCSs which allows for *distributed computation* and employing *MPC paradigm* to generate a sequence of *optimally predicted control values* over a *finite horizon* for an *optimal node (plant input)*.

In our case, the distributed computation comes from buffers $\Sigma_{\mathfrak{b}_{\cdot}}$. These are devices which perform very simple computation and have a limited memory; e.g., a parallel-in-

serial-out shift register. In order to explain the function of a buffer(s), let us consider a buffer $\Sigma_{\mathfrak{b}_r}$, $r \in \{1, \ldots, \mathrm{R}\}$ depicted in Fig. 3.2. Furthermore, for the sake of simplicity let us fix the length of a buffer (e.g., number of memory places) to three. Now, let us assume that at time instant $k = 0$ this buffer was addressed and that the corresponding packet (carrying a sequence of optimally predicted control values) was sent. There are two possible outcomes. Either the transmission was successful or, due to a packet dropout, it was not. In the case of a successful transmission, buffer initial content $(x_{\mathfrak{b}_r}(0), x_{\mathfrak{b}_r}(1), x_{\mathfrak{b}_r}(2))$ is being replaced with the packet content, namely, $(u_{\mathfrak{p}_r}(0), u_{\mathfrak{p}_r}(1), u_{\mathfrak{p}_r}(2))$, and the first element of this sequence $u_{\mathfrak{p}_r}(0)$ is applied to the plant. On the other hand, if there was a packet dropout, the buffer content remains in the buffer and the first element $x_{\mathfrak{b}_r}(0)$ is applied to the plant. For the sake of simplicity, let us assume that for the next three consecutive time instances buffer $\Sigma_{\mathfrak{b}_r}$ has not been addressed. During this time, the buffer will shift its content for one spot, place a zero at the last spot and apply the first element to the plant; note that this behavior also applies to buffers not being addressed.

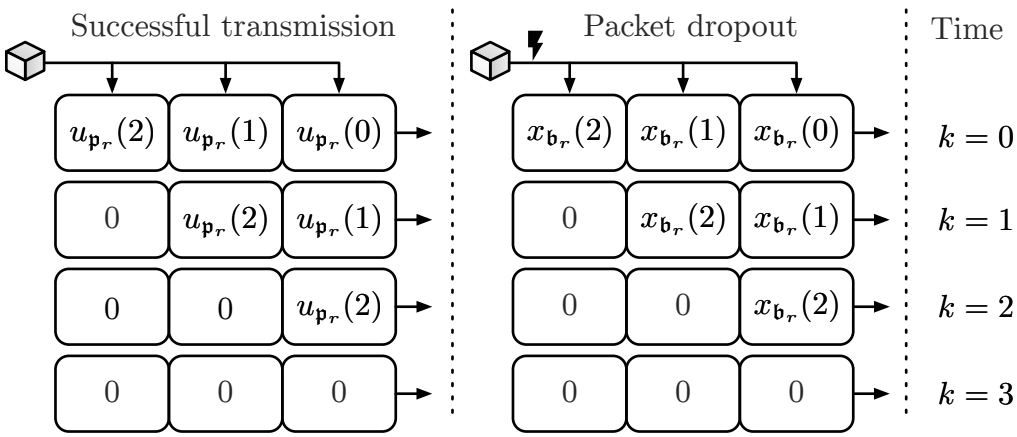


Figure 3.2: Illustration of the function of a buffer.

On the other hand, the MPC framework enables us to define an optimization problem which explicitly takes into account packet dropouts and scheduling. Then, we optimize over control sequences and its corresponding nodes resulting in a sequence of optimally predicted control values over a finite horizon for an optimal node. Finally, this forms a packet that a controller sends over a network to the corresponding buffer.

Theoretically, the considered NCS architecture is important because it provides a plat-

form for deeper understanding of the undesirable effects of packet dropouts and scheduling with the focus on mitigating them. Further, due to technological advancements in electronics in the last decade, both, buffers and microprocessors (on which the controller would be implemented), became affordable and sufficiently fast. Thus, practically, the considered NCS architecture is implementable and potential computational issues, due to on-line optimization, are addressable with current generation of microprocessors. For instance, an application captured with the considered NCS is the control of mobile robots in confined space where the position of robots is measured with sensors directly connected to the controller while the communication between the controller and robots is done through a wireless network.

The considered NCS architecture has been investigated in [30] to some extent. Namely, the authors show the convergence of plant states with respect to scheduling and packet dropout. They mitigated the corresponding undesirable effects via above-described protocol and controller co-design. In particular, they addressed packet dropout(s) with an assumption that there exists a uniform bound on the number of consecutive packet dropouts. This uniform bound is assumed smaller or equal than a finite horizon $\mathfrak{h}$ used in MPC model(s). Moreover, the corresponding optimization problems are formulated with a specific sequence of dropout outcomes. Namely, it is assumed that there is a successful transmission followed with $\mathfrak{h} - 1$ consecutive packet dropouts. Scheduling on the other hand is addressed by formulating R optimization problems which corresponds to the number of plant inputs. For each optimization problem the same sequence of dropout outcomes is considered. Solving these problems results in a set of R optimal value functions, one for each plant input (node). Determining the minimal one generates the optimal node and the corresponding sequence of optimally predicted control values. This is then sent over a network to the corresponding buffer which depending on the real packet dropout either stores the new control sequence or keeps applying its current content. Using the standard stability related assumptions for MPC, e.g., see [119], and the assumption on the uniform bound on the number of consecutive packet dropouts the authors show that the plant states converge in spite of packet dropouts and scheduling.

We adopt this approach and strengthen the corresponding result by showing UGAS

of the augmented state of the plant and buffers state. The enabling assumption is the additional lower bound on the stage cost function in terms of control only, e.g., see [160]. Using this assumption and standard assumptions for showing stability in MPC setup combined with techniques from [2, 161, 162] was sufficient to show UGAS of the augmented state of the plant and buffer state. In fact, we prove the corresponding result in two ways. One way is more standard, where we find an appropriate Lyapunov function. The other way is not so standard and it uses a cascade idea. The enabling observation is that the *overall* buffer (defined in the sequel) as a system is Input-to-State Stable with respect to the input being the arriving control sequence. Combining this observation with the fact that the plant trajectories converge (see [30]), our extra assumption on lower bound on the stage cost function in terms of control only and the mentioned advanced control techniques were sufficient to establish the result. In summary, the novelty and contribution of the findings from this chapter are in the stability property we establish for the corresponding NCS architecture and in the approach we use in the corresponding proofs.

This chapter is organized as follows. In Section 3.1 we present a detailed description of the NCS architecture considered. Section 3.2 consists of stability analysis where first the corresponding assumptions are presented and discussed. Finally, the proofs are provided in Section 3.3.

3.1 NCS Architecture

The considered NCS architecture is depicted in Fig. 3.1. A brief and high-level description of the corresponding architecture is provided in the introduction above. For a more detailed description, due to the complexity of the system at hand, we proceed by considering each part separately.

3.1.1 Plant

We consider discrete-time plants of the form

$$\Sigma_{\mathfrak{p}} : x_{\mathfrak{p}}(k+1) = f_{\mathfrak{p}}(x_{\mathfrak{p}}(k), u_{\mathfrak{p}}(k)),\ k \in \mathbb{N}_0, \tag{3.1}$$

where $x_{\mathfrak{p}} \in \mathbb{R}^{n_{\mathfrak{p}}}$ is the state and $u_{\mathfrak{p}} \in \mathbb{R}^{m_{\mathfrak{p}}}$ is the input of the plant with all elements of $\{n_{\mathfrak{p}}, m_{\mathfrak{p}}\}$ belonging to natural numbers. The mapping $f_{\mathfrak{p}} : \mathbb{R}^{n_{\mathfrak{p}}} \times \mathbb{R}^{m_{\mathfrak{p}}} \to \mathbb{R}^{n_{\mathfrak{p}}}$ is general nonlinear.

Remark: Note that we consider discrete-time plants since we assume equidistant transmission instants. Indeed this might be applicable to a smaller set of networks but the analysis becomes simpler and moreover it is a good starting analysis towards considering networks with time-varying transmission instants.

As stated in Chapter 1, whenever appropriate, we will use a succinct notation for evolution equations, e.g., we rewrite (3.1) as

$$\Sigma_{\mathfrak{p}} : x_{\mathfrak{p}}^{+} = f_{\mathfrak{p}}(x_{\mathfrak{p}}, u_{\mathfrak{p}}). \tag{3.2}$$

Furthermore, without the loss of generality, we assume that the equilibrium of the system $\Sigma_{\mathfrak{p}}$ is at the origin, more precisely

$$x_{\mathfrak{p}}^{e} = f_{\mathfrak{p}}(x_{\mathfrak{p}}^{e}, u_{\mathfrak{p}}^{e}) = f_{\mathfrak{p}}(0^{n_{\mathfrak{p}}}, 0^{m_{\mathfrak{p}}}) = 0^{n_{\mathfrak{p}}}. \tag{3.3}$$

Recall that this is always achievable by appropriate change of coordinates. For instance, let the "original" system be given as

$$\Sigma_{\mathfrak{p}}^{o} : x_{\mathfrak{p}_o}^{+} = f_{\mathfrak{p}_o}(x_{\mathfrak{p}_o}, u_{\mathfrak{p}_o}),\ x_{\mathfrak{p}_o}^{e} = f_{\mathfrak{p}_o}(x_{\mathfrak{p}_o}^{e}, u_{\mathfrak{p}_o}^{e}), \tag{3.4}$$

where $x_{\mathfrak{p}_o} \in \mathbb{R}^{n_{\mathfrak{p}}}$ is the state, $u_{\mathfrak{p}_o} \in \mathbb{R}^{m_{\mathfrak{p}}}$ is the input and $x_{\mathfrak{p}_o}^{e}$ is the equilibrium point, not necessarily the origin; with the mapping $f_{\mathfrak{p}_o} : \mathbb{R}^{n_{\mathfrak{p}}} \times \mathbb{R}^{m_{\mathfrak{p}}} \to \mathbb{R}^{n_{\mathfrak{p}}}$ being general nonlinear. Then, with change of coordinates

$$\left.\begin{aligned} x_{\mathfrak{p}} &= x_{\mathfrak{p}_o} - x_{\mathfrak{p}_o}^{e}, \\ u_{\mathfrak{p}} &= u_{\mathfrak{p}_o} - u_{\mathfrak{p}_o}^{e}, \end{aligned}\right\} \tag{3.5}$$

one arrives at (3.1) (see also (3.2)) with the corresponding equilibrium satisfying (3.3

where

$$f_{\mathfrak{p}}(x_{\mathfrak{p}}, u_{\mathfrak{p}}) := f_{\mathfrak{p}_o}(x_{\mathfrak{p}} + x^e_{\mathfrak{p}_o}, u_{\mathfrak{p}} + u^e_{\mathfrak{p}_o}) - f_{\mathfrak{p}_o}(x^e_{\mathfrak{p}_o}, u^e_{\mathfrak{p}_o}). \tag{3.6}$$

Finally, the plant control input is *partitioned* according to

$$u_{\mathfrak{p}} := (u_{\mathfrak{p}_1}, \ldots, u_{\mathfrak{p}_{\mathrm{R}}}) \tag{3.7}$$

where

$$\left.\begin{array}{c} u_{\mathfrak{p}_r} \in \mathbb{R}^{m_{\mathfrak{p}_r}},\ m_{\mathfrak{p}_r} \in \mathbb{N},\ \forall r \in \mathcal{R}, \\ \sum_{r=1}^{\mathrm{R}} m_{\mathfrak{p}_r} = m_{\mathfrak{p}}, \\ \mathcal{R} := \{1, \ldots, \mathrm{R}\}. \end{array}\right\} \tag{3.8}$$

3.1.2 Network

The considered network is packet based and is modeled as an *erasure channel*. More precisely, the transmission effects which are also regarded as exogenous disturbances to the network, are modeled as discrete dropout process $\{w_{\mathfrak{n}}(k)\}_{k \in \mathbb{N}_0}$ where

$$w_{\mathfrak{n}}(k) := \begin{cases} 0, & \text{if dropout occurs at time instant } k, \\ 1, & \text{if dropout } \textit{does not} \text{ occur at time instant } k. \end{cases} \tag{3.9}$$

We will denote the set of dropout outcomes with

$$\mathcal{D} := \{0, 1\}. \tag{3.10}$$

Note that for each $k \in \mathbb{N}_0$, $w_{\mathfrak{n}}(k) \in \mathcal{D}$. Furthermore, we will relabel successful transmission time instants by k_i, i.e., $w_{\mathfrak{n}}(k_i) = 1$, where $i \in \mathbb{N}_0$, and introduce the set consisting of all successful transmission time instants as

$$\mathbb{K} := \{k_i \in \mathbb{N}_0 : w_{\mathfrak{n}}(k_i) = 1, k_{i+1} > k_i,\ \forall i \in \mathbb{N}_0\}. \tag{3.11}$$

We will define the number of consecutive packet dropouts between successful transmission instants k_i and k_{i+1} as

$$\Delta k_i := k_{i+1} - k_i - 1. \tag{3.12}$$

At each time instant the network will receive a *packet* π sent by the controller. This packet will be a tuple consisting of *address* and *data* fields. The address denoted via $\mathfrak{a}$ takes values from the set $\mathcal{R}$ while the data denoted by $\mathfrak{d}$ takes values from the set $\mathbb{R}^{\mathrm{L}\cdot\bar{m}_{\mathfrak{p}}}$ where L is the length of the buffer and $\bar{m}_{\mathfrak{p}} = \max\{m_{\mathfrak{p}_1}, \ldots, m_{\mathfrak{p}_{\mathrm{R}}}\}$. More precisely, we have

$$\pi := (\mathfrak{a}, \mathfrak{d}) \in \mathcal{R} \times \mathbb{R}^{\mathrm{L}\cdot\bar{m}_{\mathfrak{p}}}. \tag{3.13}$$

The number of network outputs corresponds to the number of plant inputs. However, at each time instant, due to scheduling issue we consider (i.e., at each time instant only one plant input can be accessed), in the case of no packet dropout, only one of these outputs will be "active" with respect to the plant. More precisely, the one being addressed; of course, the other network outputs are used for transmission of data to other processes connected to the network. Hence, for each $k \in \mathbb{N}_0$ we have

$$\Sigma_{\mathfrak{n}} : \begin{cases} y_{\mathfrak{n}_r}(k) = w_{\mathfrak{n}}(k)\mathfrak{d}(k), & \text{if } r = \mathfrak{a}(k), \\ y_{\mathfrak{n}_r}(k) \text{ is "inactive" with respect to the plant,} & \text{if } r \neq \mathfrak{a}(k). \end{cases} \tag{3.14}$$

Note that we do not introduce a variable which would keep track if a network output is active or not with respect to the plant. Indeed, this would be a more accurate description but it would also, unfortunately, complicate the notation, resulting in a more cumbersome presentation.

3.1.3 Buffer

As depicted in Fig. 3.1 there are R buffers between the network and the plant; one buffer for each plant input. These buffers are devices consisting of a memory unit and a simple processing unit. The dynamics of each buffer is captured by the dynamics of a linear

switched system

$$\Sigma_{x_{\mathfrak{b}_r}} : \begin{cases} x^+_{\mathfrak{b}_r} := \begin{cases} S_{m_{\mathfrak{p}_r}} x_{\mathfrak{b}_r}, & \text{if } r \neq \mathfrak{a} \text{ or } w_{\mathfrak{n}} = 0, \\ S_{m_{\mathfrak{p}_r}} \mathfrak{d}, & \text{if } r = \mathfrak{a} \text{ and } w_{\mathfrak{n}} = 1, \end{cases} \\ y_{\mathfrak{b}_r} := \begin{cases} [I^{m_{\mathfrak{p}_r}} \; 0^{m_{\mathfrak{p}_r}} \cdots 0^{m_{\mathfrak{p}_r}}] x_{\mathfrak{b}_r}, & \text{if } r \neq \mathfrak{a} \text{ or } w_{\mathfrak{n}} = 0, \\ [I^{m_{\mathfrak{p}_r}} \; 0^{m_{\mathfrak{p}_r}} \cdots 0^{m_{\mathfrak{p}_r}}] \mathfrak{d}, & \text{if } r = \mathfrak{a} \text{ and } w_{\mathfrak{n}} = 1, \end{cases} \end{cases}, \tag{3.15}$$

for each $r \in \mathcal{R}$, where

$$S_{m_{\mathfrak{p}_r}} := \begin{bmatrix} 0_{m_{\mathfrak{p}_r}} & I_{m_{\mathfrak{p}_r}} & 0_{m_{\mathfrak{p}_r}} & \cdots & 0_{m_{\mathfrak{p}_r}} \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0_{m_{\mathfrak{p}_r}} & \cdots & 0_{m_{\mathfrak{p}_r}} & I_{m_{\mathfrak{p}_r}} & 0_{m_{\mathfrak{p}_r}} \\ 0_{m_{\mathfrak{p}_r}} & \cdots & \cdots & 0_{m_{\mathfrak{p}_r}} & I_{m_{\mathfrak{p}_r}} \\ 0_{m_{\mathfrak{p}_r}} & \cdots & \cdots & \cdots & 0_{m_{\mathfrak{p}_r}} \end{bmatrix} \in \mathbb{R}^{\mathrm{L} \cdot m_{\mathfrak{p}_r} \times \mathrm{L} \cdot m_{\mathfrak{p}_r}}, \tag{3.16}$$

is the *shift* matrix. As name suggests, this matrix when multiplying a corresponding vector, will shift the values of a vector for one spot and place a zero value at the last spot. Let us consider a very simple illustrative example.

Example 3.1 (Shift matrix)**.** *Consider the following shift matrix,*

$$S_3 := \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}$$

and a corresponding vector $\nu = (\nu_1, \nu_2, \nu_3) \in \mathbb{R}^3$*. Then, it follows*

$$S_3 \nu = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \nu_1 \\ \nu_2 \\ \nu_3 \end{bmatrix} = \begin{bmatrix} \nu_2 \\ \nu_3 \\ 0 \end{bmatrix}.$$

Note that with a small change to the shift matrix we can achieve that "last" value of a vector is repeated ad infinitum. Namely, let us consider the following matrix

$$\tilde{S}_3 := \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \end{bmatrix}.$$

Then, it follows

$$\tilde{S}_3 \nu = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \nu_1 \\ \nu_2 \\ \nu_3 \end{bmatrix} = \begin{bmatrix} \nu_2 \\ \nu_3 \\ \nu_3 \end{bmatrix},$$

leading to $\tilde{S}_3 \cdots \tilde{S}_3 \tilde{S}_3 \nu = (\nu_3, \nu_3, \nu_3)$. □

Now, note the interconnection

$$y_{\mathfrak{b}_r} = u_{\mathfrak{p}_r}, \tag{3.17}$$

for each $r \in \mathcal{R}$. To simplify notation and presentation we define the overall buffer as

$$x_{\mathfrak{b}} := (x_{\mathfrak{b}_1}, \ldots, x_{\mathfrak{b}_{\mathrm{R}}}) \in \mathbb{R}^{\mathrm{L} \cdot m_{\mathfrak{p}}}. \tag{3.18}$$

(Note that $n_{\mathfrak{b}} := \mathrm{L} \cdot m_{\mathfrak{p}}$.) Then, we capture its dynamics via

$$\Sigma_{\mathfrak{b}} : \begin{cases} x_{\mathfrak{b}}^{+} := f_{\mathfrak{b}}(x_{\mathfrak{b}}, \mathfrak{a}, \mathfrak{d}, w_{\mathfrak{n}}) := \Xi(\mathfrak{a}, w_{\mathfrak{n}}) S x_{\mathfrak{b}} + S \Pi(\mathfrak{d}, \mathfrak{a}, w_{\mathfrak{n}}), \\ y_{\mathfrak{b}} := h_{\mathfrak{b}}(x_{\mathfrak{b}}, \mathfrak{a}, \mathfrak{d}, w_{\mathfrak{n}}) := \Gamma\Big(\Xi(\mathfrak{a}, w_{\mathfrak{n}}) x_{\mathfrak{b}} + \Pi(\mathfrak{d}, \mathfrak{a}, w_{\mathfrak{n}})\Big), \end{cases} \tag{3.19}$$

where

$$\Xi(\mathfrak{a}, w_{\mathfrak{n}}) := \begin{cases} \mathrm{diag}(I_{\mathrm{L} \cdot m_{\mathfrak{p}_1}}, \ldots, I_{\mathrm{L} \cdot m_{\mathfrak{p}_{\mathfrak{a}-1}}}, 0_{\mathrm{L} \cdot m_{\mathfrak{p}_{\mathfrak{a}}}}, I_{\mathrm{L} \cdot m_{\mathfrak{p}_{\mathfrak{a}+1}}}, \ldots, I_{\mathrm{L} \cdot m_{\mathfrak{p}_{\mathrm{R}}}}), & \text{if } w_{\mathfrak{n}} = 1, \\ I_{\mathrm{L} \cdot m_{\mathfrak{p}}}, & \text{if } w_{\mathfrak{n}} = 0, \end{cases} \tag{3.20}$$

is a mapping that selects the addressed element of the corresponding overall buffer;

$$S := \mathrm{diag}(S_{m_{\mathfrak{p}_1}}, \ldots, S_{m_{\mathfrak{p}_{\mathrm{R}}}}), \tag{3.21}$$

is the overall shift matrix;

$$\Pi(\mathfrak{d}, \mathfrak{a}, w_{\mathfrak{n}}) := \begin{cases} \left(0^{\mathrm{L}\cdot m_{\mathfrak{p}_1}}, \ldots, 0^{\mathrm{L}\cdot m_{\mathfrak{p}_{\mathfrak{a}-1}}}, \mathfrak{d}, 0^{\mathrm{L}\cdot m_{\mathfrak{p}_{\mathfrak{a}+1}}}, \ldots, 0^{\mathrm{L}\cdot m_{\mathfrak{p}_{\mathrm{R}}}}\right), & \text{if } w_{\mathfrak{n}} = 1, \\ 0^{\mathrm{L}\cdot m_{\mathfrak{p}}}, & \text{if } w_{\mathfrak{n}} = 0, \end{cases} \tag{3.22}$$

is a mapping that updates the buffer state in case of a successful transmission;

$$\left.\begin{aligned} &\Gamma := [\varsigma_1 \ \cdots \ \varsigma_{\mathrm{R}}], \\ &\varsigma_r := \begin{bmatrix} 0_{m_{\mathfrak{p}_1}} & 0_{m_{\mathfrak{p}_1}} & \cdots & 0_{m_{\mathfrak{p}_1}} \\ \vdots & \vdots & \cdots & \vdots \\ 0_{m_{\mathfrak{p}_{r-1}}} & 0_{m_{\mathfrak{p}_{r-1}}} & \cdots & 0_{m_{\mathfrak{p}_{r-1}}} \\ I_{m_{\mathfrak{p}_r}} & 0_{m_{\mathfrak{p}_r}} & \cdots & 0_{m_{\mathfrak{p}_r}} \\ 0_{m_{\mathfrak{p}_{r+1}}} & 0_{m_{\mathfrak{p}_{r+1}}} & \cdots & 0_{m_{\mathfrak{p}_{r+1}}} \\ \vdots & \vdots & \cdots & \vdots \\ 0_{m_{\mathfrak{p}_{\mathrm{R}}}} & 0_{m_{\mathfrak{p}_{\mathrm{R}}}} & \cdots & 0_{m_{\mathfrak{p}_{\mathrm{R}}}} \end{bmatrix} \in \mathbb{R}^{m_{\mathfrak{p}} \times \mathrm{L}\cdot m_{\mathfrak{p}_r}}, \ \forall r \in \mathcal{R}, \end{aligned}\right\} \tag{3.23}$$

is a matrix which selects only the first element of each individual buffer. Let us illustrate the former notation on a simple example.

Example 3.2 (Buffer dynamics)**.** *Let $x_{\mathfrak{b}} \in \mathbb{R}^{3\cdot 3}$ and let $(\mathfrak{a}, \mathfrak{d}) = (1, (\mathfrak{d}_1, \mathfrak{d}_2, \mathfrak{d}_3))$. There are two possibilities, namely, a packet dropout ($w_{\mathfrak{n}} = 0$) and a successful transmission ($w_{\mathfrak{n}} = 1$). Let us consider these two cases.*

Packet dropout:

$$
\begin{aligned}
x_{\mathfrak{b}}^{+} = &\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_{\mathfrak{b}_{11}} \\ x_{\mathfrak{b}_{12}} \\ x_{\mathfrak{b}_{13}} \\ x_{\mathfrak{b}_{21}} \\ x_{\mathfrak{b}_{22}} \\ x_{\mathfrak{b}_{23}} \\ x_{\mathfrak{b}_{31}} \\ x_{\mathfrak{b}_{32}} \\ x_{\mathfrak{b}_{33}} \end{bmatrix} \\
&+ \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} x_{\mathfrak{b}_{12}} \\ x_{\mathfrak{b}_{13}} \\ 0 \\ x_{\mathfrak{b}_{22}} \\ x_{\mathfrak{b}_{23}} \\ 0 \\ x_{\mathfrak{b}_{32}} \\ x_{\mathfrak{b}_{33}} \\ 0 \end{bmatrix},
\end{aligned}
$$

$$
y_{\mathfrak{b}} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_{\mathfrak{b}_{11}} \\ x_{\mathfrak{b}_{12}} \\ x_{\mathfrak{b}_{13}} \\ x_{\mathfrak{b}_{21}} \\ x_{\mathfrak{b}_{22}} \\ x_{\mathfrak{b}_{23}} \\ x_{\mathfrak{b}_{31}} \\ x_{\mathfrak{b}_{32}} \\ x_{\mathfrak{b}_{33}} \end{bmatrix} = \begin{bmatrix} x_{\mathfrak{b}_{11}} \\ x_{\mathfrak{b}_{21}} \\ x_{\mathfrak{b}_{31}} \end{bmatrix}.
$$

Successful transmission:

$$
\begin{aligned}
x_{\mathfrak{b}}^{+} = &\begin{bmatrix} 0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0\\ 0&0&0&1&0&0&0&0&0\\ 0&0&0&0&1&0&0&0&0\\ 0&0&0&0&0&1&0&0&0\\ 0&0&0&0&0&0&1&0&0\\ 0&0&0&0&0&0&0&1&0\\ 0&0&0&0&0&0&0&0&1 \end{bmatrix} \begin{bmatrix} 0&1&0&0&0&0&0&0&0\\ 0&0&1&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0\\ 0&0&0&0&1&0&0&0&0\\ 0&0&0&0&0&1&0&0&0\\ 0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&1&0\\ 0&0&0&0&0&0&0&0&1\\ 0&0&0&0&0&0&0&0&0 \end{bmatrix} \begin{bmatrix} x_{\mathfrak{b}_{11}}\\ x_{\mathfrak{b}_{12}}\\ x_{\mathfrak{b}_{13}}\\ x_{\mathfrak{b}_{21}}\\ x_{\mathfrak{b}_{22}}\\ x_{\mathfrak{b}_{23}}\\ x_{\mathfrak{b}_{31}}\\ x_{\mathfrak{b}_{32}}\\ x_{\mathfrak{b}_{33}} \end{bmatrix} \\
&+ \begin{bmatrix} 0&1&0&0&0&0&0&0&0\\ 0&0&1&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0\\ 0&0&0&0&1&0&0&0&0\\ 0&0&0&0&0&1&0&0&0\\ 0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&1&0\\ 0&0&0&0&0&0&0&0&1\\ 0&0&0&0&0&0&0&0&0 \end{bmatrix} \begin{bmatrix} \mathfrak{d}_1\\ \mathfrak{d}_2\\ \mathfrak{d}_3\\ 0\\ 0\\ 0\\ 0\\ 0\\ 0 \end{bmatrix} = \begin{bmatrix} \mathfrak{d}_2\\ \mathfrak{d}_3\\ 0\\ x_{\mathfrak{b}_{22}}\\ x_{\mathfrak{b}_{23}}\\ 0\\ x_{\mathfrak{b}_{32}}\\ x_{\mathfrak{b}_{33}}\\ 0 \end{bmatrix},
\end{aligned}
$$

$$
y_{\mathfrak{b}} = \begin{bmatrix} 1&0&0&0&0&0&0&0&0\\ 0&0&0&1&0&0&0&0&0\\ 0&0&0&0&0&0&1&0&0 \end{bmatrix} \begin{bmatrix} \mathfrak{d}_1\\ \mathfrak{d}_2\\ \mathfrak{d}_3\\ x_{\mathfrak{b}_{21}}\\ x_{\mathfrak{b}_{22}}\\ x_{\mathfrak{b}_{23}}\\ x_{\mathfrak{b}_{31}}\\ x_{\mathfrak{b}_{32}}\\ x_{\mathfrak{b}_{32}} \end{bmatrix} = \begin{bmatrix} \mathfrak{d}_1\\ x_{\mathfrak{b}_{21}}\\ x_{\mathfrak{b}_{31}} \end{bmatrix}.
$$

□

Finally, note the interconnection

$$
y_{\mathfrak{b}} = u_{\mathfrak{p}}. \tag{3.24}
$$

3.1.4 Controller

The controller is an MPC controller. Recall that we want to mitigate the effects of packet dropouts and scheduling. As discussed in the introduction of this thesis and this chapter, these issues will be addressed by explicitly incorporating their effects into the prediction model(s). For the ease of presentation let us assume that all inputs have the same dimension which enables us to consider a node $r \in \mathcal{R}$ instead all of them independently; even in case that inputs have different dimensions the corresponding changes would be minimal.The corresponding nominal prediction model used for generating predictions is given as

$$\Sigma_{\mathfrak{p}}^{\mathfrak{m}} : \begin{cases} \tilde{x}_{\mathfrak{p}}(k+i+1) := f_{\mathfrak{p}}(\tilde{x}_{\mathfrak{p}}(k+i), \tilde{u}_{\mathfrak{p}}^{r}(k+i)), \\ \tilde{x}_{\mathfrak{p}}(k) = x_{\mathfrak{p}}(k),\ k \in \mathbb{N}_0,\ i \in \{0, \ldots, \mathfrak{h}-1\} \end{cases} \tag{3.25}$$

where $\mathfrak{h} \in \mathbb{N}$ is a *finite horizon*; in (3.25) we do not use a succinct notation since we want to point out to a distinction between a "real" time (k) and a "prediction" (i) time. Note that at each time instant k, a model (3.25) is initialized by the corresponding measurement of the plant state, i.e., $\tilde{x}_{\mathfrak{p}}(k) = x_{\mathfrak{p}}(k)$. Moreover, note that for each of these optimization problems we can only minimize over the control values of the corresponding node while control values of other nodes are fixed. This is denoted by $\tilde{u}_{\mathfrak{p}}^{r}$ which is defined as

$$\tilde{u}_{\mathfrak{p}}^{r} := \begin{bmatrix} \Gamma_{m_1}\Big(\Xi(r, \tilde{w}_{\mathfrak{n}})\tilde{x}_{\mathfrak{b}} + \Pi(\{\tilde{u}_{\mathfrak{p}_r}\}_{k}^{k+\mathfrak{h}-1}, r, \tilde{w}_{\mathfrak{n}})\Big) \\ \vdots \\ \Gamma_{m_{r-1}}\Big(\Xi(r, \tilde{w}_{\mathfrak{n}})\tilde{x}_{\mathfrak{b}} + \Pi(\{\tilde{u}_{\mathfrak{p}_r}\}_{k}^{k+\mathfrak{h}-1}, r, \tilde{w}_{\mathfrak{n}})\Big) \\ \tilde{u}_{\mathfrak{p}_r} \\ \Gamma_{m_{r+1}}\Big(\Xi(r, \tilde{w}_{\mathfrak{n}})\tilde{x}_{\mathfrak{b}} + \Pi(\{\tilde{u}_{\mathfrak{p}_r}\}_{k}^{k+\mathfrak{h}-1}, r, \tilde{w}_{\mathfrak{n}})\Big) \\ \vdots \\ \Gamma_{m_{\mathrm{R}}}\Big(\Xi(r, \tilde{w}_{\mathfrak{n}})\tilde{x}_{\mathfrak{b}} + \Pi(\{\tilde{u}_{\mathfrak{p}_r}\}_{k}^{k+\mathfrak{h}-1}, r, \tilde{w}_{\mathfrak{n}})\Big) \end{bmatrix} \in \mathbb{R}^{m_{\mathfrak{p}}}, \tag{3.26}$$

where

$$\Sigma_{\mathfrak{b}}^{\mathfrak{m}} : \tilde{x}_{\mathfrak{b}}^{+} := f_{\mathfrak{b}}(\tilde{x}_{\mathfrak{b}}, r, \{\tilde{u}_{\mathfrak{p}_r}\}_{k}^{k+\mathfrak{h}-1}, \tilde{w}_{\mathfrak{n}}),\ \tilde{x}_{\mathfrak{b}} = x_{\mathfrak{b}}, \tag{3.27}$$

is an overall buffer *model* initialized with $x_{\mathfrak{b}}$;

$$\tilde{u}_{\mathfrak{p}_r} \in \{\tilde{u}_{\mathfrak{p}_r}\}_k^{k+\mathfrak{h}-1}, \tag{3.28}$$

is a current predicted control value for node r that belongs to a sequence of predicted control values for node r over a finite horizon $\mathfrak{h}$;

$$\tilde{w}_{\mathfrak{n}} \in \{\tilde{w}_{\mathfrak{n}}\}_k^{k+\mathfrak{h}-1} := \{1, 0, \ldots, 0\}, \tag{3.29}$$

is a current predicted dropout outcome that belongs to a sequence of *chosen* dropout outcomes over a finite horizon $\mathfrak{h}$ which captures the scenario of having a successful transmission and then $\mathfrak{h} - 1$ consecutive dropouts;

$$\Gamma = \left[\Gamma_{m_{\mathfrak{p}_1}}^\top \cdots \Gamma_{m_{\mathfrak{p}_R}}^\top\right]^\top, \ \Gamma_{m_{\mathfrak{p}_r}} \in \mathbb{R}^{m_{\mathfrak{p}_r} \times \mathfrak{h} \cdot m_{\mathfrak{p}}}, \forall r \in \mathcal{R}, \tag{3.30}$$

where $\Gamma_{m_{\mathfrak{p}_\cdot}}$ are rows of the matrix Γ.

Remark: As stated above, the chosen sequence $\{\tilde{w}_{\mathfrak{n}}\}_k^{k+\mathfrak{h}-1} := \{1, 0, \ldots, 0\}$ captures the scenario where after a successful transmission, $\mathfrak{h} - 1$ consecutive dropouts occur. This captures one of the scenarios where over a finite horizon $\mathfrak{h}$ there will be only one successful transmission. Having no successful transmissions over entire horizon $\mathfrak{h}$ would be the worst case possible. The next worst case would be to have only one. Indeed, one would need some kind of metric to compare all possible $\mathfrak{h}$ scenarios. However, from control point of view it makes sense to consider scenario where a successful transmission is followed with $\mathfrak{h} - 1$ consecutive packet dropouts. Namely, in this case controller computes $\mathfrak{h}$ new control values. Otherwise, up until successful transmission, old values would be used and less new values would be computed, e.g., (3.27) would be shorter. Correspondingly, buffers would ran out of data much quicker. □

Note that the control predictions depend on the content of the overall buffer. Hence, we assume that this content is known to the controller. This can be done by sending buffer values along with plant measurements or perhaps via acknowledgments of receipt; for the ease of presentation we denote any of these cases in Fig. 3.1 with ACK. However, if such acknowledgments are not available then one could adopt a stochastic control frame-

work as in [163], or use a control which accounts for all transmission scenarios as in [164].

It will be convenient for presentation purposes to introduce the "NCS" state as augmentation of plant and buffer state, that is

$$x := (x_{\mathfrak{p}}, x_{\mathfrak{b}}) \in \mathbb{R}^{n_{\mathfrak{p}} \times \mathfrak{h} \cdot m_{\mathfrak{p}}}. \tag{3.31}$$

The cost function is then defined as

$$J(x, \{\tilde{u}_{\mathfrak{p}_r}\}_k^{k+\mathfrak{h}-1}) := \sum_{i=k}^{k+\mathfrak{h}-1} l(\phi_{f_{\mathfrak{p}}}(i-k, x_{\mathfrak{p}}, \{\tilde{u}_{\mathfrak{p}}^r\}_k^{k+\mathfrak{h}-1}), \tilde{u}_{\mathfrak{p}}^r(i)) + g(\phi_{f_{\mathfrak{p}}}(\mathfrak{h}, x_{\mathfrak{p}}, \{\tilde{u}_{\mathfrak{p}}^r\}_k^{k+\mathfrak{h}-1})) \tag{3.32}$$

where $l : \mathbb{R}^{n_{\mathfrak{p}}} \times \mathbb{R}^{m_{\mathfrak{p}}} \to \mathbb{R}_{\geq 0}$ is a stage cost function and $g : \mathbb{R}^{n_{\mathfrak{p}}} \to \mathbb{R}_{\geq 0}$ is a terminal cost function. Recall again that we have R nodes, hence, we will have R cost functions. Correspondingly, from each we will obtain the corresponding sequence of optimally predicted control values for the corresponding node $\{u_{\mathfrak{p}_r}^*\}_k^{k+\mathfrak{h}-1}$, namely

$$V(x, \{u_{\mathfrak{p}_r}^*\}_k^{k+\mathfrak{h}-1}) := \min_{\{\tilde{u}_{\mathfrak{p}_r}\}_k^{k+\mathfrak{h}-1}} J(x, \{\tilde{u}_{\mathfrak{p}_r}\}_k^{k+\mathfrak{h}-1}). \tag{3.33}$$

Then, we will extract the sequence of optimally predicted control values that produces the minimal cost value, i.e.,

$$V(x) = V(x, \{u_{\mathfrak{p}_{r^*}}^*\}_k^{k+\mathfrak{h}-1}) := \min_r V(x, \{u_{\mathfrak{p}_r}^*\}_k^{k+\mathfrak{h}-1}) \tag{3.34}$$

Then, extracting the values for the optimal node and its corresponding sequence of optimal control values reduces to

$$r^* := \operatorname*{argmin}_r V(x, \{u_{\mathfrak{p}_r}^*\}_k^{k+\mathfrak{h}-1}), \tag{3.35}$$

and

$$\{u_{\mathfrak{p}_{r^*}}^*\}_k^{k+\mathfrak{h}-1} := \operatorname*{argmin}_{\{\tilde{u}_{\mathfrak{p}_{r^*}}\}_k^{k+\mathfrak{h}-1}} J(x, \{\tilde{u}_{\mathfrak{p}_{r^*}}\}_k^{k+\mathfrak{h}-1}). \tag{3.36}$$

The packet that controller sends to a network is then formed as

$$\pi = (\mathfrak{a}, \mathfrak{d}) := (r^*, (u^*_{\mathfrak{p}_{r^*}}(k_i), \ldots, u^*_{\mathfrak{p}_{r^*}}(k_i + \mathfrak{h} - 1)). \tag{3.37}$$

Recall that from an optimization viewpoint, at every discrete-time instant the controller needs to solve R deterministic finite horizon optimization problems. A key feature here is that these R optimization problems can be carried out in parallel. Thus, the computational burden scales linearly with the number of plant inputs (nodes). If implemented on hardware with parallel processors, computation times will be comparable to those of regular MPC with horizon $\mathfrak{h}$.

Note also, that our method is designed so that the controller addresses the buffer most in need. Therefore, it may occur that some buffers will run out of data more often than others. However, if appropriate assumptions are satisfied, then the proposed method will guarantee stability in the presence of dropouts and scheduling, and even exogenous disturbances on the plant (considered in the next chapter).

Before we state the closed-loop system we add a remark on the reasons why the network is absent in feedback connection.

Remark: Recall that throughout this work we assumed that the controller has direct access to measurements of plant state. This assumption is made since accurate plant measurements are crucial in making predictions in MPC framework; e.g., see [107] for in-depth explanations. Furthermore, due to lack of network imperfections in sensor-controller connection the algorithms for acknowledgments of receipt will be simpler and reliable. For instance, see [21, 94, 165] for complications that arise when a network is included in sensor-controller connection. However, if one would take into account sensor-controller connection imperfections within the scheme proposed one could use state observers with intermittent observations, e.g., see [40, 166–168]. (For robustness, which is considered in the next chapter, one would need state observation errors to be bounded, which is reasonable to expect due to bounded dropouts.) Alternatively, one could also restrict the controller to only calculate control sequences at instances of successful sensor-data receptions. Notice that works of [21, 94] provide high-level insight and results related to stabilization of NCSs which additionally include network in sensor-controller

communication, scheduling of plant measurements, model uncertainties, time-varying delays, irregularity of transfer intervals and/or design of the corresponding protocol. Extending our results to such NCSs would introduce extra complexity which is outside the scope of the thesis and will be considered in the future.

Finally, the closed-loop system is given as

$$\Sigma_x : x^+ = \begin{bmatrix} f_{\mathfrak{p}}(x_{\mathfrak{p}}, h_{\mathfrak{b}}(x_{\mathfrak{b}}, r^*, \{u^*_{r^*}\}_k^{k+\mathfrak{h}-1}, w_{\mathfrak{n}})) \\ f_{\mathfrak{b}}(x_{\mathfrak{b}}, r^*, \{u^*_{r^*}\}_k^{k+\mathfrak{h}-1}, w_{\mathfrak{n}}) \end{bmatrix} =: f(x, w_{\mathfrak{n}}). \tag{3.38}$$

3.2 Stability analysis

We concentrate on establishing Uniform Global Asymptotic Stability (UGAS) of the system Σ_x, see (3.38). First, we state our assumptions. These are also necessary for further presentation since some notation depends on it.

3.2.1 Assumptions

As discussed in the introduction of the thesis, there are many causes for packet dropouts, some of them being excessive (infinite) delays, packet collisions, traffic congestion and/or failed transmissions. Moreover, they are inevitable, see [17–20], and, they occur *randomly*. Hence, imposing any deterministic finite bound on the number of *consecutive* packet dropouts seems "unrealistic". However, the networks are designed to have a high throughput, thus, from a practical point of view (and experience), dropouts occur with low probability. Nonetheless, even rare packet dropouts can cause instability in the corresponding NCS, e.g., for unstable plants. Hence, in these cases it makes sense to assume a deterministic finite bound on the number of *consecutive* packet dropouts which is done next.

Assumption 3.1 (Bound on the number of consecutive packet dropouts)**.** *There exists*[1] $\Delta k \in \mathbb{N}$ *such that* $\Delta k \leq \mathrm{L}$ *and* $\Delta k_i \leq \Delta k - 1$ *for each* $k_i \in \mathbb{N}_0$. □

[1]Recall that $\mathrm{L} \in \mathbb{N}$ is the length of a buffer.

As indicated above, the assumptions made will impact our notation to some extent. The first effect is the definition of a set consisting of *successful transmission instants* such that the number of consecutive packet dropouts is at most $\Delta k - 1$. More precisely,

$$\mathbb{K}_{\Delta k} := \{k_i \in \mathbb{K} : \Delta k_i \leq \Delta k - 1, k_0 \leq \Delta k\}. \tag{3.39}$$

Further, we define a set consisting of all infinite sequences of dropout outcomes satisfying Assumption 3.1 now as

$$\mathcal{S}^{\mathcal{D}}_{\mathbb{K}_{\Delta k}} := \{\{w_{\mathfrak{n}}\}_0^{\infty} \in \mathcal{S}^{\mathcal{D}} : k_i \in \mathbb{K}_{\Delta k}\}. \tag{3.40}$$

Furthermore, note the following equality

$$\{w_{\mathfrak{n}}\}_{k_i}^{k_{i+1}-1} = \{w_{\mathfrak{n}}\}_{k_i}^{k_i+\Delta k} = \{w_{\mathfrak{n}}(k_i), w_{\mathfrak{n}}(k_i+1), \ldots, w_{\mathfrak{n}}(k_{i+1}-1)\} = \{1, 0, \ldots, 0\} \tag{3.41}$$

which enables us to partition each $\{w_{\mathfrak{n}}\}_0^k \subset \{w_{\mathfrak{n}}\}_0^{\infty} \in \mathcal{S}^{\mathcal{D}}_{\mathbb{K}_{\Delta k}}$ as

$$\{w_{\mathfrak{n}}\}_0^k = \{\{w_{\mathfrak{n}}\}_0^{k_0-1}, \{w_{\mathfrak{n}}\}_{k_0}^{k_1-1}, \ldots, \{w_{\mathfrak{n}}\}_{k_i}^{k_{i+1}-1}, \{w_{\mathfrak{n}}\}_{k_{i+1}}^{k_{i+2}-1}, \ldots, \{w_{\mathfrak{n}}\}_{k_{j-1}}^{k_j-1}, \{w_{\mathfrak{n}}\}_{k_j}^{k}\}. \tag{3.42}$$

Next, let us proceed with the MPC related assumptions, but before we begin, we state a remark regarding the *horizon* used in MPC.

Remark: The prediction horizon in MPC is assumed to be greater or equal than the *bound* on the number of consecutive packet dropouts Δk, however, for the presentation purposes we will set it to be equal, that is

$$\mathfrak{h} := \Delta k, \ \forall r \in \mathcal{R}. \tag{3.43}$$

Indeed, as stated at the beginning of this section, dropouts are random and, thus, we can not really deterministically measure or ensure the bound on the number of consecutive packet dropouts. However, due to the design of networks so that they have high throughput, practice and experience indicate that the probability of dropouts is relatively

low, [17–20]. Thus, imposing a sufficiently large deterministic finite bound on the number of consecutive packet dropouts makes sense. Practically, one would typically set the prediction horizon to be sufficiently long, and then would claim the stability for a situation where the actual bound on the number of consecutive packet dropouts is smaller than this horizon. Indeed, increasing the horizon does require larger buffers and more computation but that is something we have control over. □

Remark: Furthermore, notice that the model of the plant has to be "accurate enough". For instance, if the model is not accurate, then the longer the horizon, the worse our predictions are and this would affect the quality of predicted control values at the end of the horizon. Analyzing robustness with respect to model uncertainty is particularly important and it is left for further research. □

Now, the first MPC assumption is concerned with a stage and a terminal cost function, see (3.32).

Assumption 3.2 (Semi-positive definiteness and lower bounds on stage cost function)**.** *There exist class-$\mathcal{K}_\infty$ functions $\alpha_{x_\mathfrak{p}}$ and $\alpha_{u_\mathfrak{p}}$ such that*

$$\left.\begin{array}{c} l(x_\mathfrak{p}, u_\mathfrak{p}) \geq \alpha_{x_\mathfrak{p}}(|x_\mathfrak{p}|),\ l(x_\mathfrak{p}, u_\mathfrak{p}) \geq \alpha_{u_\mathfrak{p}}(|u_\mathfrak{p}|),\ l(0^{n_\mathfrak{p}}, 0^{m_\mathfrak{p}}) = 0, \\ g(x_\mathfrak{p}) \geq 0,\ g(0^{n_\mathfrak{p}}) = 0, \end{array}\right\} \tag{3.44}$$

for each $x_\mathfrak{p} \in \mathbb{R}^{n_\mathfrak{p}}$ and each $u_\mathfrak{p} \in \mathbb{R}^{m_\mathfrak{p}}$. □

The stated assumption, without a lower bound on stage cost function in terms of control, is a "standard" one, see for instance [104–121]. The additional part, i.e., a lower bound on stage cost function in terms of control *only*, is the *enabling* piece in establishing UGAS; this additional bound on the stage cost function was used for instance in [160].

The following assumption is a modified "standard" stability-related assumption, for instance see [107]. In particular this assumption is used to construct a sequence of feasible control values.

Assumption 3.3 (Terminal control law)**.** *For some node $r \in \mathcal{R}$ there exists a* terminal control law $\kappa_r : \mathbb{R}^{n_\mathfrak{p} \times \mathfrak{h} \cdot m_\mathfrak{p}} \to \bar{\mathbb{U}}_{\mathfrak{p}_r}$ *such that*

$$\left.\begin{aligned} g(f_{\mathfrak{p}}(x_{\mathfrak{p}}, \kappa_r(x_{\mathfrak{p}}, x_{\mathfrak{b}}))) - g(x_{\mathfrak{p}}) + l(x_{\mathfrak{p}}, \kappa_r(x_{\mathfrak{p}}, x_{\mathfrak{b}})) \leq 0, \\ f_{\mathfrak{p}}(x_{\mathfrak{p}}, \kappa_r(x_{\mathfrak{p}}, x_{\mathfrak{b}})) \in \mathbb{R}^{n_{\mathfrak{p}}}, \\ \kappa_r(x_{\mathfrak{p}}, x_{\mathfrak{b}}) \in \bar{\mathbb{U}}_{\mathfrak{p}_r} := 0^{m_{\mathfrak{p}_1}} \times \cdots \times 0^{m_{\mathfrak{p}_{r-1}}} \times \mathbb{R}^{m_{\mathfrak{p}_r}} \times 0^{m_{\mathfrak{p}_{r+1}}} \times \cdots \times 0^{m_{\mathfrak{p}_R}}, \end{aligned}\right\} \tag{3.45}$$

holds for each $(x_{\mathfrak{p}}, x_{\mathfrak{b}}) \in \mathbb{R}^{n_{\mathfrak{p}} \times \mathfrak{h} \cdot m_{\mathfrak{p}}}$. □

Notice that the control law κ_r is not necessarily used on the plant. It is just a construct adopted to establish stability results; e.g., see [107] for non-networked case. Establishing easy-to-check conditions so that the latter assumption is satisfied would depend on the plant dynamics and the chosen cost function. Locally, linearization could be a right approach to establish easy-to-check conditions. Also, notice that $\bar{\mathbb{U}}_{\mathfrak{p}_r}$ captures the scheduling constraints. The final assumption is related to the optimal value function. Namely, we assume that it is possible to bound it by a class-$\mathcal{K}$ function.

Assumption 3.4 (Class-$\mathcal{K}$ bound on the optimal value function)**.** *There exist a class-$\mathcal{K}$ function γ_V such that*

$$V(x) \leq \gamma_V(|x|) \tag{3.46}$$

holds for each $x = (x_{\mathfrak{p}}, x_{\mathfrak{b}}) \in \mathbb{R}^{n_{\mathfrak{p}} \times \mathfrak{h} \cdot m_{\mathfrak{p}}}$. □

For instance, this bound can be ensured by assuming asymptotic controllability as in Section III of [123].

We are now ready to proceed with the exposition of the main results.

3.2.2 Results

To establish UGAS of the system Σ_x (see (3.38)), we first establish that the corresponding system is *UGAS at successful transmission instants from set* $\mathbb{K}_{\Delta k}$ (defined in the sequel). Then, we address the "inter-sample" behavior which establishes the result. What we mean by inter-sample behaviour is illustrated in Fig. 3.3.

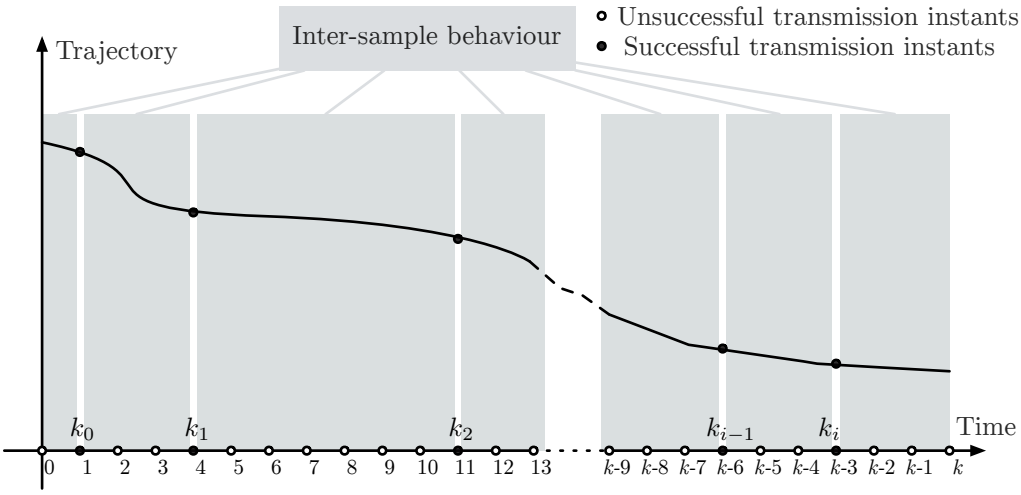


Figure 3.3: Illustration of what we refer to as inter-sample behavior; k denotes current discrete time.

This idea originates from [2, Theorem 2]. Namely, the corresponding result states that a sampled-data system is UGAS, if and only if, the corresponding discrete-time system is UGAS and uniformly bounded over a period T (UGBT); which needs to be longer than the inter-sample interval lengths. We will omit the corresponding details and steer the focus on how this result is modified so as to establish UGAS of the system Σ_x (see (3.38)); for those interested in the technical details we recommend to consult [2]. First, we define *UGAS at successful transmission instants from the set* $\mathbb{K}_{\Delta k}$ (UGAS-$\mathbb{K}_{\Delta k}$) of the system Σ_x (see (3.38)).

Definition 3.1 (UGAS-$\mathbb{K}_{\Delta k}$)**.** *Consider the system Σ_x (see (3.38)). We say that Σ_x is UGAS at successful transmission instances from the set $\mathbb{K}_{\Delta k}$ if there exist a class-$\mathcal{KL}$ function β such that for any initial condition $x(k_0) = x$, any k_0 and any $\{w_{\mathfrak{n}}\}_{k_0}^{k_i-1} \subset \{w_{\mathfrak{n}}\}_0^{\infty} \in \mathcal{S}^{\mathcal{D}}_{\mathbb{K}_{\Delta k}}$ we have*

$$|\phi_f(k_i - k_0, x, \{w_{\mathfrak{n}}\}_{k_0}^{k_i-1})| \leq \beta(|x|, k_i - k_0), \tag{3.47}$$

for all $k_i \in \mathbb{K}_{\Delta k}$. □

Next, we define *Uniform Global Boundedness over* Δk (UGB-Δk) of the system Σ_x (see (3.38)), which can be seen as a counterpart to a discrete-time system being UGBT in [2]. Notice that we rely on the reasoning provided in [2]. Namely, sufficient conditions for

UGB-Δk can be obtained by following the steps provided in Lemma 3 and Lemma 4 in [2].

Definition 3.2 (UGB-Δk). *Consider the system Σ_x (see (3.38)). We say the solutions of the system Σ_x are Uniformly Globally Bounded over Δk if there exist a class-$\mathcal{K}_\infty$ function $\gamma_{\Delta k}$ such that*

$$|\phi_f(k-\bar{k}_0, x, \{w_{\mathfrak{n}}\}_{\bar{k}_0}^{k-1})| \leq \gamma_{\Delta k}(|x|) \tag{3.48}$$

for all $\bar{k}_0 \in \{0, \ldots, k_i + \Delta k_i\}$, $k_i \in \mathbb{K}_{\Delta k}$ and $\{w_{\mathfrak{n}}\}_{\bar{k}_0}^{k-1} \subset \{w_{\mathfrak{n}}\}_0^\infty \in \mathcal{S}_{\mathbb{K}_{\Delta k}}^{\mathcal{D}}$ where $k \in \{\bar{k}_0, \ldots, \bar{k}_0 + \Delta k\}$. □

The last two definitions enable us to state the main technical result which we use to establish the UGAS of the system Σ_x (see (3.38)).

Lemma 3.1. *The system Σ_x (see (3.38)) is UGAS, if the following conditions hold:*

1. *The system Σ_x is UGAS-$\mathbb{K}_{\Delta k}$,*
2. *The solutions of the system Σ_x are Uniformly Globally Bounded over Δk (UGB-Δk).*

More precisely, if the last two conditions are satisfied, then there exists a class-$\mathcal{KL}$ function β such that for each $x(\bar{k}_0) = x$, each $\bar{k}_0$ and each $\{w_{\mathfrak{n}}\}_{\bar{k}_0}^{k-1} \subset \{w_{\mathfrak{n}}\}_0^\infty \in \mathcal{S}_{\mathbb{K}_{\Delta k}}^{\mathcal{D}}$ the following holds

$$|\phi_f(k-\bar{k}_0, x, \{w_{\mathfrak{n}}\}_{\bar{k}_0}^{k-1})| \leq \beta(|x|, k-\bar{k}_0) \tag{3.49}$$

for each $k \geq \bar{k}_0 \geq 0$. □

Establishing UGAS-$\mathbb{K}_{\Delta k}$ is essential. In what follows we will establish this property via two approaches because of their different insights. We will first present an approach which uses the results on stability of cascaded systems. This approach reveals how one can exploit the fact that the overall buffer as a system is ISS with respect to the input being the arriving control sequence. Moreover, the application and modification of techniques used in the proofs are useful since they provide an alternative approach for addressing packet dropouts and scheduling.

This approach will be followed by a "standard" approach in which one constructs an appropriate Lyapunov function. The merit of this approach is the corresponding construction.

Cascade-based approach

In order to establish UGAS-$\mathbb{K}_{\Delta k}$, we use and/or modify several results. First, we exploit the result that establishes the equivalence between "$\mathcal{KL}$ stability with respect to two measures" on one hand, and "uniform stability and global boundedness" and "uniform global attractivity" on the other hand; for more details, please see [161] for the continuous-time case and [162] for the discrete-time case. The mentioned result will be modified and used for plant state and control trajectories. Then, by using the fact that the overall buffer is ISS with respect to the input being the arriving control sequence, we establish an appropriate bound on the buffer trajectories. Moreover, the fact that the overall buffer is UGAS uniformly in buffer states (i.e., ISS with zero gain), allows us to regard the corresponding system as a cascade although in principal it is a feedback connection. This further enables us to use stability proofs for cascade systems, such as the one provided in [169], to establish UGAS-$\mathbb{K}_{\Delta k}$ of the system Σ_x (see (3.38)).

Let us begin with the modification of the definition of $\mathcal{KL}$ stability with respect to two measures from [162].

Definition 3.3 (Modified Definition 2.1 from [162]). *Let $\rho_i : \mathbb{R}^{n_\mathfrak{p}} \times \mathbb{R}^{\mathfrak{h}\cdot m_\mathfrak{p}} \to \mathbb{R}_{\geq 0}$, $i \in \{1,2\}$ be a positive-definite function. Consider the system Σ_x (see (3.38)). We say that Σ_x is $\mathcal{KL}$-stable with respect to (ρ_1, ρ_2) if there exist a class-$\mathcal{KL}$ function β such that for each initial condition $x(k_0) = x$, each k_0 and any $\{w_\mathfrak{n}\}_{k_0}^{k_i-1} \subset \{w_\mathfrak{n}\}_0^\infty \in \mathcal{S}^{\mathcal{D}}_{\mathbb{K}_{\Delta k}}$ the following inequality holds*

$$\rho_1(\phi_f(k_i - k_0, x, \{w_\mathfrak{n}\}_{k_0}^{k_i-1})) \leq \beta(\rho_2(x), k_i - k_0), \tag{3.50}$$

for all $k_i \in \mathbb{K}_{\Delta k}$. □

Next, we state the proposition that establishes the desired equivalence.

Proposition 3.1. *Let $\rho_i : \mathbb{R}^{n_\mathfrak{p}} \times \mathbb{R}^{\mathfrak{h}\cdot m_\mathfrak{p}} \to \mathbb{R}_{\geq 0}$, $i \in \{1,2\}$ be a positive-definite function. Consider the system Σ_x (see (Eq. 3.38)). The following are equivalent:*

- *The system Σ_x is $\mathcal{KL}$-stable with respect to (ρ_1, ρ_2),*
- *The following holds:*
  - Uniform stability and global boundedness*: There exists a class-$\mathcal{K}_\infty$ function γ such that for each initial condition $x(k_0) = x$, each k_0 and any $\{w_\mathfrak{n}\}_{k_0}^{k_i-1} \subset \{w_\mathfrak{n}\}_0^\infty \in \mathcal{S}^{\mathcal{D}}_{\mathbb{K}_{\Delta k}}$ it holds*

$$\rho_1(\phi_f(k_i - k_0, x, \{w_\mathfrak{n}\}_{k_0}^{k_i-1})) \leq \gamma(\rho_2(x)), \tag{3.51}$$

    for each $k_i \in \mathbb{K}_{\Delta k}$,
  - Uniform global attractivity*: For each $\delta > 0$ and $\epsilon > 0$, there exists $K(\delta, \epsilon) > 0$ such that that for each initial condition $x(k_0) = x$, each k_0 and any $\{w_\mathfrak{n}\}_{k_0}^{k_i-1} \subset \{w_\mathfrak{n}\}_0^\infty \in \mathcal{S}^{\mathcal{D}}_{\mathbb{K}_{\Delta k}}$ it holds*

$$\rho_2(x) \leq \delta,\ k_i - k_0 \geq K(\delta, \epsilon) \Rightarrow \rho_1(\phi_f(k_i - k_0, x, \{w_\mathfrak{n}\}_{k_0}^{k_i-1})) \leq \epsilon \tag{3.52}$$

    for each $k_i \in \mathbb{K}_{\Delta k}$. □

Remark: The latter proposition originates from [162] and [161]. Note that in these references, functions ρ_1 and ρ_2 (denoted, respectively, as σ_1 and σ_2) are required to be continuous but we require only positive-definiteness. Additionally, note that Proposition 2.2 in [162] and our Proposition 3.1, in its proofs do not require the continuity of ρ_1 and ρ_2. Moreover, note that results in [162] concentrate on difference inclusion and indeed, even in our case, due to scheduling, we have a difference inclusion. However, due to the MPC controller, which at each time instant chooses only *one* optimal node and the corresponding control sequence, our corresponding difference inclusion reduces to a difference equation, see (3.38). Hence, the reason why we use it in Proposition 3.1. □

Having stated Proposition 3.1, we steer the focus on "uniform stability and global boundedness" and "uniform global attractivity". First, we establish two lemmas that are needed for showing the mentioned properties.

Lemma 3.2. *Let Assumptions 3.1, 3.2 and 3.3 be satisfied. Then*[2]

[2]Note that we write $u_\mathfrak{p}^{r^*}$ instead of $u_\mathfrak{p}^{*r^*}$ to simplify notation.

$$\begin{aligned}\Delta V(x(k_i)) &:= V(\phi_f(\Delta k_i + 1, x, \{w_{\mathfrak{n}}\}_{k_i}^{k_i+\Delta k_i})) - V(x) \\ &\leq -\sum_{j=k_i}^{k_i+\Delta k_i} \alpha_{x_{\mathfrak{p}}}(|\phi_{f_{\mathfrak{p}}}(j-k_i, x_{\mathfrak{p}}, \{u_{\mathfrak{p}}^{r^*}\}_{k_i}^{k_i+\Delta k_i})|),\end{aligned} \tag{3.53}$$

holds for each $x(k_i) = x$ and each k_i. □

Remark: Note that Lemma 3.2 is a relaxed version of Theorem 3 in [30] and Lemma 2 in [32] for the case of no disturbances. The relaxation comes with Assumption 3.3 in which a terminal control law κ_r has the domain $\mathbb{R}^{n_{\mathfrak{p}}} \times \mathbb{R}^{\mathfrak{h} \cdot m_{\mathfrak{p}}}$ and it exists *only* for *some* node r. This assumption encapsulates their counterparts in [30] and [32] where the domain is $\mathbb{R}^{n_{\mathfrak{p}}}$ (which does not account for buffers). Furthermore, unlike in Theorem 3 in [30] but, similarly as in Assumption 3 of [32], the terminal control law κ_r from Assumption 3.3 has to exist *only* for *some* node r and not for all nodes. □

Lemma 3.3. *Let Assumptions 3.1, 3.2 and 3.3 be satisfied. Then*

$$\begin{aligned}\Delta V(x(k_i)) &:= V(\phi_f(\Delta k_i + 1, x, \{w_{\mathfrak{n}}\}_{k_i}^{k_i+\Delta k_i})) - V(x) \\ &\leq -\sum_{j=k_i}^{k_i+\Delta k_i} \alpha_{u_{\mathfrak{p}}}(|u_{\mathfrak{p}}^{r^*}(i)|),\end{aligned} \tag{3.54}$$

holds for each $x(k_i) = x$ and each k_i. □

Finally, we have all the necessary ingredients to state lemmas that establish an appropriate $\mathcal{KL}$ bound on the plant state and control trajectories, respectively.

Lemma 3.4. *Let conditions of Lemma 3.2 and Assumption 3.4 be satisfied. Then, the system Σ_x (see (3.38)) is $\mathcal{KL}$-stable with respect to (ρ_1, ρ_2) where $\rho_1(x(\cdot)) := \alpha_{x_{\mathfrak{p}}}(|x_{\mathfrak{p}}(\cdot)|)$ and $\rho_2(\bullet) := |\bullet|$.* □

Lemma 3.5. *Let conditions of Lemma 3.3 and Assumption 3.4 be satisfied. Then, the system Σ_x (see Eq. 3.38) is $\mathcal{KL}$-stable with respect to (ρ_1, ρ_2) where $\rho_1(x(\cdot)) := \alpha_{u_{\mathfrak{p}}}(|u_{\mathfrak{p}}(\cdot)|)$ and $\rho_2(\bullet) := |\bullet|$.* □

Before stating the result that establishes UGAS-$\mathbb{K}_{\Delta k}$, we need a lemma that establishes ISS of the overall buffer (see (3.27)) with respect to the input being the data $\mathfrak{d}$ sent by the controller. This lemma is stated next.

Lemma 3.6. *Consider the overall buffer given by (3.27) and consider any sequence of dropout outcomes* $\{w_{\mathfrak{n}}\}_{k_0}^{k_i} \subset \{w_{\mathfrak{n}}\}_0^{\infty} \in \mathcal{S}^{\mathcal{D}}_{\mathbb{K}_{\Delta k}}$. *Then, the overall buffer is ISS with respect to the input being the data* $\mathfrak{d}$. *Namely, it follows that*

$$|x_{\mathfrak{b}}(k_i)| \leq \beta(|x_{\mathfrak{b}}(k_j)|, k_i - k_j) + \gamma(\sup_{k_j \leq \tau \leq k_i} |\mathfrak{d}(\tau)|)$$

where $\beta \in \mathcal{KL}$, $\gamma \in \mathcal{K}$ *and* $k_i \geq k_j \geq k_0$; *also, recall that* $\mathfrak{d}(\tau) = (u^*_{\mathfrak{p}_{r^*}}(\tau), \ldots, u^*_{\mathfrak{p}_{r^*}}(\tau + \Delta k - 1))$. □

Finally, we are ready to state the theorem that establishes UGAS-$\mathbb{K}_{\Delta k}$ for this cascade-based approach.

Theorem 3.1 (UGAS-$\mathbb{K}_{\Delta k}$ - Cascade-based approach)**.** *Consider the system* Σ_x *given by (3.38). Let the following conditions be satisfied:*

- *(Lemma 3.4) The system* Σ_x *is* $\mathcal{KL}$*-stable with respect to* (ρ_1, ρ_2) *where* $\rho_1(x(\cdot)) := \alpha_{x_{\mathfrak{p}}}(|x_{\mathfrak{p}}(\cdot)|)$ *and* $\rho_2(\bullet) := |\bullet|$,
- *(Lemma 3.5) The system* Σ_x *is* $\mathcal{KL}$*-stable with respect to* $(\tilde{\rho}_1, \rho_2)$ *where* $\tilde{\rho}_1(x(\cdot)) := \alpha_{u_{\mathfrak{p}}}(|u_{\mathfrak{p}}(\cdot)|)$ *and* $\rho_2(\bullet) := |\bullet|$,
- *(Lemma 3.6) The overall buffer (3.27) is ISS with respect to the input being the data* $\mathfrak{d}$.

Then, the system Σ_x *is UGAS-*$\mathbb{K}_{\Delta k}$. □

Lyapunov approach

In the previous approach we established UGAS-$\mathbb{K}_{\Delta k}$ of the system Σ_x (see (3.38)) by focusing on trajectories of the controller, the overall buffer and the plant. This led to an approach that is not so common in the NCS literature. On the other hand, in this section, we use a relatively standard approach of showing stability in control literature. Namely,

by constructing an appropriate Lyapunov function. We do not provide construction procedure here because the proof of the following theorem establishes UGAS-$\mathbb{K}_{\Delta k}$ of the system Σ_x via construction of an appropriate Lyapunov function.

Theorem 3.2 (UGAS-$\mathbb{K}_{\Delta k}$ - Lyapunov approach)**.** *Consider the system Σ_x given by (3.38). Consider any sequence of dropout outcomes $\{w_{\mathfrak{n}}\}_{k_0}^{k_i-1} \subset \{w_{\mathfrak{n}}\}_0^{\infty} \in \mathcal{S}^{\mathcal{D}}_{\mathbb{K}_{\Delta k}}$. Let the following conditions be satisfied:*

- *(Lemma 3.3) Dissipation inequality*

$$\begin{aligned}\Delta V(x(k_i)) &:= V(\phi_f(\Delta k_i + 1, x, \{w_{\mathfrak{n}}\}_{k_i}^{k_i+\Delta k_i})) - V(x)\\ &\leq -\sum_{j=k_i}^{k_i+\Delta k_i} \alpha_{x_{\mathfrak{p}}}(|\phi_{f_{\mathfrak{p}}}(j-k_i, x_{\mathfrak{p}}, \{u_{\mathfrak{p}}^{r^*}\}_{k_i}^{k_i+\Delta k_i})|),\end{aligned}$$

  holds for each $x(k_i) = x$ and each k_i and where V is given in (3.34),

- *(Lemma 3.5) The system Σ_x is $\mathcal{KL}$-stable with respect to (ρ_1, ρ_2) where $\rho_1(x(\cdot)) := \alpha_{u_{\mathfrak{p}}}(|u_{\mathfrak{p}}(\cdot)|)$ and $\rho_2(\bullet) := |\bullet|$.*

Then, system Σ_x is UGAS-$\mathbb{K}_{\Delta k}$. □

We proceed with the proofs of all the stated results, chronologically ordered.

3.3 Proofs

Proof of Lemma 3.1. Before we begin, note that this proof follows the lines of the proof of Theorem 2 from [2]. Consequently, instead of using the solution mapping ϕ_f for the trajectory of the system Σ_x (see (3.38)), we will use the corresponding variable x, as is done in [2].

Let the conditions of Lemma 3.1 be satisfied. Consider any $\bar{k}_0 \in \{0, \ldots, k_i + \Delta k_i\} \subset \{0, \ldots, k_i + \Delta k\}$, $k_i \in \mathbb{K}_{\Delta k}$. From the inequality (3.48) it follows that[3]

[3]Note that instead of writing $\phi_f(k - \bar{k}_0, x, \{w_{\mathfrak{n}}\}_{\bar{k}_0}^{k-1})$ we write $x(k)$ because of the reason stated at the beginning of the proof.

$$|x(k)| \leq \gamma_{\Delta k}(|x(\bar{k}_0)|), \tag{3.55}$$

holds for all $k \in \{\bar{k}_0, \ldots, \underbrace{k_i + \Delta k_i + 1}_{k_{i+1}}\} \subset \{\bar{k}_0, \ldots, \bar{k}_0 + \Delta k\}$; note that in the latter inequality, unlike in the inequality (3.48), we write $x(\bar{k}_0)$ instead of x. Next, from the property of the exponential function, namely, $\exp(T - \tau) \geq 1,\ \forall T \geq \tau$, we can express the inequality (3.55) as

$$\begin{aligned} |x(k)| &\leq \gamma_{\Delta k}(|x(\bar{k}_0)|) \exp(\Delta k - (k - \bar{k}_0)) \\ &=: \beta_1(|x(\bar{k}_0)|, k - \bar{k}_0), \end{aligned} \tag{3.56}$$

where $k \in \{\bar{k}_0, \ldots, k_{i+1}\}$. Moreover, UGB-$\Delta k$ property, i.e., the the inequality (3.55), also yields[4]

$$|x(k)| \leq \gamma_{\Delta k}(|x(k_i + \bar{k})|), \tag{3.57}$$

where $k \in \{k_i + \bar{k}, \ldots, k_i + \bar{k} + \Delta k_i\} \subset \{k_i + \bar{k}, \ldots, k_i + \bar{k} + \Delta k\},\ \bar{k} \geq 0$; again, notice that in the latter inequality, we write $x(k_i + \bar{k})$ instead of x for the initial state. Then, using both properties, UGAS-$\mathbb{K}_{\Delta k}$ and UGB-Δk, it follows[5]

$$\begin{aligned} |x(k)| &\leq \gamma_{\Delta k}(|x(k_i + \bar{k})|) \\ &\leq \gamma_{\Delta k}(\tilde{\beta}(|x(k_i)|, \bar{k})) \\ &\leq \gamma_{\Delta k}(\tilde{\beta}(\gamma_{\Delta k}(|x(\bar{k}_0)|), \bar{k})) \\ &=: \tilde{\beta}_2(|x(\bar{k}_0)|, \bar{k}). \end{aligned} \tag{3.58}$$

Now, if the function $\tilde{\beta}_2$ is a UIB function (see Section 1 for a definition), then, according to Corollary 1 from [2], it follows

[4]Please notice once again that instead of the solution mapping ϕ_f for the trajectory of the system Σ_x (see (3.38)), we use the corresponding variable x, as is done in [2].

[5]Note that in the definition of UGAS-$\mathbb{K}_{\Delta k}$ property we used symbol β for the corresponding class-$\mathcal{KL}$ function while in the upcoming inequality we use symbol $\tilde{\beta}$ instead.

$$\tilde{\beta}_2(|x(\bar{k}_0)|, \bar{k}) \leq P^{\Delta k}\tilde{\beta}_2(|x(\bar{k}_0)|, \bar{k} + \Delta k),\ P > 1. \tag{3.59}$$

On the other hand, if the function $\tilde{\beta}_2$ is not a UIB function, then, according to Lemma 1 from [2], we can upper bound it with a UIB function. Namely, we can majorize it with a UIB function $\bar{\beta}_2(s, \tau) := \max_{\eta \in \{0,\ldots,\tau\}} 2^{-\eta}\tilde{\beta}_2(s, \tau - \eta)$ with $P = 2$. Then, we can apply Corollary 1 from [2] to arrive at the inequality similar to the inequality (3.59), as desired. Moreover, since $k - \bar{k}_0 < k_i + \bar{k} + \Delta k - k_i = \bar{k} + \Delta k$ it follows

$$\begin{aligned} |x(k)| &\leq P^{\Delta k}\tilde{\beta}_2(|x(\bar{k}_0)|, \bar{k} + \Delta k) \\ &\leq P^{\Delta k}\tilde{\beta}_2(|x(\bar{k}_0)|, k - \bar{k}_0) \\ &=: \beta_2(|x(\bar{k}_0)|, k - \bar{k}_0). \end{aligned} \tag{3.60}$$

Finally, by introducing a new class-$\mathcal{KL}$ function

$$\beta(s, \tau) := \max\{\beta_1(s, \tau), \beta_2(s, \tau)\} \tag{3.61}$$

we satisfy the inequality (3.49), as desired.

□

Proof of Proposition 3.1. The proof follows the same lines as in [162]. □

Proof of Lemma 3.2. As in the proof of Theorem 3 in [30] and the relating part of Lemma 2 in [32] for the case of no disturbances, we will consider two cases (due to Assumption 3.1). The first case will be the case where $\Delta k_i < \Delta k - 1$ while the second case is where $\Delta k_i = \Delta k - 1$. In both cases, the key is to find a sequence *feasible* control values over the prediction horizon.

1. $\Delta k_i < \Delta k - 1$: Consider a node from Assumption 3.3 at time instant k_{i+1}, that is, $r(k_{i+1})$, which is not necessarily the same as the optimal node from the previous successful transmission instant k_i, i.e., $r^*(k_i)$. Consequently, consider a control sequence

$$\{\tilde{u}_{\mathfrak{p}}^{r(k_{i+1})}\}_{k_{i+1}}^{k_{i+1}+\Delta k-1} = \{u_{\mathfrak{p}}^{r^*(k_i)}(k_{i+1}), \ldots, u_{\mathfrak{p}}^{r^*(k_i)}(k_i + \Delta k - 1), \\ \tilde{u}_{\mathfrak{p}}^{r(k_{i+1})}(k_i + \Delta k), \ldots, \tilde{u}_{\mathfrak{p}}^{r(k_{i+1})}(k_{i+1} + \Delta k - 1)\} \quad (3.62)$$

where $u_{\mathfrak{p}}^{r^*(k_i)}(\cdot)$ stands for plant input with node $r^*(k_i)$ being the last updated part of plant input; i.e., the last updated control values are the ones for node $r^*(k_i)$. More precisely, the first $\Delta k - \Delta k_i - 1$ elements are from the buffers which originate from past optimizations, last one received being obtained in optimization at time instant k_i; see Section 3.1.4 for more details. The rest of elements come from Assumption 3.3, namely

$$\tilde{u}_{\mathfrak{p}}^{r(k_{i+1})}(k_{i+1} + j) := \kappa_{r(k_{i+1})}(\tilde{x}(k_{i+1} + j)), \quad (3.63)$$

for all $j \in \{\Delta k - \Delta k_i - 1, \ldots, \Delta k - 1\}$, where

$$\tilde{x}^+ = f(\tilde{x}, \tilde{w}_{\mathfrak{n}}) = \begin{bmatrix} f_{\mathfrak{p}}(\tilde{x}_{\mathfrak{p}}, \kappa_{r(k_{i+1})}(\tilde{x})) \\ \Xi(r(k_{i+1}), \tilde{w}_{\mathfrak{n}}) S \tilde{x}_{\mathfrak{b}} \end{bmatrix}, \quad (3.64)$$

with

$$\left.\begin{aligned} \tilde{w}_{\mathfrak{n}} \in \{\tilde{w}_{\mathfrak{n}}\}_{k_i+\Delta k}^{k_{i+1}+\Delta k-1} &= \{0, \ldots, 0\}, \\ \tilde{x} = \phi_f(\Delta k, x, \{w_{\mathfrak{n}}\}_{k_i}^{k_i+\Delta k-1}). \end{aligned}\right\} \quad (3.65)$$

Note that the sequence given in the equation (3.62) is a feasible one. Now, direct application of this sequence in the corresponding cost function produces the following

$$
\begin{aligned}
&J\Big(\phi_f(\Delta k_i+1, x, \{w_{\mathfrak{n}}\}_{k_i}^{k_i+\Delta k_i}), \{\tilde{u}_{\mathfrak{p}_{r(k_{i+1})}}\}_{k_{i+1}}^{k_{i+1}+\Delta k-1})\Big) \\
&\quad = V(x) - \sum_{j=k_i}^{k_i+\Delta k_i} l\Big(\phi_{f_{\mathfrak{p}}}(j-k_i, x_{\mathfrak{p}}, \{u_{\mathfrak{p}}^{r^*}\}_{k_i}^{k_i+\Delta k_i}), u_{\mathfrak{p}}^{r^*(k_i)}(j)\Big) \\
&\qquad + \sum_{j=k_i+\Delta k}^{k_{i+1}=\Delta k-1} \Big\{g\Big(f_{\mathfrak{p}}(\tilde{x}(j), \tilde{u}_{\mathfrak{p}}(j))\Big) - g\Big(\tilde{x}(j)\Big) + l\Big(\tilde{x}(j), \tilde{u}_{\mathfrak{p}}(j)\Big)\Big\} \\
&\quad \leq V(x) - \sum_{j=k_i}^{k_i+\Delta k_i} l\Big(\phi_{f_{\mathfrak{p}}}(j-k_i, x_{\mathfrak{p}}, \{u_{\mathfrak{p}}^{r^*}\}_{k_i}^{k_i+\Delta k_i}), u_{\mathfrak{p}}^{r^*(k_i)}(j)\Big)
\end{aligned}
\tag{3.66}
$$

where $\{\tilde{u}_{\mathfrak{p}_{r(k_{i+1})}}\}_{k_{i+1}}^{k_{i+1}+\Delta k-1}$ is a sequence of corresponding control values for node $r(k_{i+1})$ from a feasible control sequence given in the equation (3.62); also, we write $\{u_{\mathfrak{p}}^{r^*}\}_{k_i}^{k_i+\Delta k_i}$ instead of $\{u_{\mathfrak{p}}^{r^*(k_i)}\}_{k_i}^{k_i+\Delta k_i}$ to simplify notation. Now, due to optimality, we have

$$
\begin{aligned}
&V\Big(\phi_f(\Delta k_i+1, x, \{w_{\mathfrak{n}}\}_{k_i}^{k_i+\Delta k_i})\Big) \\
&\quad \leq J\Big(\phi_f(\Delta k_i+1, x, \{w_{\mathfrak{n}}\}_{k_i}^{k_i+\Delta k_i}), \{\tilde{u}_{\mathfrak{p}_{r(k_{i+1})}}\}_{k_{i+1}}^{k_{i+1}+\Delta k-1}\Big),
\end{aligned}
\tag{3.67}
$$

thus, application of Assumption 3.2 (note that the lower bound in terms of the plant state is used, but not the bound in terms of the plant control input), yields

$$
\begin{aligned}
\Delta V(x(k_i)) &:= V\Big(\phi_f(\Delta k_i+1, x, \{w_{\mathfrak{n}}\}_{k_i}^{k_i+\Delta k_i})\Big) - V(x) \\
&\leq \sum_{j=k_i}^{k_i+\Delta k_i} l\Big(\phi_{f_{\mathfrak{p}}}(j-k_i, x_{\mathfrak{p}}, \{u_{\mathfrak{p}}^{r^*}\}_{k_i}^{k_i+\Delta k_i}), u_{\mathfrak{p}}^{r^*(k_i)}(j)\Big) \\
&\leq -\sum_{j=k_i}^{k_i+\Delta k_i} \alpha_{x_{\mathfrak{p}}}\Big(|\phi_{f_{\mathfrak{p}}}(j-k_i, x_{\mathfrak{p}}, \{u_{\mathfrak{p}}^{r^*}\}_{k_i}^{k_i+\Delta k_i})|\Big),
\end{aligned}
\tag{3.68}
$$

as desired.

2. $\Delta k_i = \Delta k - 1$: The only difference is that now, we consider a *feasible* control sequence $\{\tilde{u}_{\mathfrak{p}}^{r(k_{i+1})}(k_{i+1}), \ldots, \tilde{u}_{\mathfrak{p}}^{r(k_{i+1})}(k_{i+1}+\Delta k-1)\}$, whose values come from Assump-

tion 3.3.

□

Proof of Lemma 3.3. Follows the exact same lines as the proof of Lemma 3.2 where, when applying Assumption 3.2 we employ the lower bound in terms of the plant control input.

□

Proof of Lemma 3.4. Let the conditions of Lemma 3.2 and Assumption 3.4 be satisfied. From the inequality (3.53) it follows

$$V(x(k_{i+1})) - V(x(k_i)) \leq -\alpha_{x_\mathfrak{p}}(|\phi_{f_\mathfrak{p}}(0, x_\mathfrak{p}, \{\})|) = -\alpha_{x_\mathfrak{p}}(|x_\mathfrak{p}(k_i)|). \tag{3.69}$$

Now, consider time instances for the set $\{k_0, \ldots, k_i\} \in \mathbb{K}_{\Delta k}$ and the corresponding inequalities of the from given in the previous inequality, namely

$$\left.\begin{aligned} V(x(k_1)) - V(x(k_0)) &\leq -\alpha_{x_\mathfrak{p}}(|x_\mathfrak{p}(k_0)|), \\ V(x(k_2)) - V(x(k_1)) &\leq -\alpha_{x_\mathfrak{p}}(|x_\mathfrak{p}(k_1)|), \\ &\vdots \\ V(x(k_i)) - V(x(k_{i-1})) &\leq -\alpha_{x_\mathfrak{p}}(|x_\mathfrak{p}(k_{i-1})|). \end{aligned}\right\} \tag{3.70}$$

Adding all previous inequalities yields

$$V(x(k_i)) \leq V(x(k_0)) - \sum_{j=0}^{i-1} \alpha_{x_\mathfrak{p}}(|x_\mathfrak{p}(k_j)|). \tag{3.71}$$

Note that due to Assumption 3.2 and the definition of cost function given in the equation (3.32) it holds that

$$V(x(k_i)) \geq \alpha_{x_\mathfrak{p}}(|x_\mathfrak{p}(k_i)|). \tag{3.72}$$

Last inequality, combined with the inequality from Assumption 3.4 and inequality given in (3.71) results in

$$\alpha_{x_\mathfrak{p}}(|x_\mathfrak{p}(k_i)|) \leq \gamma_V(|x(k_0)|) - \sum_{j=0}^{i-1} \alpha_{x_\mathfrak{p}}(|x_\mathfrak{p}(k_j)|). \tag{3.73}$$

We now proceed to show *Uniform stability and global boundedness* and *Uniform global attractivity*.

Uniform stability and global boundedness: Using the fact that $-\sum_{j=0}^{i-1} \alpha_{x_\mathfrak{p}}(|x_\mathfrak{p}(k_j)|) \leq 0$ yields

$$\alpha_{x_\mathfrak{p}}(|x_\mathfrak{p}(k_i)|) \leq \gamma_V(|x(k_0)|) \tag{3.74}$$

were according to inequality given in (3.51) we have that

$$\left.\begin{aligned} \rho_1(x(\cdot)) &:= \alpha_{x_\mathfrak{p}}(|x_\mathfrak{p}(\cdot)|), \\ \gamma(\cdot) &:= \gamma_V(\cdot), \\ \rho_2(\cdot) &:= |\cdot|, \end{aligned}\right\} \tag{3.75}$$

as desired.

Uniform global attractivity: Let us assume that there exist some $\delta > 0$ and some $\epsilon > 0$ so that for each $K(\delta, \epsilon) > 0$ and each $x(k_0)$ and $\{w_\mathfrak{n}\}_{k_0}^{k_i-1} \subset \{w_\mathfrak{n}\}_0^\infty \in \mathcal{S}^{\mathcal{D}}_{\mathbb{K}_{\Delta k}}$ it holds

$$\rho_2(x) \leq \delta,\ k_i - k_0 \geq K(\delta, \epsilon) \Rightarrow \rho_1(\phi_f(k_i - k_0, x, \{w_\mathfrak{n}\}_{k_0}^{k_i-1})) \geq \epsilon. \tag{3.76}$$

Noting that $-\rho_1(\phi_f(0, x, \{\})) = -\rho_1(x(k-i)) \leq -\epsilon$ and using (3.75) in the inequality given in (3.73) yields

$$\alpha_{x_\mathfrak{p}}(|x_\mathfrak{p}(k_i)|) \leq \gamma_V(\delta) - \epsilon \sum_{j=1}^{i} j. \tag{3.77}$$

Finally, from the latter inequality, it follow that for sufficiently large i, $\alpha_{x_\mathfrak{p}}(\cdot)$ will be upper bounded by a negative number which is a contradiction since this function is class-$\mathcal{KL}$. □

Proof of Lemma 3.5. Follows the same lines as the proof of Lemma 3.4. □

Proof of Lemma 3.6. Let the Lyapunov candidate function be of the following form

$$W(x_\mathfrak{b}) := \sum_{r=1}^{\mathrm{R}} x_{\mathfrak{b}_r}^\top \Pi_r x_{\mathfrak{b}_r}, \tag{3.78}$$

where $\mathbb{R}^{\mathfrak{h}\cdot m_{\mathfrak{p}_r}\times\mathfrak{h}\cdot m_{\mathfrak{p}_r}} \ni \Pi_r^\top = \Pi_r > 0$ for each $r \in \mathrm{R}$. Next, let us consider the difference of the previous Lyapunov function between successful time instances k_i and k_{i+1}, namely

$$\Delta W(x_{\mathfrak{b}}) := W(\phi_{f_{\mathfrak{b}}}(\Delta k_i + 1, x_{\mathfrak{b}}, \{\mathfrak{a}\}_{k_i}^{k_i+\Delta k_i}, \{\mathfrak{d}\}_{k_i}^{k_i+\Delta k_i}, \{w_{\mathfrak{n}}\}_{k_i}^{k_i+\Delta k_i})) - W(x_{\mathfrak{b}}). \tag{3.79}$$

Now, recall that $\{w_{\mathfrak{n}}\}_{k_i}^{k_i+\Delta k_i} = \{1, 0, \ldots, 0\}$; see (3.41). Thus, $\{\mathfrak{a}\}_{k_i}^{k_i+\Delta k_i} = \{r^*(k_i), \ldots, r^*(k_i)\}$, $\{\mathfrak{d}\}_{k_i}^{k_i+\Delta k_i} = \{\mathfrak{d}(k_i), S_{\mathrm{L}}\mathfrak{d}(k_i), \ldots, S_{\mathrm{L}}^{\Delta k_i}\mathfrak{d}(k_i)\}$ where $\mathfrak{d}(k_i) = (u^*_{\mathfrak{p}_{r^*}}(k_i), \ldots, u^*_{\mathfrak{p}_{r^*}}(k_i + \Delta k - 1))$; recall that L is length of each individual buffer. Now, it follows

$$\begin{aligned}
\Delta W(x_{\mathfrak{b}}) &= \sum_{\substack{r=1\\ r\neq r^*}}^{\mathrm{R}} x_{\mathfrak{b}_r}^\top \big((S^{\Delta k_i+1})^\top \Pi_r S^{\Delta k_i+1} - \Pi_r\big) x_{\mathfrak{b}_r} - x_{\mathfrak{b}_{r^*}}^\top \Pi_{r^*} x_{\mathfrak{b}_{r^*}} + \mathfrak{d}^\top \Pi_{r^*} \mathfrak{d} \\
&\leq -\sum_{\substack{r=1\\ r\neq r^*}}^{\mathrm{R}} x_{\mathfrak{b}_r}^\top \Lambda_{(r,\Delta k_i)} x_{\mathfrak{b}_r} - x_{\mathfrak{b}_{r^*}}^\top \Pi_{r^*} x_{\mathfrak{b}_{r^*}} + \mathfrak{d}^\top \Pi_{r^*} \mathfrak{d} \\
&\leq -\underbrace{\min_r\{\min_{\Delta k_i}\{\Lambda_{(r,\Delta k_i)}\}, |\Pi_r|\}}_{c_1} \sum_{r=1}^{\mathrm{R}} |x_{\mathfrak{b}_r}|^2 + \underbrace{\max_r\{|\Pi_r|\}}_{c_2} |\mathfrak{d}|^2 \\
&\leq -c_1|x_{\mathfrak{b}}|^2 + c_2|\mathfrak{d}|^2 \\
&\leq -\frac{c_1}{2}|x_{\mathfrak{b}}|^2,\ \forall |x_{\mathfrak{b}}| \geq \sqrt{\frac{2c_2}{c_1}}|\mathfrak{d}|,
\end{aligned} \tag{3.80}$$

where $(S^{\Delta k_i+1})^\top \Pi_r S^{\Delta k_i+1} - \Pi_r = -\Lambda_{(r,\Delta k_i)}$, for each $r \in \mathrm{R}$ and where $\mathbb{R}^{\mathfrak{h}\cdot m_{\mathfrak{p}_r}\times\mathfrak{h}\cdot m_{\mathfrak{p}_r}} \ni \Lambda_{(r,\Delta k_i)}^\top = \Lambda_{(r,\Delta k_i)} > 0$; this is due to the fact that for any Δk_i, satisfying $1 \leq \Delta k_i \leq \Delta k - 1$, the resulting matrix $S^{\Delta k_i+1}$ is nilpotent, and thus, it has all of its eigenvalues located strictly inside the unit circle. □

Proof of Theorem 3.1. Let the conditions of Lemmas 3.4, 3.5 and 3.6 be satisfied. Consider the initial successful transmission instant $k_0 \in \mathbb{K}_{\Delta k}$. Conclusions of Lemmas 3.6 and 3.4 provide

$$\left.\begin{aligned}
|x_{\mathfrak{b}}(k_i)| \leq \beta_1(|x_{\mathfrak{b}}(k_j)|, k_i - k_j) + \gamma_1(\sup_{k_j\leq\tau\leq k_i} |\mathfrak{d}(\tau)|) \\
|x_{\mathfrak{p}}(k_i)| \leq \beta_2(|x_{\mathfrak{p}}(k_i)|, k_i - k_j)
\end{aligned}\right\} \tag{3.81}$$

where $k_i \geq k_j \geq k_0$, β_1 and β_2 are class-$\mathcal{KL}$ and γ_1 is a class-$\mathcal{K}$ function. Now, recall that $\mathfrak{d}(\tau) = (u^*_{\mathfrak{p}_{r^*}}(\tau), \ldots, u^*_{\mathfrak{p}_{r^*}}(\tau + \Delta k - 1))$ and note that the conclusion of Lemma 3.5 results in

$$|u_{\mathfrak{p}}(\tau)| \leq \beta_3(|x(k_j)|, \tau - k_j) \tag{3.82}$$

where β_3 is a class-$\mathcal{KL}$ function. Now, due to the properties of class-$\mathcal{KL}$ function it follows $\sum_{j=\tau}^{\tau+\Delta k-1} |u_{\mathfrak{p}}(j)| \leq (\Delta k - 1)\beta_3(|x(k_j)|, \tau - k_j)$. Further, due to properties of Euclidean norm, $|\mathfrak{d}(\tau)| \leq \sum_{j=\tau}^{\tau+\Delta k-1} |u_{\mathfrak{p}}(j)|$, thus we have

$$|\mathfrak{d}(\tau)| \leq (\Delta k - 1)\beta_3(|x(k_j)|, \tau - k_j). \tag{3.83}$$

Further, let $k_j = \lceil \frac{k_i - k_0}{2} \rceil$. Then

$$|x_{\mathfrak{b}}(k_i)| \leq \beta_1\Big(\Big|x_{\mathfrak{b}}\Big(\Big\lceil \frac{k_i + k_0}{2} \Big\rceil\Big)\Big|, \Big\lceil \frac{k_i - k_0}{2} \Big\rceil\Big) + \gamma\Big(\sup_{k_j \leq \tau \leq k_i} |\mathfrak{d}(\tau)|\Big). \tag{3.84}$$

To estimate $|x_{\mathfrak{b}}(k_j)|$, we apply the first inequality of (3.81) with $k_j = k_0$ and k_i replaced with k_j, which yields

$$|x_{\mathfrak{b}}(k_j)| \leq \beta_1\Big(|x_{\mathfrak{b}}(k_0)|, \Big\lceil \frac{k_i - k_0}{2} \Big\rceil\Big) + \gamma\Big(\sup_{k_0 \leq \tau \leq k_j} |\mathfrak{d}(\tau)|\Big). \tag{3.85}$$

Using the inequality (3.83), results in

$$\left.\begin{aligned} \sup_{k_0 \leq \tau \leq k_j} |\mathfrak{d}(\tau)| &\leq (\Delta k - 1)\beta_3(|x(k_0)|, 0), \\ \sup_{k_j \leq \tau \leq k_i} |\mathfrak{d}(\tau)| &\leq (\Delta k - 1)\beta_3\Big(|x(k_0)|, \Big\lceil \tfrac{k_i - k_0}{2} \Big\rceil\Big). \end{aligned}\right\} \tag{3.86}$$

Substituting the inequality (3.85) in the inequality (3.84) and using the inequality (3.86) together with the inequalities $|x_{\mathfrak{p}}(k_0)| \leq |x(k_0)|$, $|x_{\mathfrak{b}}(k_0)| \leq |x(k_0)|$ and $|x(k_i)| \leq |x_{\mathfrak{p}}(k_i)| + |x_{\mathfrak{b}}(k_i)|$, yields

$$|x(k_i)| \leq \beta_4(|x(k_0)|, k_i - k_0), \tag{3.87}$$

where

$$\begin{aligned}&\beta_4(t,s)\\&\quad=\beta_1\Big(\beta_1\Big(t,\frac{s}{2}\Big)+\gamma_1((\Delta k-1)\beta_3(t,0)),\frac{s}{2}\Big)+\gamma_1\Big((\Delta k-1)\beta_3\Big(t,\frac{s}{2}\Big)\Big)+\beta_2(t,s),\end{aligned}\tag{3.88}$$

as desired. □

Proof of Theorem 3.2. Let the conditions of Lemmas 3.3 and 3.5 be satisfied. Let the Lyapunov function be of the following form

$$L(x):=cU(x)+V(x)+W(x_{\mathfrak{b}}),\ c>0\tag{3.89}$$

where $V(x)$ is given in the equation (3.34), $W(x)$ is given in the equation (3.78), while $U(x)$ is defined as follows

$$U(x):=\sum_{j=k}^{\infty}v_{\mathfrak{p}}^{\top}(\phi_f(j-k,x,\{w_{\mathfrak{n}}\}_k^{k+j}))\Omega v_{\mathfrak{p}}(\phi_f(j-k,x,\{w_{\mathfrak{n}}\}_k^{k+j})),\ \forall j\geq k\geq 0,\tag{3.90}$$

where $\mathbb{R}^{m_{\mathfrak{p}}\times m_{\mathfrak{p}}}\ni\Omega^{\top}=\Omega>0$, $v_{\mathfrak{p}}(\phi_f(j-k,x,\{w_{\mathfrak{n}}\}_k^{k+j})):=u_{\mathfrak{p}}(j)$ and $v_{\mathfrak{p}}:\mathbb{R}^{n_{\mathfrak{p}}\times\mathfrak{h}\cdot m_{\mathfrak{p}}}\to\mathbb{R}^{m_{\mathfrak{p}}}$ is possibly a *discontinuous* function.

One can easily show that both, V and W are radially unbounded. Namely, for V, one would use assumptions 3.2 and 3.4, while for W, one would use the fact that the function is quadratic. Moreover, since V and U have the same argument and V is radially unbounded there is no need to show that U is radially unbounded. Hence, we conclude that L is radially unbounded.

Now, we proceed with analyzing the difference of L between time instances k_i and k_{i+1}, namely

$$\begin{aligned}\Delta L(x(k_i))&:=L(\phi_f(\Delta k_i+1,x,\{w_{\mathfrak{n}}\}_{k_i}^{k_i+\Delta k_i}))-L(x(k_i))\\&=c\Delta U(x(k_i))+\Delta V(x(k_i))+\Delta W(x_{\mathfrak{b}}(k_i))\end{aligned}\tag{3.91}$$

Note that $\Delta V(x(k_i))$ is considered in the inequality (3.68) while $\Delta W(x_{\mathfrak{b}}(k_i))$ is considered

in the inequality (3.80). Thus, we will concentrate on $\Delta U(x(k_i))$ resulting in

$$
\begin{aligned}
\Delta U(x(k_i)) &:= U(\phi_f(\Delta k_i + 1, x, \{w_{\mathfrak{n}}\}_{k_i}^{k_i+\Delta k_i})) - U(x(k_i)) \\
&\leq \sum_{j=k_{i+1}}^{\infty} v_{\mathfrak{p}}^{\top}(\phi_f(j-k, x, \{w_{\mathfrak{n}}\}_k^{k+j}))\Omega v_{\mathfrak{p}}(\phi_f(j-k, x, \{w_{\mathfrak{n}}\}_k^{k+j})) \\
&\quad - \sum_{j=k_i}^{\infty} v_{\mathfrak{p}}^{\top}(\phi_f(j-k, x, \{w_{\mathfrak{n}}\}_k^{k+j}))\Omega v_{\mathfrak{p}}(\phi_f(j-k, x, \{w_{\mathfrak{n}}\}_k^{k+j})) \\
&\leq - \sum_{j=k_i}^{k_i+\Delta k_i} v_{\mathfrak{p}}^{\top}\phi_f(j-k, x, \{w_{\mathfrak{n}}\}_k^{k+j}))\Omega v_{\mathfrak{p}}(\phi_f(j-k, x, \{w_{\mathfrak{n}}\}_k^{k+j})) \\
&\leq -\lambda_{\min}(\Omega) \sum_{j=k_i}^{k_i+\Delta k_i} |u_{\mathfrak{p}}(j)|^2 \\
&\leq -\lambda_{\min}(\Omega)|\mathfrak{d}(k_i)|^2
\end{aligned}
\tag{3.92}
$$

Finally, using latter inequality together with inequalities provided in (3.68) and (3.80) and choosing c so that $(c\lambda_{\min}(\Omega) - c_2) \geq 0$, yields

$$
\begin{aligned}
\Delta L(x(k_i)) &\leq -\alpha_{x_{\mathfrak{p}}}(x_{\mathfrak{p}}(k_i)) - c_1|x_{\mathfrak{b}}(k_i)|^2 - (c\lambda_{\min}(\Omega) - c_2)|\mathfrak{d}(k_i)|^2 \\
&\leq -\alpha_{x_{\mathfrak{p}}}(x_{\mathfrak{p}}(k_i)) - c_1|x_{\mathfrak{b}}(k_i)|^2,
\end{aligned}
\tag{3.93}
$$

as desired. □

This page intentionally left blank.

Chapter 4

Stability with respect to Packet Dropouts and Scheduling - Economic MPC

The stability result from the previous chapter is being extended to the case where the cost function (e.g., see (3.32)) is more general. In particular, the cost function need not be a positive definite function with respect to the corresponding equilibrium point; e.g., see Definition 1.1. This scenario corresponds to the so-called Economic MPC.

More precisely, recall that for the standard MPC the following holds for the corresponding cost function $l : \mathbb{R}^{n_\mathfrak{p}} \times \mathbb{R}^{m_\mathfrak{p}} \to \mathbb{R}$

$$0 = l(0^{n_\mathfrak{p}}, 0^{m_\mathfrak{p}}) \leq l(x_\mathfrak{p}, u_\mathfrak{p}) \text{ for any admissible } (x_\mathfrak{p}, u_\mathfrak{p}), \tag{4.1}$$

where $x_\mathfrak{p}$ is the state and $u_\mathfrak{p}$ is the input of the corresponding plant. On the other hand, for the Economic MPC, the inequality (4.1) cannot be generally assumed and the following can happen[1]

$$\left.\begin{array}{c} 0 < l(0^{n_\mathfrak{p}}, 0^{m_\mathfrak{p}}) \text{ or,} \\ l(0^{n_\mathfrak{p}}, 0^{m_\mathfrak{p}}) > l(x_\mathfrak{p}, u_\mathfrak{p}) \text{ for any admissible } (x_\mathfrak{p}, u_\mathfrak{p}), \end{array}\right\} \tag{4.2}$$

where $(x_\mathfrak{p}, u_\mathfrak{p})$ does not correspond to any equilibrium point[2].

Theoretically, Economic MPC is important since it considers more general cost functions. Furthermore, it enables one to additionally and explicitly address the economical

[1]Note that in both inequalities, (4.1) and (4.2), we assume that the origin is the equilibrium of the corresponding system, i.e., we apply the appropriate change of coordinates (if necessary); see (3.3)–(3.5).

[2]The latter discussion is borrowed from [149]; see the second paragraph in the second section in [149].

cost of plant operation, for instance, the economical cost related to transition from one set-point to another in industrial plants; note that standard MPC focuses only control performance, e.g., track as quick as possible a set-point, and ignores the corresponding economical cost. In NCS setting, for instance, Economic MPC can be used to minimize the price related to the usage of network resources.

Unfortunately, the fundamental difference between the standard and Economic MPC (see the inequalities in (4.2)) is a sufficient condition that the extensive collection of results for the standard MPC stability analysis does not simply extend to Economic MPC. However, a noticeable progress has been documented in [147–156]. This chapter enlarges the latter collection of results by adding a result applicable to NCS. In particular we extend our results by considering a stage function that captures the economic aspects of control system design, i.e., we consider Economic MPC. Indeed, here we consider a basic Economic MPC which hopefully will be a good starting point for further extensions which would include Economic MPC with periodic terminal constraints and/or average constraints.

The considered NCS architecture is identical as the one in Chapter 3. The only difference is that the corresponding NCS is governed by Economic MPC. Again the network induces the issues of packet dropouts and scheduling and our interest is to establish stability with respect to these issues. We accomplish this by extending some results from [153] so that we can use our results established in Chapter 3 "off the shelf". In particular, we first extend the definition of strict dissipativity introduced in [149,153]. Then, inspired with the ideas in [153], we formulate the corresponding stage and terminal cost function. These are formulated in such a way that only mild changes in the assumption related to terminal control law are needed; there is an extra term in the corresponding inequality. Finally, this enables us to directly apply the stability results we established in Chapter 3.

Before outlining this chapter note that because the considered NCS is identical to the one presented in the previous chapter we will not present the model once again. As mentioned above, the only difference is that the system is governed with Economic instead of standard MPC controller. However, for the sake of the flow of presentation let us rewrite the closed loop system, namely

$$\Sigma_x : x^+ = \begin{bmatrix} f_{\mathfrak{p}}(x_{\mathfrak{p}}, h_{\mathfrak{b}}(x_{\mathfrak{b}}, r^*, \{u^*_{r^*}\}_k^{k+\mathfrak{h}-1}, w_{\mathfrak{n}})) \\ f_{\mathfrak{b}}(x_{\mathfrak{b}}, r^*, \{u^*_{r^*}\}_k^{k+\mathfrak{h}-1}, w_{\mathfrak{n}}) \end{bmatrix} =: f(x, w_{\mathfrak{n}}), \tag{4.3}$$

and bear in mind that the control includes economic aspects. So, in Section 4.1 we provide the main discussion including definitions, assumptions and results while Section 4.2 provides the corresponding proofs.

4.1 UGAS-Economic MPC

To establish UGAS for the system Σ_x (see (4.3)), governed with an Economic MPC controller, we will use the ideas from [153] and the stability results established for the standard MPC in the previous chapter. First note that the results established in [153] used plant in original coordinates, e.g., see (3.4), where the equilibrium point is not necessarily the origin. However, without loss of the generality and with a slight abuse of the notation (with respect to the results established in [153]), we will assume that the appropriate change of coordinates were applied beforehand so that the equilibrium point is at the origin (the corresponding equations change slightly and thus we omit the corresponding derivations). Moreover, with regards to the stability results established in the previous chapter, which are to be used in the sequel, we will focus only on *Cascade-based approach*.

As mentioned in the previous paragraph, we will use the ideas form [153]. The results that are established in the latter reference rely on the so-called *strict dissipativity* property of the corresponding system. Consequently, we adopt this property and modify it so it fits our needs. Let us begin by providing the corresponding definition.

Definition 4.1 (Strictly dissipative system [149, 153])**.** *The system $\Sigma_{\mathfrak{p}}$, see (3.2), is* strictly dissipative *with respect to supply rate $s : \mathbb{R}^{n_{\mathfrak{p}}} \times \mathbb{R}^{m_{\mathfrak{p}}} \to \mathbb{R}$, if there exist a* storage *function $\lambda : \mathbb{R}^{n_{\mathfrak{p}}} \to \mathbb{R}$ and a positive definite function $\rho : \mathbb{R}^{n_{\mathfrak{p}}} \times \mathbb{R}^{m_{\mathfrak{p}}} \to \mathbb{R}_{\geq 0}$ such that*

$$\lambda(f(x_{\mathfrak{p}}, u_{\mathfrak{p}})) - \lambda(x_{\mathfrak{p}}) \leq -\rho(x_{\mathfrak{p}}, u_{\mathfrak{p}}) + s(x_{\mathfrak{p}}, u_{\mathfrak{p}}) \tag{4.4}$$

holds for all $(x_{\mathfrak{p}}, u_{\mathfrak{p}}) \in \mathbb{R}^{n_{\mathfrak{p}}} \times \mathbb{R}^{m_{\mathfrak{p}}}$. □

Note that unlike in [149, 153] the function $\rho(\cdot)$ has as an argument both, state and control. The definition is tailored according to the requirements of the stability results from the previous chapter.

Remark: Note that by assuming a lower bound only in terms of the state, e.g., $\exists \bar{\rho} \in \mathcal{PD}, \bar{\rho} : \mathbb{R}^{n_{\mathfrak{p}}} \to \mathbb{R}$ such that $\rho(x_{\mathfrak{p}}, u_{\mathfrak{p}}) \geq \bar{\rho}(x_{\mathfrak{p}})$, our definition subsumes the definition given in [149, 153]. □

Moreover, let us illustrate the notion of strictly dissipative system stated in Definition 4.1 via an example. Namely, we will use the same example as in [149] where the notion of strictly dissipative system is introduced. We will not repeat the entire example but only the important parts; for all details please see [149].

Example 4.1 (Example 4.3 from [149])**.** *Consider the following scalar linear system*

$$x_{\mathfrak{p}}^{+} = \alpha x_{\mathfrak{p}} + (1-\alpha) u_{\mathfrak{p}}, \tag{4.5}$$

where $\alpha \in [0, 1)$ is a parameter to be discussed later, along with the non-convex cost functional

$$l(x_{\mathfrak{p}}, u_{\mathfrak{p}}) = (x_{\mathfrak{p}} + \frac{u_{\mathfrak{p}}}{3})(2u_{\mathfrak{p}} - x_{\mathfrak{p}}) + (x_{\mathfrak{p}} - u_{\mathfrak{p}})^4. \tag{4.6}$$

Notice that, regardless of α, for each input $u_{\mathfrak{p}}$, there exist a unique equilibrium $x_{\mathfrak{p}}^{e} = u_{\mathfrak{p}}$. Moreover,

$$l(x_{\mathfrak{p}}, u_{\mathfrak{p}})|_{x_{\mathfrak{p}}=u_{\mathfrak{p}}} = \frac{4u_{\mathfrak{p}}^2}{3} \tag{4.7}$$

so that $(x_{\mathfrak{p}}^{s}, u_{\mathfrak{p}}^{s}) = (0, 0)$ is the best steady-state[3] *and $l(x_{\mathfrak{p}}^{s}, u_{\mathfrak{p}}^{s}) = 0$. The point $(0, 0)$ is not, however, the global minimum of $l(x_{\mathfrak{p}}, u_{\mathfrak{p}})$, which in fact has two global minima for*

$$(x_{\mathfrak{p}}, u_{\mathfrak{p}}) = \pm\left(\frac{21\sqrt{6}}{64}, \frac{7\sqrt{6}}{192}\right).$$

In fact $(0, 0)$ is a saddle-point of $l(x_{\mathfrak{p}}, u_{\mathfrak{p}})$ and the level-set $L_0 = \{(x_{\mathfrak{p}}, u_{\mathfrak{p}}) | l(x_{\mathfrak{p}}, u_{\mathfrak{p}}) = 0\}$ is in $(0, 0)$ tangent to the lines of equation $u_{\mathfrak{p}} = -3x_{\mathfrak{p}}$ and $u_{\mathfrak{p}} = \frac{x_{\mathfrak{p}}}{2}$. Notice that zero-average period-2 solutions of (4.5) are possible. these correspond to the input sequences of alternating

[3] According to conventions used in [149], $(x_{\mathfrak{p}}^{s}, u_{\mathfrak{p}}^{s})$ is the best feasible pair of the equilibrium state and the associated control input.

signs, namely $+u_{\mathfrak{p}}, -u_{\mathfrak{p}}, +u_{\mathfrak{p}}, -u_{\mathfrak{p}}, \ldots$, *with the resulting periodic state sequence*

$$-\frac{(1+\alpha)}{(1-\alpha)}u_{\mathfrak{p}}, \frac{(1+\alpha)}{(1-\alpha)}u_{\mathfrak{p}}, -\frac{(1+\alpha)}{(1-\alpha)}u_{\mathfrak{p}}, \cdots .$$

Choosing $\alpha = 0$ *or sufficiently small, yields period-2 solutions which, suitably tunning the input amplitude* $u_{\mathfrak{p}}$, *belong to the sublevel-set* $L_{\leq 0} = \{(x_{\mathfrak{p}}, u_{\mathfrak{p}}) | l(x_{\mathfrak{p}}, u_{\mathfrak{p}}) \leq 0\}$, *thus outperforming the best steady state. Under such circumstances, one cannot expect dissipativity to hold. For large values of* α, *however, the period-2 solution leave* $L_{\leq 0}$. *One may, therefore wonder for which values of* α *(if any) the system (4.5) fulfills strict dissipativity.*

Correspondingly, as in [149], let us consider as a candidate function for a storage function the quadratic function $\lambda(x_{\mathfrak{p}}) = kx_{\mathfrak{p}}^2$. *Moreover, let the supply function given as* $\rho(x_{\mathfrak{p}}, u_{\mathfrak{p}}) := \epsilon(x_{\mathfrak{p}}^2 + u_{\mathfrak{p}}^2)$, $\epsilon > 0$. *Then, following the exact same procedure as in [149], one can show that strict dissipativity holds for all* $\alpha \in [0, 1)$ *for which there exist* k *and* $\epsilon > 0$ *so that*

$$k(x_{\mathfrak{p}}^+)^2 - kx_{\mathfrak{p}}^2 \leq -\epsilon(x_{\mathfrak{p}}^2 + u_{\mathfrak{p}}^2) + l(x_{\mathfrak{p}}, u_{\mathfrak{p}}), \tag{4.8}$$

holds for all $(x_{\mathfrak{p}}, u_{\mathfrak{p}}) \in \mathbb{R}^2$. *Similarly as in [149], in order to show that (4.8) holds it is enough to show that*

$$\left(x_{\mathfrak{p}} + \frac{u_{\mathfrak{p}}}{3}\right)(2u_{\mathfrak{p}} - x_{\mathfrak{p}}) + kx_{\mathfrak{p}}^2 - k(\alpha x_{\mathfrak{p}} + (1-\alpha)u_{\mathfrak{p}})^2 \geq \epsilon(x_{\mathfrak{p}}^2 + u_{\mathfrak{p}}^2) > 0.$$

Namely, the corresponding matrix

$$Q = \begin{bmatrix} k(1-\alpha^2) - 1 & \frac{5}{6} - k\alpha(1-\alpha) \\ \frac{5}{6} - k\alpha(1-\alpha) & \frac{2}{3} - k(1-\alpha)^2 \end{bmatrix}, \tag{4.9}$$

remains the same. □

Now, as mentioned above, in order to establish UGAS for the system Σ_x (see (4.3)), governed with an Economic MPC controller, we will focus only on *Cascade-based approach*; see Subsection 3.2.2 - Cascade-based approach. Correspondingly, we proceed with outlining the sufficient assumptions needed to establish UGAS for system Σ_x, (see (4.3)) governed with Economic MPC controller.

Assumptions

Before stating the assumptions we will remind the reader that the same remarks hold for the horizon in MPC controller; see the remark just after the equation (3.42). Succinctly, the horizon $\mathfrak{h}$ is set to be equal to the bound on the number of consecutive packet dropouts, that is, $\mathfrak{h} := \Delta k$ for each $r \in \mathcal{R}$.

The first assumption is related to the *strict dissipativity* of the system $\Sigma_{\mathfrak{p}}$, see (3.2).

Assumption 4.1 (Strictly dissipative system)**.** *The system $\Sigma_{\mathfrak{p}}$, see (3.2), is strictly dissipative with the supply rate*

$$s(x_{\mathfrak{p}}, u_{\mathfrak{p}}) := l(x_{\mathfrak{p}}, u_{\mathfrak{p}}) - l(0^{n_{\mathfrak{p}}}, 0^{m_{\mathfrak{p}}}), \tag{4.10}$$

and there exist class-$\mathcal{K}_\infty$ functions $\rho_{x_{\mathfrak{p}}}$ and $\rho_{u_{\mathfrak{p}}}$ such that

$$\left.\begin{aligned} \rho(x_{\mathfrak{p}}, u_{\mathfrak{p}}) &\geq \rho_{x_{\mathfrak{p}}}(|x_{\mathfrak{p}}|), \\ \rho(x_{\mathfrak{p}}, u_{\mathfrak{p}}) &\geq \rho_{u_{\mathfrak{p}}}(|u_{\mathfrak{p}}|), \end{aligned}\right\} \tag{4.11}$$

holds for each $(x_{\mathfrak{p}}, u_{\mathfrak{p}}) \in \mathbb{R}^{n_{\mathfrak{p}}} \times \mathbb{R}^{m_{\mathfrak{p}}}$. □

The reason we assume lower bounds to be of class-$\mathcal{K}_\infty$ comes from the fact that we are interested in global property, that is, UGAS. In future work the corresponding UGAS for Standard MPC (which is used to establish desired property for the case of Economic MPC) will be established with weaker assumptions which will enable usage of results from [153] without imposition of class-$\mathcal{K}_\infty$ lower bounds.

We proceed with the assumption on terminal control law which is instrumental in establishing dissipation inequalities in terms of state and control, e.g., see Lemma 3.2 and Lemma 3.3.

Assumption 4.2 (Terminal control law)**.** *For some node $r \in \mathcal{R}$ there exist a* terminal control law $\kappa_r : \mathbb{R}^{n_{\mathfrak{p}} \times \mathfrak{h} \cdot m_{\mathfrak{p}}} \to \bar{\mathbb{U}}_{\mathfrak{p}_r}$ *such that*

$$\left.\begin{aligned} g(f_{\mathfrak{p}}(x_{\mathfrak{p}}, \kappa_r(x_{\mathfrak{p}}, x_{\mathfrak{b}}))) - g(x_{\mathfrak{p}}) + l(x_{\mathfrak{p}}, \kappa_r(x_{\mathfrak{p}}, x_{\mathfrak{b}})) - l(0^{n_{\mathfrak{p}}}, 0^{m_{\mathfrak{p}}}) \leq 0, \\ f_{\mathfrak{p}}(x_{\mathfrak{p}}, \kappa_r(x_{\mathfrak{p}}, x_{\mathfrak{b}})) \in \mathbb{R}^{n_{\mathfrak{p}}}, \\ \kappa_r(x_{\mathfrak{p}}, x_{\mathfrak{b}}) \in \bar{\mathbb{U}}_{\mathfrak{p}_r}, \end{aligned}\right\} \tag{4.12}$$

holds for each $(x_{\mathfrak{p}}, x_{\mathfrak{b}}) \in \mathbb{R}^{n_{\mathfrak{p}} \times \mathfrak{h} \cdot m_{\mathfrak{p}}}$, *where,* $\bar{\mathbb{U}}_{\mathfrak{p}_r} := 0^{m_{\mathfrak{p}_1}} \times \cdots \times 0^{m_{\mathfrak{p}_{r-1}}} \times \mathbb{R}^{m_{\mathfrak{p}_r}} \times 0^{m_{\mathfrak{p}_{r+1}}} \times \cdots \times 0^{m_{\mathfrak{p}_R}}$. □

Remark: Recall that since we are considering Economic MPC, $l(0^{n_{\mathfrak{p}}}, 0^{m_{\mathfrak{p}}})$ is not necessarily equal to zero as is the case in Standard MPC. □

Results

Let us start by recalling that in order to establish UGAS for system Σ_x (4.3), we will exploit the ideas from [153] and stability results established in see Subsection 3.2.2 - Cascade-based approach.

The crucial part in Subsection 3.2.2 - Cascade-based approach, is to establish the corresponding dissipation inequalities. Correspondingly, following [153], first, we define the *rotated* stage and terminal cost functions and then establish the desired dissipation inequalities. Once that is done, the desired UGAS for system Σ_x (4.3) follows directly from the analysis done in the previous chapter, i.e., Subsection 3.2.2 - Cascade-based approach.

Let us proceed with providing the definitions of the *rotated* stage and terminal cost functions, namely

$$\left.\begin{aligned} \bar{l}(x_{\mathfrak{p}}, u_{\mathfrak{p}}) &:= l(x_{\mathfrak{p}}, u_{\mathfrak{p}}) - l(0^{n_{\mathfrak{p}}}, 0^{m_{\mathfrak{p}}}) + \lambda(x_{\mathfrak{p}}) - \lambda(f_{\mathfrak{p}}(x_{\mathfrak{p}}, u_{\mathfrak{p}})), \\ \bar{g}(x_{\mathfrak{p}}) &:= g(x_{\mathfrak{p}}) - g(0^{n_{\mathfrak{p}}}) + \lambda(x_{\mathfrak{p}}) - \lambda(0^{n_{\mathfrak{p}}}), \end{aligned}\right\} \tag{4.13}$$

which results in the following *rotated* cost function

$$\begin{aligned} &\bar{J}(x, \{\tilde{u}_{\mathfrak{p}_r}\}_k^{k+\mathfrak{h}-1}) := \\ &\sum_{i=k}^{k+\mathfrak{h}-1} \bar{l}(\phi_{f_{\mathfrak{p}}}(i-k, x_{\mathfrak{p}}, \{\tilde{u}_{\mathfrak{p}}^r\}_k^{k+\mathfrak{h}-1}), \tilde{u}_{\mathfrak{p}}^r(i)) \bar{g}(\phi_{f_{\mathfrak{p}}}(\mathfrak{h}, x_{\mathfrak{p}}, \{\tilde{u}_{\mathfrak{p}}^r\}_k^{k+\mathfrak{h}-1})). \end{aligned} \tag{4.14}$$

Notice that the computation procedure of the optimal node and the corresponding controls is not affected with the change of the definition of the cost function. Thus, we omit rewriting it; see Subsection 3.1.4. Furthermore, in the sequel, (notation-wise) in order to make distinction between optimal value function for a node (see (3.33)) and the optimal value function (see (3.34)) for a Standard MPC and an Economic MPC, we put a

bar above the corresponding symbols; note that we do not change the symbols for the resulting optimal node and corresponding controls.

Finally, before presenting the corresponding result we state Lemma 9 from [153] which is used in establishing desired dissipation inequalities.

Lemma 4.1 (Modified terminal cost [153])**.** *The pair* $(\bar{g}, \bar{l})$ *satisfies the following property if and only if* (g, l) *satisfies Assumption 4.2.*

$$\bar{g}(f_{\mathfrak{p}}(x_{\mathfrak{p}}, \kappa_r(x_{\mathfrak{p}}, x_{\mathfrak{b}}))) \leq \bar{g}(x_{\mathfrak{p}}) - \bar{l}(x_{\mathfrak{p}}, \kappa_r(x_{\mathfrak{p}}, x_{\mathfrak{b}})) \tag{4.15}$$

for all $(x_{\mathfrak{p}}, x_{\mathfrak{b}}) \in \mathbb{R}^{n_{\mathfrak{p}}} \times \mathbb{R}^{\mathfrak{h} \cdot m_{\mathfrak{p}}}$. □

UGAS

As mentioned in the introduction of this chapter, we extend our results established in the previous chapter to the case of Economic MPC. With the aid of the ideas from [153], this amounts to applying the results established in the previous chapter "off the shelf". However, to do so we need the desired dissipation inequalities which we state next.

Lemma 4.2. *Let Assumptions 3.1, 4.1, 4.2 and conditions of Lemma 4.1 be satisfied. Then*

$$\begin{aligned} \Delta\bar{V}(x(k_i)) &:= \bar{V}(\phi_f(\Delta k_i + 1, x, \{w_{\mathfrak{n}}\}_{k_i}^{k_i+\Delta k_i})) - \bar{V}(x) \\ &\leq - \sum_{j=k_i}^{k_i+\Delta k_i} \rho_{x_{\mathfrak{p}}}(|\phi_{f_{\mathfrak{p}}}(j - k_i, x_{\mathfrak{p}}, \{u_{\mathfrak{p}}^{r^*}\}_{k_i}^{k_i+\Delta k_i})|), \end{aligned} \tag{4.16}$$

for each $k_i \in \mathbb{K}_{\Delta k}$. □

Lemma 4.3. *Let Assumptions 3.1, 4.1, 4.2 and conditions of Lemma 4.1 be satisfied. Then*

$$\begin{aligned} \Delta\bar{V}(x(k_i)) &:= \bar{V}(\phi_f(\Delta k_i + 1, x, \{w_{\mathfrak{n}}\}_{k_i}^{k_i+\Delta k_i})) - \bar{V}(x) \\ &\leq - \sum_{j=k_i}^{k_i+\Delta k_i} \rho_{u_{\mathfrak{p}}}(|u_{\mathfrak{p}}^{r^*}(i)|), \end{aligned} \tag{4.17}$$

for each $k_i \in \mathbb{K}_{\Delta k}$. □

Now, as mentioned above, with the latter two lemmas our UGAS results established in the previous chapter can be applied "off the self" to show UGAS for the system Σ_x (see (4.3)), governed with an Economic MPC controller. Thus, we omit restating the corresponding lemmas and the theorem, i.e., see Lemma 3.4–3.6 and and Theorem 3.1. Moreover, for the same reasons, we also omit the Lyapunov-based approach; notice that in Lyapunov case only Lemma 4.2 would be used.

Finally, similarly as in the previous chapter, we proceed with the proofs of all the stated results, chronologically ordered.

4.2 Proofs

Proof of Lemma 4.1. Proof of Lemma 4.1 follows the exact same lines as proof of the Lemma 9 in [153] and thus is omitted. □

Proof of Lemma 4.2. The proof of Lemma 4.2 is very similar to the proof of Lemma 3.2.

Again, there are two cases, namely $\Delta k_i \leq \Delta k - 1$ and $\Delta k_i = \Delta k - 1$. Since, the second case is very simple we will omit it; recall that the only change is different feasible sequence.

Let us consider a node from Assumption 4.2 at time instant k_{i+1}, i.e., $r(k_{i+1})$, which is not necessarily the same as $r^*(k_i)$. Similarly as in the proof of Lemma 3.2, let us consider a control sequence

$$\begin{aligned}&\{\tilde{u}_{\mathfrak{p}}^{r(k_{i+1})}\}_{k_{i+1}}^{k_{i+1}+\Delta k-1}\\&= \{u_{\mathfrak{p}}^{r^*(k_i)}(k_{i+1}), \dots u_{\mathfrak{p}}^{r^*(k_i)}(k_i + \Delta k - 1), \tilde{u}_{\mathfrak{p}}^{r(k_{i+1})}(k_i + \Delta k), \dots, \tilde{u}_{\mathfrak{p}}^{r(k_{i+1})}(k_{i+1} + \Delta k - 1)\},\end{aligned} \tag{4.18}$$

where control values $u_{\mathfrak{p}}^{r^*(k_i)}(\cdot)$ are obtained by using Economic MPC, e.g., by minimizing rotated cost function, see (4.14). Similarly as earlier, $u_{\mathfrak{p}}^{r^*(k_i)}(\cdot)$ stands for plant input with node $r^*(k_i)$ being the last updated part of plant input. More precisely, the first $\Delta k - \Delta k_i - 1$ elements are from the buffers which originate from past optimizations using Economic

MPC, last one received being obtained in optimization at time instant k_i. The rest of elements come from Assumption 4.2, namely

$$\tilde{u}_{\mathfrak{p}}^{r(k_{i+1})}(k_{i+1}+j) := \kappa_{r(k_{i+1})}(\tilde{x}(k_{i+1}+j)), \tag{4.19}$$

for each $j \in \{\Delta k - \Delta k_i - 1, \ldots, \Delta k - 1\}$ where

$$\tilde{x}^+ = f(\tilde{x}, \tilde{w}_{\mathfrak{n}}) = \begin{bmatrix} f_{\mathfrak{p}}(\tilde{x}_{\mathfrak{p}}, \kappa_{r(k_{i+1})}(\tilde{x})) \\ \Xi(r(k_{i+1}), \tilde{w}_{\mathfrak{n}}) S \tilde{x}_{\mathfrak{b}} \end{bmatrix} \tag{4.20}$$

with

$$\left.\begin{aligned} \tilde{w}_{\mathfrak{n}} \in \{\tilde{w}_{\mathfrak{n}}\}_{k_i+\Delta k}^{k_{i+1}+\Delta k-1} = \{0, \ldots, 0\}, \\ \tilde{x} = \phi_f(\Delta k, x, \{w_{\mathfrak{n}}\}_{k_i}^{k_i+\Delta k-1}). \end{aligned}\right\} \tag{4.21}$$

Note that the considered control sequence is a feasible one. Again, direct application of the sequence given in (4.18) in the corresponding rotated cost function produces the following

$$\begin{aligned} &\bar{J}(\phi_f(\Delta k_i + 1, x, \{w_{\mathfrak{n}}\}_{k_i}^{k_i+\Delta k_i}), \{\tilde{u}_{\mathfrak{p}_{r(k_{i+1})}}\}_{k_{i+1}}^{k_{i+1}+\Delta k-1})) \\ &= \bar{V}(x) - \sum_{j=k_i}^{k_i+\Delta k_i} \bar{l}(\phi_{f_{\mathfrak{p}}}(j-k_i, x_{\mathfrak{p}}, \{u_{\mathfrak{p}}^{r^*}\}_{k_i}^{k_i+\Delta k_i}), u_{\mathfrak{p}}^{r^*(k_i)}(j)) \\ &\quad + \sum_{j=k_i+\Delta k}^{k_{i+1}=\Delta k-1} \left\{ \bar{g}(f_{\mathfrak{p}}(\tilde{x}(j), \tilde{u}_{\mathfrak{p}}(j))) - \bar{g}(\tilde{x}(j)) + \bar{l}(\tilde{x}(j), \tilde{u}_{\mathfrak{p}}(j)) \right\} \\ &\leq \bar{V}(x) - \sum_{j=k_i}^{k_i+\Delta k_i} \bar{l}(\phi_{f_{\mathfrak{p}}}(j-k_i, x_{\mathfrak{p}}, \{u_{\mathfrak{p}}^{r^*}\}_{k_i}^{k_i+\Delta k_i}), u_{\mathfrak{p}}^{r^*(k_i)}(j)) \end{aligned} \tag{4.22}$$

where $\{\tilde{u}_{\mathfrak{p}_{r(k_{i+1})}}\}_{k_{i+1}}^{k_{i+1}+\Delta k-1}$ is a sequence of corresponding control values for node $r(k_{i+1})$ from a feasible control sequence given in (4.18); similarly as above, again, we write $\{u_{\mathfrak{p}}^{r^*}\}_{k_i}^{k_i+\Delta k_i}$ instead of $\{u_{\mathfrak{p}}^{r^*(k_i)}\}_{k_i}^{k_i+\Delta k_i}$ to simplify notation. Due to optimality, we have

$$\begin{aligned}&\bar{V}(\phi_f(\Delta k_i+1,x,\{w_{\mathfrak{n}}\}_{k_i}^{k_i+\Delta k_i}))\\&\quad\le \bar{J}(\phi_f(\Delta k_i+1,x,\{w_{\mathfrak{n}}\}_{k_i}^{k_i+\Delta k_i}),\{\tilde{u}_{\mathfrak{p}_{r(k_{i+1})}}\}_{k_{i+1}}^{k_{i+1}+\Delta k-1}).\end{aligned} \tag{4.23}$$

Further algebraic manipulations yield

$$\begin{aligned}&\Delta\bar{V}(x(k_i))\\&\quad:=\bar{V}(\phi_f(\Delta k_i+1,x,\{w_{\mathfrak{n}}\}_{k_i}^{k_i+\Delta k_i}))-\bar{V}(x)\\&\quad\le-\sum_{j=k_i}^{k_i+\Delta k_i}\bar{l}(\phi_{f_{\mathfrak{p}}}(j-k_i,x_{\mathfrak{p}},\{u_{\mathfrak{p}}^{r^*}\}_{k_i}^{k_i+\Delta k_i}),u_{\mathfrak{p}}^{r^*(k_i)}(j))\\&\quad=-\sum_{j=k_i}^{k_i+\Delta k_i}l(\phi_{f_{\mathfrak{p}}}(j-k_i,x_{\mathfrak{p}},\{u_{\mathfrak{p}}^{r^*}\}_{k_i}^{k_i+\Delta k_i}),u_{\mathfrak{p}}^{r^*(k_i)}(j))\\&\qquad+(\Delta k_i+1)l(x_{\mathfrak{p}_s},u_{\mathfrak{p}_s})\\&\qquad+\lambda(\phi_{f_{\mathfrak{p}}}(\Delta k_i+1,x_{\mathfrak{p}},\{u_{\mathfrak{p}}\}_{k_i}^{k_i+\Delta k_i}))-\lambda(x_{\mathfrak{p}}).\end{aligned} \tag{4.24}$$

Now, summing (4.10) and (4.1) from k_i to $k_i+\Delta k_i$ results in

$$\begin{aligned}&\sum_{j=k_i}^{k_i+\Delta k_i}s(\phi_{f_{\mathfrak{p}}}(k_i-j,x_{\mathfrak{p}},\{u_{\mathfrak{p}}\}_{k_i}^{j-1}),u_{\mathfrak{p}}(j))\\&\qquad=\sum_{j=k_i}^{k_i+\Delta k_i}l(\phi_{f_{\mathfrak{p}}}(j-k_i,x_{\mathfrak{p}},\{u_{\mathfrak{p}}\}_{k_i}^{j-1}),u_{\mathfrak{p}}(j))-(\Delta k_i+1)l(x_{\mathfrak{p}_s},u_{\mathfrak{p}_s})\end{aligned} \tag{4.25}$$

and

$$\begin{aligned}&\lambda(\phi_{f_{\mathfrak{p}}}(\Delta k_i+1,x_{\mathfrak{p}},\{u_{\mathfrak{p}}\}_{k_i}^{k_i+\Delta k_i}))-\lambda(x_{\mathfrak{p}})\\&\quad\le-\sum_{j=0}^{\Delta k_i}\rho(\phi_{f_{\mathfrak{p}}}(j,x_{\mathfrak{p}},\{u_{\mathfrak{p}}\}_{k_i}^{k_i+j-1}))+\sum_{j=0}^{\Delta k_i}s(\phi_{f_{\mathfrak{p}}}(j,x_{\mathfrak{p}},\{u_{\mathfrak{p}}\}_{k_i}^{k_i+j-1}),u_{\mathfrak{p}}(k_i+j)).\end{aligned} \tag{4.26}$$

Finally, using the first lower bound from (4.11) and combining it with (4.25) and (4.26) in (4.24), provides the desired dissipation inequality (4.16).

□

Proof of Lemma 4.3. The proof of Lemma 4.3 follows the exact same lines as the proof of Lemma 4.2, with the change being the usage of the second lower bound from (4.11). □

Chapter 5

Robustness with respect to exogenous disturbances

Robustness is another control system property which provides a deeper understanding of the corresponding system. Thus, in order to fulfill our goal of deeper understanding of NCSs affected with packet dropouts and scheduling, we analyze robustness of the NCS architecture investigated in Chapter 3.

Namely, the plant is affected with exogenous disturbances $w_{\mathfrak{p}}$, see Fig. 5.1. Other parts and characteristics of the considered NCS architecture remain the same. In particular, the network is affected with packet dropouts $w_{\mathfrak{n}}$ and it allows access to only one plant input at each time instant resulting in the scheduling issue. Again, we address these issues with the same approach, that is, we carry out a protocol and controller co-design but with the focus on robustness. Recall that this protocol and controller co-design entails the exploitation of the flexible nature of NCSs which allows for adding extra devices to the architecture, i.e., buffers (distributed computation), and the usage of an MPC framework.

Robustness analysis is very important in general. For instance, real systems age and the corresponding models might not be valid over time or we cannot obtain a good model to start with. Notice that both scenarios usually translate into parameter uncertainty or variation, leading to unbounded terms in the vector fields. On the other hand, some times we simply have exogenous disturbances. Model uncertainty, within our setup, can definitely result in poor performance since the control predictions over longer horizons will not be good and due to packet dropouts we might end up using them.

As mentioned above, the NCS architecture is the same as the one in Chapter 5 besides the fact that the plant is affected with exogenous disturbances. Thus, correspondingly,

for the purpose of robustness analysis, we assume that the mapping that describes the plant dynamics possess some continuity; e.g., see Assumption 5.1. Similarly as in [30,31] and in the previous chapters, we address packet dropouts through the assumption on a finite uniform bound on the number of consecutive packet dropouts, while scheduling is addressed via an appropriate assumption on terminal control law. Furthermore, we use a relatively restrictive assumption which is the continuity of the corresponding optimal value function. This assumption is used in establishing robustness results in MPC literature for non-networked control system, e.g., see [107], and NCSs, e.g., see [31], but it is considered relatively restrictive; relaxing this assumption is definitely one of the future directions. Finally, we assume appropriate assumptions so that recursive feasibility is satisfied; explicit investigation of recursive feasibility is left for future work.

The contribution of this chapter consists of several results. First, we employ a concept of ℓ_2 stability with nonlinear gains introduced in [158] to capture robustness of the system at hand. We show that, if the number of consecutive packet dropouts is uniformly bounded and strictly smaller than the horizon in MPC, then partial nonlinear gain ℓ_2 stability can be achieved via similar assumptions used for showing stability in networked MPC [31]. We also present additional assumptions which leads to, a more traditional, linear gain ℓ_2 stability. For the one node case, we present nonlinear gain ℓ_2 stability, which represents an alternative robustness characterization of the NCS configuration considered in [31]. Finally, we also show that by exploiting stronger assumptions, which mimic those used in the latter reference, Input-to-State Stability (ISS) can be achieved in a similar way for the aggregated state of the plant and the buffer state. In this chapter, we also illustrate, via simulation, that the dynamic scheduling using the proposed method outperforms the static scheduling.

This chapter is organized as follows. In Section 5.1 we present the corresponding NCS. Note that even though the considered NCS architecture is very similar to the one considered in the previous chapter we will rewrite some equations from Chapter 3; only the ones that are necessary for the flow of presentation. Section 5.2 encapsulate all assumptions, results and simulations while Section 5.3 provides proofs.

5.1 NCS architecture

The considered NCS architecture is depicted in Fig. 5.1. As indicated in the introduction above and illustrated in the figure, the only difference between this NCS architecture and the NCS architecture considered in Chapter 3, is that the plant is affected by exogenous disturbance $w_{\mathfrak{p}}$. Thus, let us recall once again that the network is prone to packet dropouts $w_{\mathfrak{n}}$ and that it allows at each time instant access to only one plant input (R nodes - the scheduling issue). To address these issues, similarly as in Chapter 3, we carry out a protocol and controller co-design (see [30] and Chapter 3).

Due to the complexity of the considered NCS architecture, similarly as in Chapter 3, we present each part of the corresponding NCS architecture separately. However, since most of the part parts are exactly the same, we just present the final and important equations; we do this for the sake of the flow and self-containedness of the presentation.

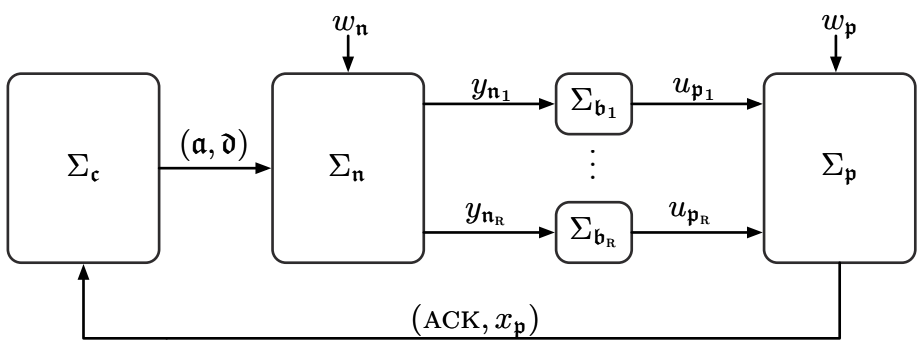


Figure 5.1: A NCS architecture for investigating robustness with respect to packet dropouts and scheduling.

5.1.1 Plant

The plant of interest is given as

$$\Sigma_{\mathfrak{p}} : x_{\mathfrak{p}}(k+1) = f_{\mathfrak{p}}(x_{\mathfrak{p}}(k), u_{\mathfrak{p}}(k), w_{\mathfrak{p}}(k)),\ k \in \mathbb{N}_0, \tag{5.1}$$

where $x_{\mathfrak{p}} \in \mathbb{X}_{\mathfrak{p}} \subseteq \mathbb{R}^{n_{\mathfrak{p}}}$ is the state, $u_{\mathfrak{p}} \in \mathbb{U}_{\mathfrak{p}} \subseteq \mathbb{R}^{m_{\mathfrak{p}}}$ is the input and $w_{\mathfrak{p}} \in \mathbb{W}_{\mathfrak{p}} \subseteq \mathbb{R}^{q_{\mathfrak{p}}}$ is the exogenous disturbance of the plant with all elements of $\{n_{\mathfrak{p}}, m_{\mathfrak{p}}, q_{\mathfrak{p}}\}$ belonging to natural

numbers. The mapping $f_{\mathfrak{p}} : \mathbb{X}_{\mathfrak{p}} \times \mathbb{U}_{\mathfrak{p}} \times \mathbb{W}_{\mathfrak{p}} \to \mathbb{X}_{\mathfrak{p}}$ is assumed to be general nonlinear. Similarly as in the previous chapter and throughout this thesis, whenever *appropriate*, we use a succinct notation for mathematical objects involving time, e.g., we write equation (5.1) (whenever *appropriate*) as

$$\Sigma_{\mathfrak{p}} : x_{\mathfrak{p}}^{+} = f_{\mathfrak{p}}(x_{\mathfrak{p}}, u_{\mathfrak{p}}, w_{\mathfrak{p}}). \tag{5.2}$$

The plant control input is *partitioned* in the same way as in the previous chapter, that is

$$u_{\mathfrak{p}} := (u_{\mathfrak{p}_1}, \ldots, u_{\mathfrak{p}_{\mathrm{R}}}) \tag{5.3}$$

with

$$\left.\begin{array}{c} u_{\mathfrak{p}_r} \in \mathbb{U}_{\mathfrak{p}_r} \subseteq \mathbb{R}^{m_{\mathfrak{p}_r}},\ \forall r \in \mathcal{R}, \\ \sum_{r=1}^{\mathrm{R}} m_{\mathfrak{p}_r} = m, \\ \mathcal{R} := \{1, \ldots, \mathrm{R}\}. \end{array}\right\} \tag{5.4}$$

5.1.2 Network

In the same manner as in the previous chapter, the transmission effects are modeled as

$$w_{\mathfrak{n}}(k) := \begin{cases} 0, & \text{if dropout occurs at the time instant } k, \\ 1, & \text{if dropout } \textit{does not} \text{ occur at the time instant } k. \end{cases} \tag{5.5}$$

The set consisting of all successful transmission time instants is given as

$$\mathbb{K} := \{k_i \in \mathbb{N}_0 : w_{\mathfrak{n}}(k_i) = 1, k_{i+1} > k_i,\ \forall i \in \mathbb{N}_0\}, \tag{5.6}$$

while the number of consecutive packet dropouts between successful transmission instants k_i and k_{i+1} is defined as

$$\Delta k_i := k_{i+1} - k_i - 1. \tag{5.7}$$

Finally, at each time instant the network will receive a *packet* π sent by the controller and defined as

$$\pi := (\mathfrak{a}, \mathfrak{d}) \in \mathcal{R} \times \mathbb{R}^{\mathrm{L} \cdot \bar{m}_{\mathfrak{p}}}, \tag{5.8}$$

where the address is denoted via $\mathfrak{a}$ and it take values from the set $\mathcal{R}$, while the data is denoted by $\mathfrak{d}$ and it take values from set $\mathbb{R}^{\mathrm{L} \cdot \bar{m}_{\mathfrak{p}}}$; recall that L is the length of a buffer and $\bar{m}_{\mathfrak{p}} = \max\{m_{\mathfrak{p}_1}, \ldots, m_{\mathfrak{p}_{\mathrm{R}}}\}$.

5.1.3 Buffer

Buffers models remain the same as in the previous chapters and the overall buffer is defined as

$$x_{\mathfrak{b}} := (x_{\mathfrak{b}_1}, \ldots, x_{\mathfrak{b}_{\mathrm{R}}}) \in \mathbb{U}_{\mathfrak{p}}^{\mathrm{L}}, \tag{5.9}$$

while its corresponding dynamics is captured via

$$\Sigma_{x_{\mathfrak{b}_r}} : \begin{cases} x_{\mathfrak{b}_r}^{+} := \begin{cases} S_{m_{\mathfrak{p}_r}} x_{\mathfrak{b}_r}, & \text{if } r \neq \mathfrak{a} \text{ or } w_{\mathfrak{n}} = 0, \\ S_{m_{\mathfrak{p}_r}} \mathfrak{d}, & \text{if } r = \mathfrak{a} \text{ and } w_{\mathfrak{n}} = 1, \end{cases} \\ y_{\mathfrak{b}_r} := \begin{cases} [I^{m_{\mathfrak{p}_r}} \; 0^{m_{\mathfrak{p}_r}} \cdots 0^{m_{\mathfrak{p}_r}}] x_{\mathfrak{b}_r}, & \text{if } r \neq \mathfrak{a} \text{ or } w_{\mathfrak{n}} = 0, \\ [I^{m_{\mathfrak{p}_r}} \; 0^{m_{\mathfrak{p}_r}} \cdots 0^{m_{\mathfrak{p}_r}}] \mathfrak{d}, & \text{if } r = \mathfrak{a} \text{ and } w_{\mathfrak{n}} = 1, \end{cases} \end{cases}, \tag{5.10}$$

Finally, recall the interconnection

$$y_{\mathfrak{b}} = u_{\mathfrak{p}}. \tag{5.11}$$

5.1.4 Controller

Note that the MPC controller uses a nominal model for making predictions, i.e., the model with no disturbances. Hence, there are no differences to the model introduced in the previous chapter. However, there are some differences during minimization of the

corresponding cost functions.

The nominal model for node $r \in \mathcal{R}$ is given as[1]

$$\Sigma_{\mathfrak{p}}^{\mathfrak{m}} : \begin{cases} \tilde{x}_{\mathfrak{p}}(k+i+1) := f_{\mathfrak{p}}(\tilde{x}_{\mathfrak{p}}(k+i), \tilde{u}_{\mathfrak{p}}^{r}(k+i)), \\ \tilde{x}_{\mathfrak{p}}(k) = x_{\mathfrak{p}}(k),\ k \in \mathbb{N}_0,\ i \in \{0, \dots, \mathfrak{h}-1\}, \end{cases} \tag{5.12}$$

where $\mathfrak{h} \in \mathbb{N}$ is a *finite horizon* and

$$\tilde{u}_{\mathfrak{p}}^{r} := \begin{bmatrix} \Gamma_{m_1}\Big(\Xi(r, \tilde{w}_{\mathfrak{n}})\tilde{x}_{\mathfrak{b}} + \Pi(\{\tilde{u}_{\mathfrak{p}_r}\}_k^{k+\mathfrak{h}-1}, r, \tilde{w}_{\mathfrak{n}})\Big) \\ \vdots \\ \Gamma_{m_{r-1}}\Big(\Xi(r, \tilde{w}_{\mathfrak{n}})\tilde{x}_{\mathfrak{b}} + \Pi(\{\tilde{u}_{\mathfrak{p}_r}\}_k^{k+\mathfrak{h}-1}, r, \tilde{w}_{\mathfrak{n}})\Big) \\ \tilde{u}_{\mathfrak{p}_r} \\ \Gamma_{m_{r+1}}\Big(\Xi(r, \tilde{w}_{\mathfrak{n}})\tilde{x}_{\mathfrak{b}} + \Pi(\{\tilde{u}_{\mathfrak{p}_r}\}_k^{k+\mathfrak{h}-1}, r, \tilde{w}_{\mathfrak{n}})\Big) \\ \vdots \\ \Gamma_{m_{\mathrm{R}}}\Big(\Xi(r, \tilde{w}_{\mathfrak{n}})\tilde{x}_{\mathfrak{b}} + \Pi(\{\tilde{u}_{\mathfrak{p}_r}\}_k^{k+\mathfrak{h}-1}, r, \tilde{w}_{\mathfrak{n}})\Big) \end{bmatrix} \in \mathbb{R}^{m_{\mathfrak{p}}}, \tag{5.13}$$

with

$$\left.\begin{array}{c} \Sigma_{\mathfrak{b}}^{\mathfrak{m}} : \tilde{x}_{\mathfrak{b}}^{+} := f_{\mathfrak{b}}(\tilde{x}_{\mathfrak{b}}, r, \{\tilde{u}_{\mathfrak{p}_r}\}_k^{k+\mathfrak{h}-1}, \tilde{w}_{\mathfrak{n}}),\ \tilde{x}_{\mathfrak{b}} = x_{\mathfrak{b}}, \\ \tilde{w}_{\mathfrak{n}} \in \{\tilde{w}_{\mathfrak{n}}\}_k^{k+\mathfrak{h}-1} := \{1, 0, \dots, 0\}, \\ \tilde{u}_{\mathfrak{p}_r} \in \{\tilde{u}_{\mathfrak{p}_r}\}_k^{k+\mathfrak{h}-1}, \\ \Gamma = \left[\Gamma_{m_{\mathfrak{p}_1}}^{\top} \cdots \Gamma_{m_{\mathfrak{p}_{\mathrm{R}}}}^{\top}\right]^{\top},\ \Gamma_{m_{\mathfrak{p}_r}} \in \mathbb{R}^{m_{\mathfrak{p}_r} \times \mathfrak{h} \cdot m_{\mathfrak{p}}}, \forall r \in \mathcal{R}. \end{array}\right\} \tag{5.14}$$

The cost function is defined as

$$\begin{aligned} & J(x, \{\tilde{u}_{\mathfrak{p}_r}\}_k^{k+\mathfrak{h}-1}) \\ & := \sum_{i=k}^{k+\mathfrak{h}-1} l(\phi_{f_{\mathfrak{p}}}(i-k, x_{\mathfrak{p}}, \{\tilde{u}_{\mathfrak{p}}^{r}\}_k^{k+\mathfrak{h}-1}), \tilde{u}_{\mathfrak{p}}^{r}(i)) + g(\phi_{f_{\mathfrak{p}}}(\mathfrak{h}, x_{\mathfrak{p}}, \{\tilde{u}_{\mathfrak{p}}^{r}\}_k^{k+\mathfrak{h}-1})), \end{aligned} \tag{5.15}$$

where

[1]Note that in the equation (5.12) we slightly abuse notation with respect to the plant model (5.1). Namely, for the nominal model (i.e., no exogenous disturbances) we write $f_{\mathfrak{p}}(\tilde{x}_{\mathfrak{p}}(k+i), \tilde{u}_{\mathfrak{p}}^{r}(k+i))$ instead of $f_{\mathfrak{p}}(\tilde{x}_{\mathfrak{p}}(k+i), \tilde{u}_{\mathfrak{p}}^{r}(k+i), 0^{q_{\mathfrak{p}}})$. We do this for the sake of shortening the corresponding equations in proof section which in turn should make the corresponding proof easier to follow.

$$x := (x_{\mathfrak{p}}, x_{\mathfrak{b}}) \in \mathbb{X} \subseteq \mathbb{X}_{\mathfrak{p}} \times \mathbb{U}_{\mathfrak{p}}^{\mathfrak{h}}. \tag{5.16}$$

Note that now $l : \mathbb{X}_{\mathfrak{p}} \times \mathbb{U}_{\mathfrak{p}} \to \mathbb{R}_{\geq 0}$, while $g : \mathbb{X}_{\mathfrak{p}_{\mathfrak{t}}} \to \mathbb{R}_{\geq 0}$, where $\mathbb{X}_{\mathfrak{p}_{\mathfrak{t}}} \subseteq \mathbb{X}_{\mathfrak{p}}$ is a set that contains the origin, e.g., $0^{n_{\mathfrak{p}}} \in \mathbb{X}_{\mathfrak{p}_{\mathfrak{t}}}$. Furthermore, we will denote the set of all feasible initial plant states as $\mathbb{X}_{\mathfrak{p}_{\mathfrak{f}}}$; note that this is a set for which the left hand side of the equation (5.15) is bounded.

For each node r, a corresponding cost function (5.15) is minimized with constraints on control predictions (5.13) and corresponding predicted plant solutions $\phi_{f_{\mathfrak{p}}}$. More precisely

$$\tilde{u}_{\mathfrak{p}}^{r}(k+i) \in \mathbb{U}_{\mathfrak{p}}, \tag{5.17}$$

for each $r \in \mathcal{R}$, each $k \in \mathbb{N}_0$ and each $i \in \{0, \ldots, \mathfrak{h}\}$. Further, the solutions of the model (5.12) are constrained with

$$\phi_{f_{\mathfrak{p}}}(i-k, x_{\mathfrak{p}}, \{\tilde{u}_{\mathfrak{p}}^{r}\}_{k}^{k+\mathfrak{h}-1}) \in \mathbb{X}_{\mathfrak{p}}, \tag{5.18}$$

for each $k \in \mathbb{N}_0$ and each $i \in \{k, \ldots, k+\mathfrak{h}-1\}$. Finally, the solution at the end of the horizon has to satisfy a *terminal state*-like constraint, namely

$$\phi_{f_{\mathfrak{p}}}(\mathfrak{h}, x_{\mathfrak{p}}, \{\tilde{u}_{\mathfrak{p}}^{r}\}_{k}^{k+\mathfrak{h}-1}) \in \mathbb{X}_{\mathfrak{p}_{\mathfrak{t}}}. \tag{5.19}$$

Now, recall that there are R models of (5.12)–(5.14), with the corresponding cost function (5.15) and constraints (5.17)–(5.19). To obtain the *optimal* node and the corresponding sequence of optimal control values over a *finite* horizon $\mathfrak{h}$, first, each cost function is minimized with respect to its $\{\tilde{u}_{\mathfrak{p}_r}\}_{k}^{k+\mathfrak{h}-1}$. More precisely,

$$V(x, \{u_{\mathfrak{p}_r}^{*}\}_{k}^{k+\mathfrak{h}-1}) := \begin{cases} \text{minimize} & J(x, \{\tilde{u}_{\mathfrak{p}_r}\}_{k}^{k+\mathfrak{h}-1}) \\ \text{subject to} & \begin{cases} (5.12)\text{–}(5.14) \\ (5.17)\text{–}(5.19). \end{cases} \end{cases} \tag{5.20}$$

Repeating this for each node results in $\{V(x, \{u_{\mathfrak{p}_r}^{*}\}_{k}^{k+\mathfrak{h}-1})\}_{r \in \mathcal{R}}$, from which we obtain the

minimum, namely

$$V(x) = V(x, \{u^*_{\mathfrak{p}_{r^*}}\}_k^{k+\mathfrak{h}-1}) := \min_r V(x, \{u^*_{\mathfrak{p}_r}\}_k^{k+\mathfrak{h}-1}), \tag{5.21}$$

which defines the optimal value function. Extracting the corresponding optimal node and its sequence of optimal control values reduces to

$$r^* := \operatorname*{argmin}_r V(x, \{u^*_{\mathfrak{p}_r}\}_k^{k+\mathfrak{h}-1}), \tag{5.22}$$

and

$$\{u^*_{\mathfrak{p}_{r^*}}\}_k^{k+\mathfrak{h}-1} := \operatorname*{argmin}_{\{\tilde{u}_{\mathfrak{p}_{r^*}}\}_k^{k+\mathfrak{h}-1}} J(x, \{\tilde{u}_{\mathfrak{p}_{r^*}}\}_k^{k+\mathfrak{h}-1}). \tag{5.23}$$

These values form the packet that controller sends to the network, namely

$$\pi = (\mathfrak{a}, \mathfrak{d}) := (r^*, (u^*_{\mathfrak{p}_{r^*}}(k_i), \ldots, u^*_{\mathfrak{p}_{r^*}}(k_i + \mathfrak{h} - 1)). \tag{5.24}$$

Finally, the closed-loop system is then given as

$$\Sigma_x : x^+ = \begin{bmatrix} f_{\mathfrak{p}}(x_{\mathfrak{p}}, h_{\mathfrak{b}}(x_{\mathfrak{b}}, r^*, \{u^*_{\mathfrak{p}_{r^*}}\}_k^{k+\mathfrak{h}-1}, w_{\mathfrak{n}}), w_{\mathfrak{p}}) \\ f_{\mathfrak{b}}(x_{\mathfrak{b}}, r^*, \{u^*_{\mathfrak{p}_{r^*}}\}_k^{k+\mathfrak{h}-1}, w_{\mathfrak{n}}) \end{bmatrix} =: f(x, w_{\mathfrak{n}}, w_{\mathfrak{p}}). \tag{5.25}$$

Note that a remark that addresses the possibility of the inclusion of a network in sensor-controller connection and consideration of a more complex NCS within the framework proposed is provided above the equation (3.38).

5.2 Robustness Analysis

Several results are established. The first one is *partial nonlinear gain ℓ_2 stability* for the plant state. This notion of robustness characterization was introduced in [158]. Then, we establish conditions for a more familiar notion of *partial linear gain ℓ_2 stability* for the plant state. This is followed by considering two special scenarios. The first one is a disturbance-free scenario, in which we recover the result from [30]. It should be noted

that we use a weaker assumption on the terminal control law to arrive at the same result. However, even though we recover (establish) a bit stronger result we have shown that by using slightly stronger assumptions we can establish a much stronger results, i.e., UGAS; see Chapter 3. The second scenario is the no scheduling scenario, in which we provide an alternative robustness characterization of NCS architecture considered in [31]. Finally, we establish sufficient conditions, which closely follow those in [31], to show ISS for the overall NCS state consisting of augmentation of plant and buffer state.

Let us proceed with the definitions of mentioned robustness characterizations, which will be followed with the exposition of the assumptions used. Main results are presented at the end of this section.

First, we provide the definition of *partial nonlinear gain ℓ_2 stability* with respect to system Σ_x (see (5.25)).

Definition 5.1 (Partial nonlinear ℓ_2 stability)**.** *Consider the system Σ_x (see (5.25)). The system Σ_x is said to be* partially nonlinear gain ℓ_2 stable *with respect to sets $\mathcal{S}^{\mathcal{D}}$ and $\mathcal{S}^{\mathbb{W}}$ if for a given finite horizon $\mathfrak{h} \in \mathbb{N}$ there exist class-$\mathcal{K}_\infty$ functions γ_1 and γ_3 and class-$\mathcal{K}$ functions γ_2 and γ_4 such that*

$$\sum_{i=0}^{k} \gamma_1(|\phi_{f_{\mathfrak{p}}}(i, x, \{w_{\mathfrak{n}}\}_0^{i-1}, \{w_{\mathfrak{p}}\}_0^{i-1})|) \leq \gamma_2(|x|) + \gamma_3\Big(\sum_{i=0}^{k-1} \gamma_4(|w_{\mathfrak{p}}(i)|)\Big), \tag{5.26}$$

holds for any $x(0) = x = (x_{\mathfrak{p}}, x_{\mathfrak{b}}) \in \mathbb{X}$, each $k \in \mathbb{N}_0$, and any subsequence $\{w_{\mathfrak{n}}\}_0^{k-1} \subset \{w_{\mathfrak{n}}\}_0^{\infty} \in \mathcal{S}^{\mathcal{D}}$ and $\{w_{\mathfrak{p}}\}_0^{k-1} \subset \{w_{\mathfrak{p}}\}_0^{\infty} \in \mathcal{S}^{\mathbb{W}}$. □

A more familiar robustness notion is stated next.

Definition 5.2 (Partial linear ℓ_2 stability)**.** *Consider the system Σ_x (see (5.25)). The system Σ_x is said to be* partially linear gain ℓ_2 stable *with respect to sets $\mathcal{S}^{\mathcal{D}}$ and $\mathcal{S}^{\mathbb{W}}$ if for a given finite horizon $\mathfrak{h} \in \mathbb{N}$ there exist quadratic functions $c_1 s^2$, $c_2 s^2$ and $c_4 s^2$ and a linear function $c_3 s$ where each element of $\{c_1, c_2, c_3, c_4\}$ is a positive real number, such that*

$$\sum_{i=0}^{k} |\phi_{f_{\mathfrak{p}}}(i, x, \{w_{\mathfrak{n}}\}_0^{i-1}, \{w_{\mathfrak{p}}\}_0^{i-1})|^2 \leq \frac{c_2}{c_1}|x|^2 + \frac{c_3 c_4}{c_1} \sum_{i=0}^{k-1} |w_{\mathfrak{p}}(i)|^2, \tag{5.27}$$

holds for any $x(0) = x = (x_{\mathfrak{p}}, x_{\mathfrak{b}}) \in \mathbb{X}$, *each* $k \in \mathbb{N}_0$, *and any subsequence* $\{w_{\mathfrak{n}}\}_0^{k-1} \subset \{w_{\mathfrak{n}}\}_0^{\infty} \in \mathcal{S}^{\mathcal{D}}$ *and* $\{w_{\mathfrak{p}}\}_0^{k-1} \subset \{w_{\mathfrak{p}}\}_0^{\infty} \in \mathcal{S}^{\mathbb{W}}$. □

Notice that in comparison to Definition 5.1 $\gamma_1(s) = c_1 s^2$, $\gamma_2(s) = c_2 s^2$,$\gamma_4(s) = c_4 s^2$ and $\gamma_3(s) = c_3 s$. Next, we provide the assumptions used for establishing the results.

5.2.1 Assumptions

The first assumption is concerned with the continuity of the map for plant state.

Assumption 5.1. *There exist class-*$\mathcal{K}$ *functions* $\alpha_{x_{\mathfrak{p}}}$ *and* $\alpha_{w_{\mathfrak{p}}}$ *such that*

$$|f_{\mathfrak{p}}(x_{\mathfrak{p}}, u_{\mathfrak{p}}, w_{\mathfrak{p}}) - f_{\mathfrak{p}}(\hat{x}_{\mathfrak{p}}, u_{\mathfrak{p}}, 0^{q_{\mathfrak{p}}})| \leq \alpha_{x_{\mathfrak{p}}}(|x_{\mathfrak{p}} - \hat{x}_{\mathfrak{p}}|) + \alpha_{w_{\mathfrak{p}}}(|w_{\mathfrak{p}}|), \tag{5.28}$$

holds for each $x \in \mathbb{X}_{\mathfrak{p}}$, $\hat{x}_{\mathfrak{p}} \in \mathbb{X}_{\mathfrak{p}}$, $u_{\mathfrak{p}} \in \mathbb{U}_{\mathfrak{p}}$ *and* $w_{\mathfrak{p}} \in \mathbb{W}_{\mathfrak{p}}$. □

Note that the stated assumption may preclude some systems with cross products, but if the sets $\mathbb{U}_{\mathfrak{p}}$ and $\mathbb{W}_{\mathfrak{p}}$ are compact, which is often the case, this issue is avoided.

The upcoming assumption is related to packet dropouts. In fact, we will use the same assumption as in the previous chapter, hence, we will just restate it for the sake of completeness.

Assumption 5.2 (Bound on the number of consecutive packet dropouts)**.** *There exists*[2] $\Delta k \in \mathbb{N}$ *such that* $\Delta k \leq \mathrm{L}$ *and* $\Delta k_i \leq \Delta k - 1$ *for each* $k_i \in \mathbb{N}_0$. □

The former assumption has the same ramifications on the notation as in the previous chapter, see equations (3.39)–(3.42).

The rest of the assumptions are MPC related. Recall from the previous chapter that it is more natural to assume that the prediction horizon in MPC is greater or equal than the *bound* on the number of consecutive packet dropouts Δk. However, similarly as in the previous chapter, for the presentation purposes, we will set it to be equal, that is

$$\mathfrak{h} := \Delta k, \tag{5.29}$$

[2]Recall that $\mathrm{L} \in \mathbb{N}$ is the length of a buffer.

for each $r \in \mathcal{R}$; for more details on this please see a remark right before the Assumption 3.2

The first MPC assumption is related to stage and terminal cost functions.

Assumption 5.3 (Semi-positive definiteness and lower bound on stage cost function)**.** *There exist a class-$\mathcal{K}_\infty$ function α such that*

$$\left.\begin{array}{c} l(x_{\mathfrak{p}}, u_{\mathfrak{p}}) \geq \alpha(|x_{\mathfrak{p}}|),\ l(0^{n_{\mathfrak{p}}}, 0^{m_{\mathfrak{p}}}) = 0,\ \forall x_{\mathfrak{p}} \in \mathbb{X}_{\mathfrak{p}_{\mathfrak{f}}},\ \forall u_{\mathfrak{p}} \in \mathbb{U}_{\mathfrak{p}}, \\ g(x_{\mathfrak{p}}) \geq 0,\ g(0^{n_{\mathfrak{p}}}) = 0,\ \forall x_{\mathfrak{p}} \in \mathbb{X}_{\mathfrak{p}_{\mathfrak{t}}}. \end{array}\right\} \tag{5.30}$$

□

Next assumption is a modification of a "standard" stability related assumption in MPC literature.

Assumption 5.4 (Terminal control law)**.** *For some node $r \in \mathcal{R}$ there exist a* terminal control law $\kappa_r : \mathbb{X}_{\mathfrak{p}_{\mathfrak{t}}} \to \tilde{\mathbb{U}}_{\mathfrak{p}_r}$ *such that*

$$\left.\begin{array}{c} g(f_{\mathfrak{p}}(x_{\mathfrak{p}}, \kappa_r(x_{\mathfrak{p}}), 0^{q_{\mathfrak{p}}})) - g(x_{\mathfrak{p}}) + l(x_{\mathfrak{p}}, \kappa_r(x_{\mathfrak{p}})) \leq 0, \\ f_{\mathfrak{p}}(x_{\mathfrak{p}}, \kappa_r(x_{\mathfrak{p}}), 0^{q_{\mathfrak{p}}}) \in \mathbb{X}_{\mathfrak{p}_{\mathfrak{t}}}, \\ \kappa_r(x_{\mathfrak{p}}) \in \tilde{\mathbb{U}}_{\mathfrak{p}_r} := 0^{m_{\mathfrak{p}_1}} \times \cdots \times 0^{m_{\mathfrak{p}_{r-1}}} \times \mathbb{U}_{\mathfrak{p}_r} \times 0^{m_{\mathfrak{p}_{r+1}}} \times \cdots \times 0^{m_{\mathfrak{p}_{\mathrm{R}}}},\ \mathbb{U}_{\mathfrak{p}_r} \subseteq \mathbb{R}^{m_{\mathfrak{p}_r}}, \end{array}\right\} \tag{5.31}$$

holds for each $x_{\mathfrak{p}} \in \mathbb{X}_{\mathfrak{p}_{\mathfrak{t}}}$. □

Note that here we assume a bit weaker assumption in comparison to Assumption 3.3. Namely, the domain of the corresponding terminal control mapping κ_r is now $\mathbb{X}_{\mathfrak{p}_{\mathfrak{t}}} \subseteq \mathbb{X}_{\mathfrak{p}} \subseteq \mathbb{R}^{n_{\mathfrak{p}}}$, whereas in Assumption 3.3 it was $\mathbb{R}^{n_{\mathfrak{p}} \times \mathfrak{h} \cdot m_{\mathfrak{p}}}$.

Further, when one investigates robustness of a systems governed by an MPC controller it is very useful to assume continuity of the optimal value function; for instance, consult [107, 122, 170] for non-networked case and [31] for networked case. We state this assumptions next.

Assumption 5.5. *There exist a class-$\mathcal{K}_\infty$ function γ_V such that*

$$|V(x_{\mathfrak{p}}, x_{\mathfrak{b}}) - V(\hat{x}_{\mathfrak{p}}, x_{\mathfrak{b}})| \leq \gamma_V(|x_{\mathfrak{p}} - \hat{x}_{\mathfrak{p}}|), \tag{5.32}$$

holds for each $(x_{\mathfrak{p}}, x_{\mathfrak{b}}) \in \mathbb{X}_{\mathfrak{p}_{\mathfrak{f}}} \times \mathbb{U}_{\mathfrak{p}}^{\mathfrak{h}}$ *and* $(\hat{x}_{\mathfrak{p}}, x_{\mathfrak{b}}) \in \mathbb{X}_{\mathfrak{p}_{\mathfrak{f}}} \times \mathbb{U}_{\mathfrak{p}}^{\mathfrak{h}}$. □

Note that the former assumption concentrates only on plant states. Moreover, note that if the set $\mathbb{X}_{\mathfrak{p}_{\mathfrak{f}}}$ is bounded, then the corresponding assumption is not strong. However, if this is not the case, then the general conditions that would relax inequality (5.32) are still an open question.

Next, similarly as in the previous chapter, we assume that we are able to bound the optimal value function with a class-$\mathcal{K}$ function.

Assumption 5.6 (Class-$\mathcal{K}$ bound on the optimal value function)**.** *There exist a class-$\mathcal{K}$ function $\gamma_{\bar{V}}$ such that*

$$V(x) \leq \gamma_{\bar{V}}(|x|), \tag{5.33}$$

holds for each $x = (x_{\mathfrak{p}}, x_{\mathfrak{b}}) \in \mathbb{X}_{\mathfrak{p}_{\mathfrak{f}}} \times \mathbb{U}_{\mathfrak{p}}^{\mathfrak{h}}$. □

Note that the former assumption can be seen as a consequence of the corresponding NCS system being asymptotically controllable to the origin; we remind the reader to consult, for instance, Section III in [123] and/or Section II in [160].

Further, note that before the first successful transmission the plant will be governed with the (initial) values in buffers in an open-loop fashion. Thus, we will assume that the corresponding plant trajectories will be bounded during this period.

Assumption 5.7 (Bound on plant trajectories until first successful transmission)**.** *Consider the system Σ_x, given by the equation (5.25). There exist class-$\mathcal{K}$ functions α_x and $\alpha_{\bar{w}_{\mathfrak{p}}}$ such that*

$$|f(x, 0, w_{\mathfrak{p}})| \leq \alpha_x(|x|) + \alpha_{\bar{w}_{\mathfrak{p}}}(|w_{\mathfrak{p}}|), \tag{5.34}$$

holds for each $x \in \mathbb{X}_{\mathfrak{p}} \times \mathbb{U}_{\mathfrak{p}}^{\mathfrak{h}}$ *and* $w_{\mathfrak{p}} \in \mathbb{W}_{\mathfrak{p}}$. □

The final assumption is related to recursive feasibility. Namely, we will assume that the set $\mathbb{X}_{\mathfrak{p}_{\mathfrak{f}}} \times \mathbb{U}_{\mathfrak{p}}^{\mathfrak{h}}$ is positively invariant.

Assumption 5.8 (Positive invariance (recursive feasibility))**.** *Consider the system Σ_x, given by the equation (5.25). For any sequence $\{w_{\mathfrak{p}}\}_0^{\infty} \in \mathcal{S}^{\mathbb{W}_{\mathfrak{p}}}$ the solutions of the system Σ_x satisfy*

$$\phi_f(i, x, 0_0^{\mathfrak{h}-1}, \{w_{\mathfrak{p}}\}_0^{\mathfrak{h}-1}) \in \mathbb{X}_{\mathfrak{p}_{\mathfrak{f}}} \times \mathbb{U}_{\mathfrak{p}}^{\mathfrak{h}} \subseteq \mathbb{X}_{\mathfrak{p}}, \tag{5.35}$$

for each $x = x(0) \in \mathbb{X}_{\mathfrak{p}}$*, each* $\{w_{\mathfrak{p}}\}_0^{\mathfrak{h}-1} \subset \{w_{\mathfrak{p}}\}_0^{\infty} \in \mathcal{S}^{\mathbb{W}_{\mathfrak{p}}}$ *and each* $i \in \{0, \ldots, \mathfrak{h}\}$*. Moreover, for the sequence of dropout outcomes* $\{1, 0, \ldots, 0\}$

$$\phi_f(i-k, x, \{1, 0, \ldots, 0\}, \{w_{\mathfrak{p}}\}_k^{k+\mathfrak{h}-1}) \in \mathbb{X}_{\mathfrak{p}_{\mathfrak{f}}} \times \mathbb{U}_{\mathfrak{p}}^{\mathfrak{h}} \subseteq \mathbb{X}_{\mathfrak{p}}, \tag{5.36}$$

holds for each $x = x(k) \in \mathbb{X}_{\mathfrak{p}_{\mathfrak{f}}} \times \mathbb{U}_{\mathfrak{p}}^{\mathfrak{h}}$*, each* $\{w_{\mathfrak{p}}\}_k^{k+\mathfrak{h}-1} \subset \{w_{\mathfrak{p}}\}_0^{\infty} \in \mathcal{S}^{\mathbb{W}_{\mathfrak{p}}}$ *and each* $i \in \{k, \ldots, k+\mathfrak{h}\}$*,* $k \in \mathbb{N}$*.* □

Remark: Note that we do not deal with the issue of recursive feasibility directly. This is left for future work. For the purposes of the present result(s) we use Assumption 5.8. Note that the first part of this assumption ensures that starting from any initial state $x(0) = x$ the trajectories will end up in the set $\mathbb{X}_{\mathfrak{p}_{\mathfrak{f}}} \times \mathbb{U}_{\mathfrak{p}}^{\mathfrak{h}}$. On the other hand, the second part mimics Assumption 3 from [31]. More precisely, it states that the set $\mathbb{X}_{\mathfrak{p}_{\mathfrak{f}}} \times \mathbb{U}_{\mathfrak{p}}^{\mathfrak{h}}$ is robust positively invariant for the mapping of system Σ_x (see (5.25)) for up to $\mathfrak{h}$ steps. □

Finally, we proceed with results obtained.

5.2.2 Results

Chronologically, we first present partial nonlinear gain ℓ_2 stability for plant states only. This result uses the concept of ℓ_2 stability with nonlinear gains; e.g., see [158]. The underpinning requirement is that the number of consecutive packet dropouts is finitely uniformly bounded from above. Then, using similar assumptions as the ones used for showing stability in networked MPC [31] does the rest of the work. This result is followed with the corresponding modifications so that a more traditional, linear gain ℓ_2 stability can be established. Then, we consider two scenarios. In the first one we omit disturbances on the plant and recover the main result from [30] with a weaker assumption on the terminal control law, making our recovery a bit stronger result. In the second scenario we omit scheduling, i.e., we consider only one node. There, we establish nonlinear gain ℓ_2 stability which can be seen as an alternative robustness characterization of

the NCS configuration considered in [31]. Finally, we conclude by exploiting stronger assumptions to establish ISS of the overall NCS state (i.e., the aggregated state of the plant and buffer state).

Partial nonlinear gain ℓ_2 stability

As mentioned above, we exploit the concept of ℓ_2 stability with nonlinear gains to state the upcoming theorem; see [158] for more details on this concept. Moreover, we use the ideas from stability results for networked MPC to state the sufficient assumptions; see, for instance, [31]. Finally, note that the enabling requirement is the uniform bound on the number of consecutive packet dropouts.

Theorem 5.1 (Partial nonlinear gain ℓ_2 stability)**.** *Consider the system Σ_x, given by the equation (5.25). Let the Assumptions 5.1 - 5.8 be satisfied. Then for a finite horizon Δk the system Σ_x is partially nonlinear gain ℓ_2 stable with respect to sets $\mathcal{S}^{\mathcal{D}}_{\mathbb{K}_{\Delta k}}$ and $\mathcal{S}^{\mathbb{W}_{\mathfrak{p}}}$.* □

To prove this result we establish four lemmas which are stated next. The first lemma uses Assumptions 5.2 and 5.8 to establish the invariance of the set $\mathbb{X}_{\mathfrak{p}_{\mathfrak{f}}} \times \mathbb{U}^{\mathfrak{h}}_{\mathfrak{p}}$.

Lemma 5.1 (Invariance of the set $\mathbb{X}_{\mathfrak{p}_{\mathfrak{f}}} \times \mathbb{U}^{\mathfrak{h}}_{\mathfrak{p}}$)**.** *Consider any $\{w_{\mathfrak{n}}\}_0^\infty \in \mathcal{S}^{\mathcal{D}}_{\mathbb{K}_{\Delta k}}$ and $\{w_{\mathfrak{p}}\}_0^\infty \in \mathcal{S}^{\mathbb{W}_{\mathfrak{p}}}$. Let the Assumptions 5.2 and 5.8 be satisfied. Then for any $x(0) \in \mathbb{X}$ it follows that $x(k_i) \in \mathbb{X}_{\mathfrak{p}_{\mathfrak{f}}} \times \mathbb{U}^{\mathfrak{h}}_{\mathfrak{p}}$ for each $k_i \in \mathbb{K}_{\Delta k}$.* □

The second lemma establishes the difference of the optimal value function between the two successful time instances. This lemma in particular is instrumental in the proof of Theorem 5.1.

Lemma 5.2 (Dissipation inequality)**.** *Consider any $\{w_{\mathfrak{n}}\}_0^\infty \in \mathcal{S}^{\mathcal{D}}_{\mathbb{K}_{\Delta k}}$ and $\{w_{\mathfrak{p}}\}_0^\infty \in \mathcal{S}^{\mathbb{W}_{\mathfrak{p}}}$. Let the Assumptions 5.1–5.6 and 5.8 be satisfied. Then, there exist a class-$\mathcal{K}_\infty$ function α_1 and a class-$\mathcal{K}$ function α_2 such that*

$$\begin{aligned} &V(x(k_{i+1})) - V(x(k_i)) \\ &\quad \leq -\sum_{j=k_i}^{k_{i+1}-1} \alpha_1(|\phi_{f_{\mathfrak{p}}}(j-k_i, x, \{w_{\mathfrak{n}}\}_{k_i}^{j-1}, \{w_{\mathfrak{p}}\}_{k_i}^{j-1})|) + \sum_{j=k_i}^{k_{i+1}-1} \alpha_2(|w_{\mathfrak{p}}(j)|), \end{aligned} \tag{5.37}$$

holds for each $x(k_i) \in \mathbb{X}_{\mathfrak{p}_\mathfrak{f}} \times \mathbb{U}_\mathfrak{p}^\mathfrak{h}$, $x(k_{i+1}) \in \mathbb{X}_{\mathfrak{p}_\mathfrak{f}} \times \mathbb{U}_\mathfrak{p}^\mathfrak{h}$, $k_i \in \mathbb{K}_{\Delta k}$ *and* $k_{i+1} \in \mathbb{K}_{\Delta k}$. □

The following two lemmas concentrate on two time intervals and establish the corresponding desired inequalities. The first one considers time interval $\{k_0, \ldots, k\}$.

Lemma 5.3 (Inequality for time interval $\{k_0, \ldots, k\}$)**.** *Let the assumptions of Lemma 5.2 be satisfied. Then for the class-*$\mathcal{K}_\infty$ *function* α_1 *(from Lemma 5.2) and the class-*$\mathcal{K}$ *function* $\gamma_{\bar{V}}$ *(from Assumption 5.6) there exist a class-*$\mathcal{K}$ *function* α_3 *such that*

$$\sum_{j=k_0}^{k-1} \alpha_1(|\phi_{f_\mathfrak{p}}(j-k_0, x, \{w_\mathfrak{n}\}_{k_i}^{j-1}, \{w_\mathfrak{p}\}_{k_i}^{j-1})|) \leq \gamma_{\bar{V}}(|x(k_0)|) + \sum_{j=k_0}^{k-1} \alpha_3(|w_\mathfrak{p}(j)|) \tag{5.38}$$

holds for any $x(k_0) \in \mathbb{X}_{\mathfrak{p}_\mathfrak{f}} \times \mathbb{U}_\mathfrak{p}^\mathfrak{h}$. □

The second one considers the time interval $\{0, \ldots, k_0\}$.

Lemma 5.4 (Inequality for time interval $\{0, \ldots, k_0\}$)**.** *Let the Assumptions 5.2 and 5.7 be satisfied. Then for a class-*$\mathcal{K}$ *function* $\gamma_{\bar{V}}$ *there exist a class-*$\mathcal{K}$ *functions* α_4 *and* α_5 *such that*

$$\gamma_{\bar{V}}(|x(k_0)|) \leq \alpha_4(|x(0)|) + \sum_{j=0}^{k_0-1} \alpha_5(|w_\mathfrak{p}(j)|) - \sum_{j=0}^{k_0-1} \gamma_{\bar{V}}(|\phi_{f_\mathfrak{p}}(j, x, \{w_\mathfrak{n}\}_0^{j-1}, \{w_\mathfrak{p}\}_0^{j-1})|), \tag{5.39}$$

holds for any $x(0) \in \mathbb{X}$ *where class-*$\mathcal{K}$ *function* $\gamma_{\bar{V}}$ *is from Assumption 5.6.* □

We now proceed with applying this result to special cases.

Partial linear gain ℓ_2 stability

The first special case is the case with a linear system and a quadratic cost; note that this scenario is very common in practice and theory. Considering this case yields the result that captures a more familiar notion of partial linear gain ℓ_2 stability. We state this result next.

Corollary 5.1 (Partial linear gain ℓ_2 stability)**.** *Consider the system* Σ_x*, given by the equation (5.25). Let the following conditions be satisfied:*

- *The cost function given in the equation (5.15) is quadratic,*
- *Assumption 5.1 with sets* $\mathbb{X}_{\mathfrak{p}}$, $\mathbb{U}_{\mathfrak{p}}$ *and* $\mathbb{W}_{\mathfrak{p}}$ *being compact, and with* $\alpha_{x_{\mathfrak{p}}}(s) := c_{x_{\mathfrak{p}}}s,\ c_{x_{\mathfrak{p}}} > 0$ *and* $\alpha_{w_{\mathfrak{p}}}(s) := c_{w_{\mathfrak{p}}}s,\ c_{w_{\mathfrak{p}}} > 0$,
- *Assumption 5.2,*
- *Assumption 5.3,*
- *Assumption 5.4,*
- *Assumption 5.5 with* $\gamma_V(s) := c_V s^2, c_V > 0$,
- *Assumption 5.6 with* $\gamma_{\bar{V}}(s) := c_{\bar{V}}s^2,\ c_{\bar{V}} > 0$,
- *Assumption 5.7 with* $\alpha_x(s) := c_x s,\ c_x > 0$ *and* $\alpha_{\bar{w}_{\mathfrak{p}}}(s) := c_{\bar{w}_{\mathfrak{p}}}s,\ c_{\bar{w}_{\mathfrak{p}}} > 0$,
- *Assumption 5.8.*

Then, for a finite horizon $\mathfrak{h}$ *from Assumption 5.2 the system* Σ_x *is partially linear gain* ℓ_2 *stable with respect to sets* $\mathcal{S}^{\mathcal{D}}_{\mathbb{K}_{\Delta k}}$ *and* $\mathcal{S}^{\mathbb{W}}$*; with the constant gains that define functions* $\gamma_i(\cdot), i \in \{1,2,3,4\}$ *in Definition 5.2 given as:*

- $c_1 := \min\{c_{\bar{V}}, \frac{1}{2}\}$,
- $c_2 := (\Delta k + 1)c_{\bar{V}}(2c_x)^{2(\Delta k-1)}c_x^2$,
- $c_2 := 1$,
- $c_4 := 2^{4(\Delta k-1)} \max\{c_{x_{\mathfrak{p}}}^{2(\Delta k-1)}c_{w_{\mathfrak{p}}}^2 \max\{\frac{\Delta k}{2}, c_V\}, \Delta k c_{\bar{V}} c_x^{2(\Delta k-1)} c_{\bar{w}_{\mathfrak{p}}}^2\}$.

□

Notice that we assume that sets $\mathbb{X}_{\mathfrak{p}}$, $\mathbb{U}_{\mathfrak{p}}$ and $\mathbb{W}_{\mathfrak{p}}$ are compact. This is done for their bound property which enables us to assume $\gamma_V(s) := c_V s^2, c_V > 0$ in Assumption 5.5.

Disturbance-free scenario

The second case is the case with no disturbances. Note that the corresponding NCS architecture becomes essentially the same as the one considered in [30] (and in Chapter 3). Thus, our next result can be seen as a strengthened main result from [30].

Corollary 5.2 (No disturbance - plant state convergence)**.** *Consider the system Σ_x, given by the equation (5.25) and let it be partially nonlinear gain ℓ_2 stable according to Definition 5.1 and let $\mathbb{W}_{\mathfrak{p}} := \{0\}$. Then*

$$\lim_{k\to\infty} |\phi_{f_{\mathfrak{p}}}(k, x, \{w_{\mathfrak{n}}\}_0^{k-1}, \{0, \ldots, 0\})| = 0 \tag{5.40}$$

for any $(x_{\mathfrak{p}}, x_{\mathfrak{b}}) \in \mathbb{X}_{\mathfrak{p}} \times \mathbb{U}_{\mathfrak{p}}^{\mathfrak{h}}$, $\{w_{\mathfrak{n}}\}_0^{k-1} \in \mathcal{S}^{\mathcal{D}}_{\mathbb{K}_{\Delta k}}$ and $\{0, \ldots, 0\} \in \mathcal{S}^{\mathbb{W}_{\mathfrak{p}}}$. □

Remark: It is worth emphasizing that the authors in [30] used a stronger assumption to come to the same conclusion. Namely, Assumption 5.4 was required to hold for all nodes. More precisely, the corresponding terminal control law was required to exist for each node while in our case we require it to exist only for some node. □

Remark: Note that the assumptions used in Chapter 5 were insufficient to establish the result from Chapter 3, i.e., UGAS. Namely, the enabling assumption was the additional lower bound on the stage cost function in terms of control only, e.g., see Assumption 3.2; see also [160]. □

One node scenario

In this section we consider a one node case. Note that this particular NCS architecture has been investigated to some extent in [31], where ISS was shown. However, partial nonlinear gain ℓ_2 stability was not considered. In the sequel we show that, with assumptions akin to those in [31], plus an extra assumption to explicitly consider the time interval up until the first successful transmission we can show partial nonlinear gain ℓ_2 stability.

Let us begin by providing the changes to the NCS architecture. The nominal model is now given as[3]

[3]Similarly as for the nominal model (5.12), note that in the equation (5.41) we slightly abuse notation with

$$\Sigma_{\mathfrak{p}}^{\mathfrak{m}} : \begin{cases} \tilde{x}_{\mathfrak{p}}(k+i+1) := f_{\mathfrak{p}}(\tilde{x}_{\mathfrak{p}}(k+i), \tilde{u}_{\mathfrak{p}}(k+i)), \\ \tilde{x}_{\mathfrak{p}}(k) := x_{\mathfrak{p}}(k), \ \forall k \in \mathbb{N}_0, \ \forall i \in \{0, \dots, \mathfrak{h}-1\} \end{cases} \tag{5.41}$$

where $\mathfrak{h} \in \mathbb{N}$ is a finite horizon. The cost function becomes

$$\begin{aligned} & J(x_{\mathfrak{p}}, \{\tilde{u}_{\mathfrak{p}}\}_k^{k+\mathfrak{h}-1}) \\ & := \sum_{i=k}^{k+\mathfrak{h}-1} l(\phi_{f_{\mathfrak{p}}}(i-k, x_{\mathfrak{p}}, \{\tilde{u}_{\mathfrak{p}}\}_k^{k+\mathfrak{h}-1}), \tilde{u}_{\mathfrak{p}}(i)) + g(\phi_{f_{\mathfrak{p}}}(\mathfrak{h}, x_{\mathfrak{p}}, \{\tilde{u}_{\mathfrak{p}}\}_k^{k+\mathfrak{h}-1})) \end{aligned} \tag{5.42}$$

and is minimized with respect to the following constraints. The control is constraint as

$$\tilde{u}_{\mathfrak{p}}(k+i) \in \mathbb{U}_{\mathfrak{p}}, \ \forall k \in \mathbb{N}_0, \tag{5.43}$$

for each $i \in \{0, \dots, \mathfrak{h}\}$. Further, the solutions of the model (5.41) are constrained with

$$\phi_{f_{\mathfrak{p}}}(i-k, x_{\mathfrak{p}}, \{\tilde{u}_{\mathfrak{p}}\}_k^{k+\mathfrak{h}-1}) \in \mathbb{X}_{\mathfrak{p}}, \tag{5.44}$$

for each $k \in \mathbb{N}_0$ and each $i \in \{k, \dots, k+\mathfrak{h}-1\}$. Finally, the solution at the end of horizon has to satisfy *terminal state*-like constraint, namely

$$\phi_{f_{\mathfrak{p}}}(\mathfrak{h}, x_{\mathfrak{p}}, \{\tilde{u}_{\mathfrak{p}}\}_k^{k+\mathfrak{h}-1}) \in \mathbb{X}_{\mathfrak{p}_t}. \tag{5.45}$$

Then the optimal value function is obtained as

$$V(x_{\mathfrak{p}}) := V(x_{\mathfrak{p}}, \{u_{\mathfrak{p}}^*\}_k^{k+\mathfrak{h}-1}) := \begin{cases} \text{minimize} & J(x_{\mathfrak{p}}, \{\tilde{u}_{\mathfrak{p}}\}_k^{k+\mathfrak{h}-1}) \\ \text{subject to} & \begin{cases} (5.41), \\ (5.43)\text{–}(5.45), \end{cases} \end{cases} \tag{5.46}$$

and the sequence of optimally predicted control values over a finite horizon is extracted as

respect to the plant model (5.1). Namely, for the nominal model (i.e., no exogenous disturbances) we write $f_{\mathfrak{p}}(\tilde{x}_{\mathfrak{p}}(k+i), \tilde{u}_{\mathfrak{p}}(k+i))$ instead of $f_{\mathfrak{p}}(\tilde{x}_{\mathfrak{p}}(k+i), \tilde{u}_{\mathfrak{p}}(k+i), 0^{q_{\mathfrak{p}}})$.

$$\{u_{\mathfrak{p}}^*\}_k^{k+\mathfrak{h}-1} := \underset{\{\tilde{u}_{\mathfrak{p}}\}_k^{k+\mathfrak{h}-1}}{\operatorname{argmin}} J(x_{\mathfrak{p}}, \{\tilde{u}_{\mathfrak{p}}\}_k^{k+\mathfrak{h}-1}). \tag{5.47}$$

Also, note that for $\mathrm{R} = 1$ buffer dynamics becomes

$$\Sigma_{\mathfrak{b}} : \begin{cases} x_{\mathfrak{b}}^+ := f_{\mathfrak{b}}(x_{\mathfrak{b}}, \mathfrak{d}, w_{\mathfrak{n}}) := (1 - w_{\mathfrak{n}})Sx_{\mathfrak{b}} + w_{\mathfrak{n}}\mathfrak{d}, \\ y_{\mathfrak{b}} := h_{\mathfrak{b}}(x_{\mathfrak{b}}, \mathfrak{d}, w_{\mathfrak{n}}) := \Gamma\Big((1 - w_{\mathfrak{n}})x_{\mathfrak{b}} + w_{\mathfrak{n}}\mathfrak{d}\Big). \end{cases} \tag{5.48}$$

Finally, the closed-loop system is now given as

$$\Sigma_{x_{\mathfrak{p}}} : x_{\mathfrak{p}}^+ = f_{\mathfrak{p}}(x_{\mathfrak{p}}, \Gamma f_{\mathfrak{b}}(x_{\mathfrak{b}}, \{u_{\mathfrak{p}}^*\}_k^{k+\mathfrak{h}-1}, w_{\mathfrak{n}}), w_{\mathfrak{p}}) := \bar{f}(x, w_{\mathfrak{n}}, w_{\mathfrak{p}}). \tag{5.49}$$

Having presented all the corresponding changes, we proceed to state the result.

Corollary 5.3 (One-node scenario)**.** *Consider the system $\Sigma_{x_{\mathfrak{p}}}$, given by the equation (5.49). Let the following be satisfied:*

- *Assumption 5.1,*
- *Assumption 5.2,*
- *Assumption 5.3,*
- *There exist a* terminal control law $\kappa : \mathbb{X}_{\mathfrak{p}_{\mathfrak{t}}} \to \mathbb{U}_{\mathfrak{p}}$ *such that*

$$\left.\begin{aligned} g(f_{\mathfrak{p}}(x_{\mathfrak{p}}, \kappa(x_{\mathfrak{p}}), 0^{q_{\mathfrak{p}}})) - g(x_{\mathfrak{p}}) + l(x_{\mathfrak{p}}, \kappa(x_{\mathfrak{p}})) \leq 0, \\ f_{\mathfrak{p}}(x_{\mathfrak{p}}, \kappa(x_{\mathfrak{p}}), 0^{q_{\mathfrak{p}}}) \in \mathbb{X}_{\mathfrak{p}_{\mathfrak{t}}}, \\ \kappa(x_{\mathfrak{p}}) \in \mathbb{U}_{\mathfrak{p}}, \end{aligned}\right\} \tag{5.50}$$

  holds for each $x_{\mathfrak{p}} \in \mathbb{X}_{\mathfrak{p}_{\mathfrak{t}}}$,

- *There exist a class-$\mathcal{K}_\infty$ function γ_V such that*

$$|V(x_{\mathfrak{p}}) - V(\hat{x}_{\mathfrak{p}})| \leq \gamma_V(|x_{\mathfrak{p}} - \hat{x}_{\mathfrak{p}}|), \tag{5.51}$$

  hold for any $x_{\mathfrak{p}} \in \mathbb{X}_{\mathfrak{p}_{\mathfrak{f}}}$ and $\hat{x}_{\mathfrak{p}} \in \mathbb{X}_{\mathfrak{p}_{\mathfrak{f}}}$,

- *Assumption 5.7,*
- *Assumption 5.8.*

Then, for a finite horizon $\mathfrak{h}$ *from Assumption 5.2 the system* $\Sigma_{x_\mathfrak{p}}$ *is partially nonlinear gain* ℓ_2 *stable with respect to sets* $\mathcal{S}^{\mathcal{D}}_{\mathbb{K}_{\Delta k}}$ *and* $\mathcal{S}^{\mathbb{W}_\mathfrak{p}}$. □

Input-to-State Stability

We finish with ISS of the augmented state of the plant and the buffer state. Again, we begin by providing the changes to the NCS architecture. Note that, similarly as for the case of one node, some of the equations are repetitions of the equations stated earlier.

The nominal model becomes[4]

$$\Sigma^{\mathfrak{m}}_{\mathfrak{p}} : \begin{cases} \begin{bmatrix} \tilde{x}_\mathfrak{p}(k+i+1) \\ \tilde{x}_\mathfrak{b}(k+i+1) \end{bmatrix} := \begin{bmatrix} f_\mathfrak{p}(\tilde{x}_\mathfrak{p}(k+i), \tilde{u}^r_\mathfrak{p}(k+i)) \\ f_\mathfrak{b}(\tilde{x}_\mathfrak{b}(k+i), r, \{\tilde{u}_{\mathfrak{p}_r}\}_k^{k+\mathfrak{h}-1}, \tilde{w}_\mathfrak{n}(k+i)) \end{bmatrix}, \\ \begin{bmatrix} \tilde{x}_\mathfrak{p}(k) \\ \tilde{x}_\mathfrak{b}(k) \end{bmatrix} = \begin{bmatrix} x_\mathfrak{p}(k) \\ x_\mathfrak{b}(k) \end{bmatrix}, \ k \in \mathbb{N}_0, \ i \in \{0, \dots, \mathfrak{h}-1\}, \end{cases} \tag{5.52}$$

where $\mathfrak{h} \in \mathbb{N}$ is a *finite horizon* and

$$\tilde{u}^r_\mathfrak{p} := \begin{bmatrix} \Gamma_{m_1}\Big(\Xi(r, \tilde{w}_\mathfrak{n})\tilde{x}_\mathfrak{b} + \Pi(\{\tilde{u}_{\mathfrak{p}_r}\}_k^{k+\mathfrak{h}-1}, r, \tilde{w}_\mathfrak{n})\Big) \\ \vdots \\ \Gamma_{m_{r-1}}\Big(\Xi(r, \tilde{w}_\mathfrak{n})\tilde{x}_\mathfrak{b} + \Pi(\{\tilde{u}_{\mathfrak{p}_r}\}_k^{k+\mathfrak{h}-1}, r, \tilde{w}_\mathfrak{n})\Big) \\ \tilde{u}_{\mathfrak{p}_r} \\ \Gamma_{m_{r+1}}\Big(\Xi(r, \tilde{w}_\mathfrak{n})\tilde{x}_\mathfrak{b} + \Pi(\{\tilde{u}_{\mathfrak{p}_r}\}_k^{k+\mathfrak{h}-1}, r, \tilde{w}_\mathfrak{n})\Big) \\ \vdots \\ \Gamma_{m_\mathbb{R}}\Big(\Xi(r, \tilde{w}_\mathfrak{n})\tilde{x}_\mathfrak{b} + \Pi(\{\tilde{u}_{\mathfrak{p}_r}\}_k^{k+\mathfrak{h}-1}, r, \tilde{w}_\mathfrak{n})\Big) \end{bmatrix} \in \mathbb{R}^{m_\mathfrak{p}}, \tag{5.53}$$

[4]Please notice once again that similarly as for the nominal model (5.12) and the nominal model (5.41), in the equation (5.52), we slightly abuse notation with respect to the plant model (5.1). Namely, again, for the nominal model (i.e., no exogenous disturbances) we write $f_\mathfrak{p}(\tilde{x}_\mathfrak{p}(k+i), \tilde{u}^r_\mathfrak{p}(k+i))$ instead of $f_\mathfrak{p}(\tilde{x}_\mathfrak{p}(k+i), \tilde{u}^r_\mathfrak{p}(k+i), 0^{q_\mathfrak{p}})$.

with

$$\left.\begin{aligned} \tilde{w}_{\mathfrak{n}} \in \{\tilde{w}_{\mathfrak{n}}\}_k^{k+\mathfrak{h}-1} &:= \{1, 0, \ldots, 0\}, \\ \tilde{u}_{\mathfrak{p}_r} \in \{\tilde{u}_{\mathfrak{p}_r}\}_k^{k+\mathfrak{h}-1}&, \\ \Gamma = \left[\Gamma_{m_{\mathfrak{p}_1}}^\top \cdots \Gamma_{m_{\mathfrak{p}_R}}^\top\right]^\top, \Gamma_{m_{\mathfrak{p}_r}} &\in \mathbb{R}^{m_{\mathfrak{p}_r} \times \mathfrak{h} \cdot m_{\mathfrak{p}}}, \forall r \in \mathcal{R}. \end{aligned}\right\} \tag{5.54}$$

The cost function is defined as

$$\begin{aligned} &J(x, \{\tilde{u}_{\mathfrak{p}_r}\}_k^{k+\mathfrak{h}-1}) \\ &:= \sum_{i=k}^{k+\mathfrak{h}-1} l(\phi_f(i-k, x, \{\tilde{u}_{\mathfrak{p}}^r\}_k^{k+\mathfrak{h}-1}), \tilde{u}_{\mathfrak{p}}^r(i)) + g(\phi_f(\mathfrak{h}, x, \{\tilde{u}_{\mathfrak{p}}^r\}_k^{k+\mathfrak{h}-1})) \end{aligned} \tag{5.55}$$

where

$$x := (x_{\mathfrak{p}}, x_{\mathfrak{b}}) \in \mathbb{X} \subseteq \mathbb{X}_{\mathfrak{p}} \times \mathbb{U}_{\mathfrak{p}}^{\mathfrak{h}}. \tag{5.56}$$

The cost function is minimized with respect to the following constraints. Control is constrained as

$$\tilde{u}_{\mathfrak{p}}^r(k+i) \in \mathbb{U}_{\mathfrak{p}},\ \forall r \in \mathcal{R},\ \forall k \in \mathbb{N}_0,\ \forall i \in \{0, \ldots, \mathfrak{h}\}. \tag{5.57}$$

The solutions of the model (5.52) are constrained with

$$\phi_f(i-k, x, \{\tilde{u}_{\mathfrak{p}}^r\}_k^{k+\mathfrak{h}-1}) \in \mathbb{X},\ \forall k \in \mathbb{N}_0,\ \forall i \in \{k, \ldots, k+\mathfrak{h}-1\}. \tag{5.58}$$

Finally, the solution at the end of horizon has to satisfy *terminal state*-like constraint, namely

$$\phi_f(\mathfrak{h}, x, \{\tilde{u}_{\mathfrak{p}}^r\}_k^{k+\mathfrak{h}-1}) \in \mathbb{X}_{\mathfrak{p}_t} \times \mathbb{U}_{\mathfrak{p}}^{\mathfrak{h}}. \tag{5.59}$$

The optimal value function for each node is defined as

$$V(x, \{u^*_{\mathfrak{p}_r}\}_k^{k+\mathfrak{h}-1}) := \begin{cases} \text{minimize} & J(x, \{\tilde{u}_{\mathfrak{p}_r}\}_k^{k+\mathfrak{h}-1}) \\ \text{subject to} & \begin{cases} (5.52)\text{–}(5.54) \\ (5.57)\text{–}(5.59). \end{cases} \end{cases} \tag{5.60}$$

Repeating this procedure for each node results in $\{V(x, \{u^*_{\mathfrak{p}_r}\}_k^{k+\mathfrak{h}-1})\}_{r\in\mathcal{R}}$ from which we take the minimum, yielding the optimal value function

$$V(x) = V(x, \{u^*_{\mathfrak{p}_{r^*}}\}_k^{k+\mathfrak{h}-1}) := \min_r V(x, \{u^*_{\mathfrak{p}_r}\}_k^{k+\mathfrak{h}-1}). \tag{5.61}$$

Extracting the corresponding optimal node and its sequence of optimal control values reduces to

$$r^* := \operatorname*{argmin}_r V(x, \{u^*_{\mathfrak{p}_r}\}_k^{k+\mathfrak{h}-1}), \tag{5.62}$$

and

$$\{u^*_{\mathfrak{p}_{r^*}}\}_k^{k+\mathfrak{h}-1} := \operatorname*{argmin}_{\{\tilde{u}_{\mathfrak{p}_{r^*}}\}_k^{k+\mathfrak{h}-1}} J(x, \{\tilde{u}_{\mathfrak{p}_{r^*}}\}_k^{k+\mathfrak{h}-1}). \tag{5.63}$$

Again, these values form the packet that controller sends to a network, namely

$$\pi = (\mathfrak{a}, \mathfrak{d}) := (r^*, (u^*_{\mathfrak{p}_{r^*}}(k_i), \ldots, u^*_{\mathfrak{p}_{r^*}}(k_i + \mathfrak{h} - 1)). \tag{5.64}$$

The resulting closed-loop system is then given as[5]

$$\Sigma_x : x^+ = \begin{bmatrix} f_{\mathfrak{p}}(x_{\mathfrak{p}}, h_{\mathfrak{b}}(x_{\mathfrak{b}}, r^*, \{u^*_{\mathfrak{p}_{r^*}}\}_k^{k+\mathfrak{h}-1}, w_{\mathfrak{n}}), w_{\mathfrak{p}}) \\ f_{\mathfrak{b}}(x_{\mathfrak{b}}, r^*, \{u^*_{\mathfrak{p}_{r^*}}\}_k^{k+\mathfrak{h}-1}, w_{\mathfrak{n}}) \end{bmatrix} =: f(x, w_{\mathfrak{n}}, w_{\mathfrak{p}}). \tag{5.65}$$

Finally, we have all ingredients to state the last result of this chapter.

Theorem 5.2 (ISS). *Consider the system Σ_x, given by the equation (5.65). Let the following be satisfied:*

- *There exist class-$\mathcal{K}$ functions α_x and $\alpha_{w_{\mathfrak{p}}}$ such that*

[5]Recall that $y_{\mathfrak{b}} := h_{\mathfrak{b}}(x_{\mathfrak{b}}, \mathfrak{a}, \mathfrak{d}, w_{\mathfrak{n}}) := \Gamma\Big(\Xi(\mathfrak{a}, w_{\mathfrak{n}})x_{\mathfrak{b}} + \Pi(\mathfrak{d}, \mathfrak{a}, w_{\mathfrak{n}})\Big)$.

$$|f(x, w_{\mathfrak{n}}, w_{\mathfrak{p}}) - f(\hat{x}, w_{\mathfrak{n}}, 0^{q_{\mathfrak{p}}})| \leq \alpha_x(|x - \hat{x}|) + \alpha_{w_{\mathfrak{p}}}(|w_{\mathfrak{p}}|), \tag{5.66}$$

holds for each $x \in \mathbb{X}_{\mathfrak{p}_{\mathfrak{f}}} \times \mathbb{U}_{\mathfrak{p}}^{\mathfrak{h}}$, $\hat{x} \in \mathbb{X}_{\mathfrak{p}_{\mathfrak{f}}} \times \mathbb{U}_{\mathfrak{p}}^{\mathfrak{h}}$, $w_{\mathfrak{n}} \in \mathcal{D}$ *and* $w_{\mathfrak{p}} \in \mathbb{W}$,

- *The set* $\mathbb{W}_{\mathfrak{p}}$ *is compact and correspondingly there exist*

$$|\mathrm{w}| := \max_{k_0 \leq i \leq k} |w_{\mathfrak{p}}(i)|, \tag{5.67}$$

- *Assumption 5.2,*

- *There exist class-*$\mathcal{K}_\infty$ *function* α *such that*

$$\left.\begin{array}{c} l(x, u_{\mathfrak{p}}) \geq \alpha(|x|),\ l(0^{n_{\mathfrak{p}} \times \mathfrak{h} \cdot m_{\mathfrak{p}}}, 0^{m_{\mathfrak{p}}}) = 0,\ \forall x \in \mathbb{X}_{\mathfrak{p}_{\mathfrak{f}}} \times \mathbb{U}_{\mathfrak{p}}^{\mathfrak{h}},\ \forall u_{\mathfrak{p}} \in \mathbb{U}_{\mathfrak{p}}, \\ g(x) \geq 0,\ g(0^{n_{\mathfrak{p}} \times \mathfrak{h} \cdot m_{\mathfrak{p}}}) = 0,\ \forall x_{\mathfrak{p}} \in \mathbb{X}_{\mathfrak{p}_{\mathfrak{t}}} \times \mathbb{U}_{\mathfrak{p}}^{\mathfrak{h}}, \end{array}\right\} \tag{5.68}$$

- *For some node* $r \in \mathcal{R}$ *there exist a* terminal control law $\kappa_r : \mathbb{X}_{\mathfrak{p}_{\mathfrak{t}}} \to \tilde{\mathbb{U}}_{\mathfrak{p}_r}$ *such that*[6]

$$\left.\begin{array}{c} g\left(\begin{bmatrix} f_{\mathfrak{p}}(x_{\mathfrak{p}}, h_{\mathfrak{b}}(x_{\mathfrak{b}}, r, \{\iota_r \kappa_r\}_k^{k+\mathfrak{h}-1}, w_{\mathfrak{n}}), 0^{m_{\mathfrak{p}}}) \\ f_{\mathfrak{b}}(x_{\mathfrak{b}}, r, \{\iota_r \kappa_r\}_k^{k+\mathfrak{h}-1}, w_{\mathfrak{n}}) \end{bmatrix}\right) \\ - g(x) + l(x, h_{\mathfrak{b}}(x_{\mathfrak{b}}, r, \{\iota_r \kappa_r\}_k^{k+\mathfrak{h}-1}, w_{\mathfrak{n}})) \leq 0, \\ \begin{bmatrix} f_{\mathfrak{p}}(x_{\mathfrak{p}}, h_{\mathfrak{b}}(x_{\mathfrak{b}}, r, \{\iota_r \kappa_r\}_k^{k+\mathfrak{h}-1}, w_{\mathfrak{n}}), 0^{m_{\mathfrak{p}}}) \\ f_{\mathfrak{b}}(x_{\mathfrak{b}}, r, \{\iota_r \kappa_r\}_k^{k+\mathfrak{h}-1}, w_{\mathfrak{n}}) \end{bmatrix} \in \mathbb{X}_{\mathfrak{p}_{\mathfrak{t}}} \times \mathbb{U}_{\mathfrak{p}}^{\mathfrak{h}}, \\ \kappa_r(x_{\mathfrak{p}}) \in \tilde{\mathbb{U}}_{\mathfrak{p}_r} := 0^{m_{\mathfrak{p}_1}} \times \cdots \times 0^{m_{\mathfrak{p}_{r-1}}} \times \mathbb{U}_{\mathfrak{p}_r} \times 0^{m_{\mathfrak{p}_{r+1}}} \times \cdots \times 0^{m_{\mathfrak{p}_{\mathrm{R}}}}, \\ \mathbb{U}_{\mathfrak{p}_r} \subseteq \mathbb{R}^{m_{\mathfrak{p}_r}}, \\ \iota_r := diag(0_{m_{\mathfrak{p}_1}}, \ldots, 0_{m_{\mathfrak{p}_{r-1}}}, I_{m_{\mathfrak{p}_r}}, 0_{m_{\mathfrak{p}_{r+1}}}, \ldots, 0_{m_{\mathfrak{p}_{\mathrm{R}}}}), \end{array}\right\} \tag{5.69}$$

holds for each $x_{\mathfrak{p}} \in \mathbb{X}_{\mathfrak{p}_{\mathfrak{t}}}$

- *There exist a class-*$\mathcal{K}_\infty$ *function* γ_V *such that*

$$|V(x) - V(\hat{x})| \leq \gamma_V(|x - \hat{x}|) \tag{5.70}$$

[6]Recall that $y_{\mathfrak{b}} := h_{\mathfrak{b}}(x_{\mathfrak{b}}, \mathfrak{a}, \mathfrak{d}, w_{\mathfrak{n}}) := \Gamma\Big(\Xi(\mathfrak{a}, w_{\mathfrak{n}}) x_{\mathfrak{b}} + \Pi(\mathfrak{d}, \mathfrak{a}, w_{\mathfrak{n}})\Big)$.

holds for each $x \in \mathbb{X}_{\mathfrak{p}_f} \times \mathbb{U}_{\mathfrak{p}}^{\mathfrak{h}}$ *and* $\hat{x} \in \mathbb{X}_{\mathfrak{p}_f} \times \mathbb{U}_{\mathfrak{p}}^{\mathfrak{h}}$,

- *At the first successful transmission instant we have that* $x(k_0) \in \mathbb{X}_{\mathfrak{p}_f} \times \mathbb{U}_{\mathfrak{p}}^{\mathfrak{h}}$.

Then, there exist a class-$\mathcal{KL}$ *function* β *and a class-*$\mathcal{K}$ *function* γ *such that the states of system* Σ_x *are bounded via*

$$|x(k)| \leq \beta(|x(k_0)|, k - k_0) + \gamma(|\mathrm{w}|) \tag{5.71}$$

for each $k \geq k_0$. □

The proof of this theorem follows the exact same lines as the proof of the main result from [31] and thus is omitted.

Before closing this section we discuss some assumptions we used. Let us begin with the assumption provided by the equation (5.67). This assumption comes from the definition of ISS, for instance, see Definition 4.7 in [169]. Next we focus on the assumption given in the equation (5.66). This assumption reflects the dependency on buffer contents which is caused by dynamic scheduling. The assumption captured with the equation (5.70) mimics the frequently used assumption when one considers robustness properties of the corresponding system governed by MPC based controllers. For instance, for non-networked systems see [107, 122, 170], while for the networked cases, see [31]. Next, the assumption for stage and terminal cost functions, given by the equation (5.68), is a reasonable extension of an assumption used in non-networked cases, e.g., see [107, 119] and networked cases, e.g., see [31]. On the other hand, the assumption related to terminal control law (see (5.69)) stems from the fact that the cost function considers buffer content explicitly, and we can access only one input node at each time instant. It should be noted that, as before, the control law used in (5.69) is just a construct which is not necessarily used on the plant.

Finally, we finish by providing some insight into why ISS of the augmented state of the plant and the buffer state instead of Input-Output Stability (IOS), where the output is the plant state. As documented in [171], to show IOS one would need to show that the corresponding system satisfies the *output-uniform asymptotic gain property* (see Definition 1.6 and Theorem 1 in [171]). Due to the specific structure of the NCS configuration we

were unable to show that this property is satisfied. Recall that the network allows access to only one input node at each time instant. Hence, the buffer content is used to derive the sequence of optimal control predictions. This fact makes some needed assumptions hard to satisfy. For instance, one would need the following continuity assumption on the optimal value function:"There exist a class-$\mathcal{K}_\infty$ function σ such that $|V(x) - V(\hat{x})| \leq \sigma|x - \hat{x}|$ holds for each $x \in \mathbb{X}_{\mathfrak{p}_f} \times \mathbb{U}_{\mathfrak{p}}^{\mathfrak{h}}$ and $\hat{x} \in \mathbb{X}_{\mathfrak{p}_f} \times \mathbb{U}_{\mathfrak{p}}^{\mathfrak{h}}$". Now, note that if the optimal value function is quadratic and $x = (0^{n_\mathfrak{p}}, x_\mathfrak{b})$ and $\hat{x} = (0^{n_\mathfrak{p}}, \hat{x}_\mathfrak{b})$ where $x_\mathfrak{b} \neq \hat{x}_\mathfrak{b}$, the latter assumption is not satisfied. Thus, we focused on ISS of the overall NCS state and showed that by modifying the cost function and exploiting stronger assumptions it is achievable.

5.2.3 Simulations

Here, we illustrate the performance aspects of the used method. First, let us emphasize that we allow for Assumption 5.2 to be violated. This is due to the fact that in practice we can not guarantee the corresponding bound on the number of consecutive packet dropouts because of the random nature of dropouts; however, note that due to the fact that the networks are designed so that the network throughput is high, the number of *consecutive* packet dropouts should not be high.

The demonstration of the used method includes the comparison between dynamic and static scheduling. Moreover, when no data is received over $\mathfrak{h}$ consecutive time instances, we compare two buffer mechanisms that deal with this issue; these mechanisms correspond to different shift matrices. One mechanism corresponds to setting buffer state to zero; for instance see shift matrix S_3 in Example 3.1. The other mechanism corresponds to setting buffer state to the "last received value"; for instance, see shift matrix $\tilde{S}_3$ in Example 3.1.

The considered system is of the form

$$
\begin{aligned}
x_{\mathfrak{p}_1}^+ &= x_{\mathfrak{p}_1} + u_{\mathfrak{p}_1} \\
x_{\mathfrak{p}_2}^+ &= x_{\mathfrak{p}_2} + u_{\mathfrak{p}_2} \\
x_{\mathfrak{p}_3}^+ &= x_{\mathfrak{p}_3} + x_{\mathfrak{p}_1} u_{\mathfrak{p}_2} - x_{\mathfrak{p}_2} u_{\mathfrak{p}_1} + w_\mathfrak{p}
\end{aligned}
$$

where $(x_{\mathfrak{p}_1}, x_{\mathfrak{p}_2}, x_{\mathfrak{p}_3}) = x_{\mathfrak{p}} \in \mathbb{X}_{\mathfrak{p}} \subset \mathbb{R}^3$ and $(u_{\mathfrak{p}_1}, u_{\mathfrak{p}_2}) \in \mathbb{U}_{\mathfrak{p}} \subset \mathbb{R}^2$ where $\mathbb{X}_{\mathfrak{p}}$ is a compact set; $x_{\mathfrak{p}}(0) = (-4, 4, 4)$. Disturbance $w_{\mathfrak{p}} \in \mathbb{W} \subseteq \mathbb{R}$ is not known to the controller and it will be defined in the sequel.

The nominal model (see (5.12)) is *the discrete nonholonomic integrator* and its dynamics is considered in an MPC framework in [123] (see *Example 2*). Using results presented there it follows that by choosing $\bar{\Gamma} > 16/5$ and $\mathfrak{h} = 4$ (see *Example 2*), one can select stage and terminal cost as

$$l(x_{\mathfrak{p}}, u_{\mathfrak{p}}) = x_{\mathfrak{p}_1}^2 + x_{\mathfrak{p}_2}^2 + 10|x_{\mathfrak{p}_3}| \text{ and } g(x_{\mathfrak{p}}) = 4(x_{\mathfrak{p}_1}^2 + x_{\mathfrak{p}_2}^2 + 10|x_{\mathfrak{p}_3}|).$$

It follows that the corresponding MPC feedback law will globally asymptotically stabilize the origin; for details see [123].

Disturbance is constructed as follows, for time intervals $\{0, 1, \ldots, 20\}$ and $\{40, 41, \ldots, 100\}$ it is a random Gaussian process with zero mean and variance of 0.01, while during time interval $\{20, 21, \ldots, 40\}$ it is a step of amplitude one plus random Gaussian process with zero mean and variance of 0.3.

Recall that we allow for Assumption 5.2 to be violated since in reality the packet dropouts are random. Thus, first we consider a network which introduces *i.i.d.* dropouts with 0.2 probability where probability distribution is discrete Bernoulli distribution. Figure 5.2 illustrates comparison between dynamic[7] (solid line) and static[8] (dashed line) scheduling when buffer is set to zero while Figure 5.3 captures the scenario when buffer is set to the "last received value"[9].

First subplots of Fig. 5.2 and Fig. 5.3 clearly illustrate better performance when dynamic scheduling is applied regardless of how buffer state is set when no data is received over $\mathfrak{h}$ consecutive time instances.

Next, we increase the dropout probability to 0.6. Note that in this scenario the buffers will more frequently ran out of data due to frequent 4 or more consecutive packet dropouts

[7] Try Once Discard protocol.

[8] Round Robin protocol.

[9] In all figures from this subsection, in the first subplot we illustrate $\mathcal{L}_2$ norms of the state of the considered system while in subplots 2 (dynamic case) and 3 (static case) we depict corresponding scheduling (solid line) and packet dropouts (diamonds) for channels 1 and 2. Finally, all subplots in each figure share the same time denoted at the bottom of the last subplot.

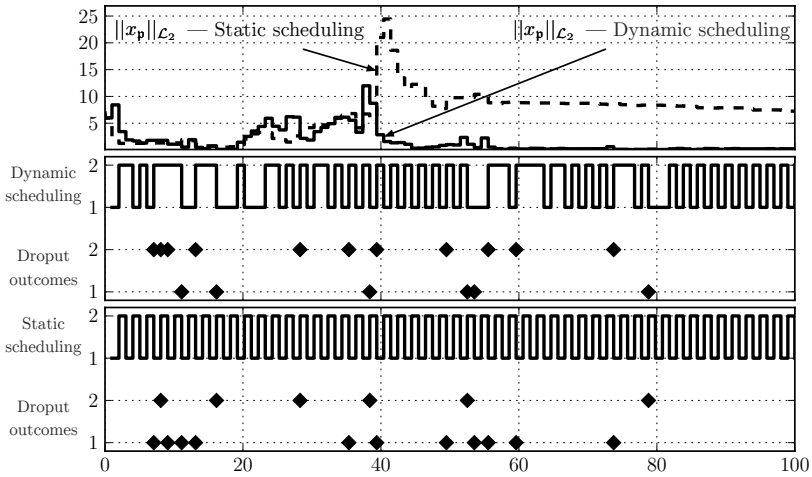


Figure 5.2: Dynamic and static scheduling comparison; buffer set to zero value; dropout probability 0.2, 1 – channel 1 (e.g., $u_{\mathfrak{p}_1}$) while 2 – channel 2 (e.g., $u_{\mathfrak{p}_2}$).

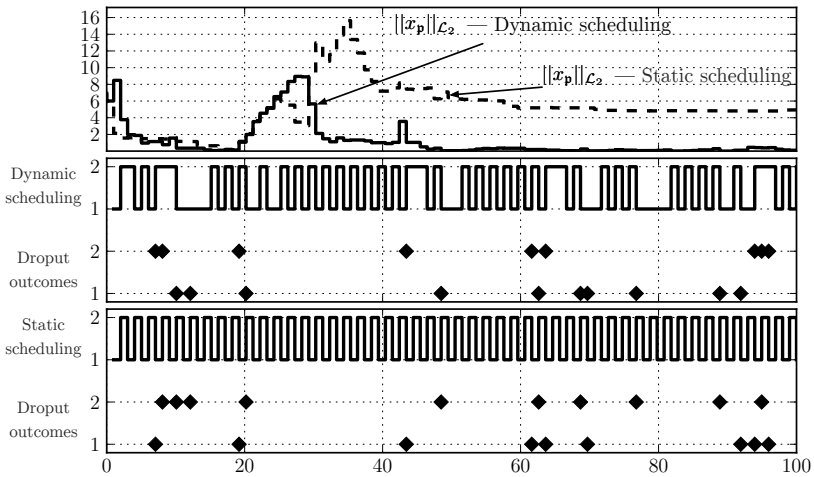


Figure 5.3: Dynamic and static scheduling comparison; buffer set to the "last value"; dropout probability 0.2, 1 – channel 1 (e.g., $u_{\mathfrak{p}_1}$) while 2 – channel 2 (e.g., $u_{\mathfrak{p}_2}$).

as documented in the second and the third subplot of Fig. 5.4 and Fig. 5.5. Notice that even in this severe dropout scenario, and despite the assumptions of the stability theorem not being satisfied, the used method gives good behavior as illustrated in first subplots of Fig. 5.4 and Fig. 5.5. Again, dynamic scheduling gives better performance when

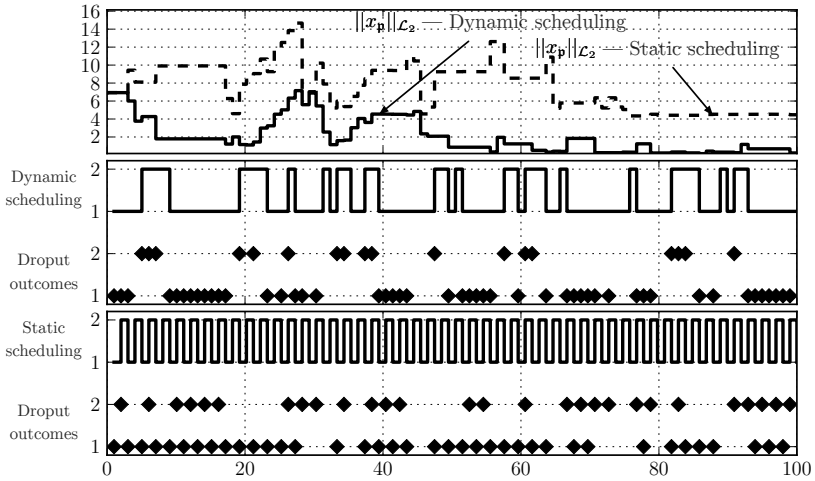


Figure 5.4: Dynamic and static scheduling comparison; buffer set to zero value; dropout probability 0.6, 1 – channel 1 (e.g., $u_{\mathfrak{p}_1}$) while 2 – channel 2 (e.g., $u_{\mathfrak{p}_2}$).

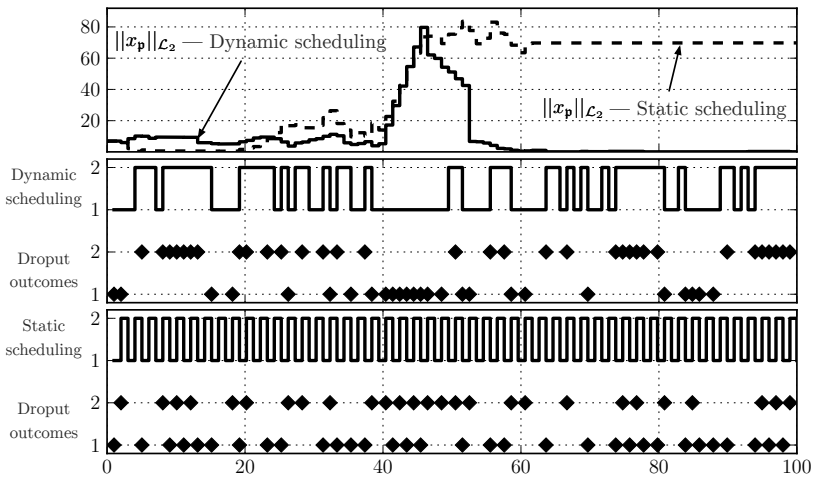


Figure 5.5: Dynamic and static scheduling comparison; buffer set to the "last value"; dropout probability 0.6, 1 – channel 1 (e.g., $u_{\mathfrak{p}_1}$) while 2 – channel 2 (e.g., $u_{\mathfrak{p}_2}$).

compared to static scheduling.

In summary, we conclude that simulations demonstrated that the used method showed good performance and that dynamic outperformed static scheduling.

Similarly as in previous chapters we proceed with proofs.

5.3 Proofs

Proof of Lemma 5.1. Assumption 5.2 implies that $k_0 \leq \mathfrak{h}$, thus, form (5.35) it follows that $x \in \mathbb{X}_{\mathfrak{p}_\mathfrak{f}} \times \mathbb{U}_\mathfrak{p}^\mathfrak{h}$. Then, by using (5.36) one can easily show by induction that $x(k_i) \in \mathbb{X}_{\mathfrak{p}_\mathfrak{f}} \times \mathbb{U}_\mathfrak{p}^\mathfrak{h}$ for any $k_i \in \mathbb{K}_{\Delta k}$. □

Proof of Lemma 5.2. We start by splitting the left hand side of inequality (5.37) into the part that involves the disturbance and the part that does not. More precisely, we have

$$\begin{aligned}
\Delta V(x(k_i)) &:= V(x(k_{i+1})) - V(x(k_i)) \\
&= V(\phi_f(\Delta k_i + 1, x, \{w_\mathfrak{n}\}_{k_i}^{k_i+\Delta k_i}, \{w_\mathfrak{p}\}_{k_i}^{k_i+\Delta k_i})) - V(x(k_i)) \\
&= V(\phi_f(\Delta k_i + 1, x, \{w_\mathfrak{n}\}_{k_i}^{k_i+\Delta k_i}, \{0, \ldots, 0\})) - V(x(k_i)) \\
&\quad + V(\phi_f(\Delta k_i + 1, x, \{w_\mathfrak{n}\}_{k_i}^{k_i+\Delta k_i}, \{w_\mathfrak{p}\}_{k_i}^{k_i+\Delta k_i})) \\
&\quad - V(\phi_f(\Delta k_i + 1, x, \{w_\mathfrak{n}\}_{k_i}^{k_i+\Delta k_i}, \{0, \ldots, 0\})).
\end{aligned} \tag{5.72}$$

- $V(\phi_f(\Delta k_i + 1, x, \{w_\mathfrak{n}\}_{k_i}^{k_i+\Delta k_i}, \{0, \ldots, 0\})) - V(x(k_i))$:

  1. $\Delta k_i < \Delta k - 1$: Consider a node from Assumption 5.4 at time instant k_{i+1}, that is, $r(k_{i+1})$, which is not necessarily the same as $r^*(k_i)$. Consequently, consider a control sequence

$$\begin{aligned}
\{\tilde{u}_\mathfrak{p}^{r(k_{i+1})}\}_{k_{i+1}}^{k_{i+1}+\Delta k-1} = \{&u_\mathfrak{p}^{r^*(k_i)}(k_{i+1}), \ldots, u_\mathfrak{p}^{r^*(k_i)}(k_i + \Delta k - 1), \\
&\tilde{u}_\mathfrak{p}^{r(k_{i+1})}(k_i + \Delta k), \ldots, \tilde{u}_\mathfrak{p}^{r(k_{i+1})}(k_{i+1} + \Delta k - 1)\}
\end{aligned} \tag{5.73}$$

  where $u_\mathfrak{p}^{r^*(k_i)}(\cdot)$ stands for plant input with node $r^*(k_i)$ being the last updated part of plant input; i.e., the last updated control values are the ones for node $r^*(k_i)$. More precisely, the first $\Delta k - \Delta k_i - 1$ elements are from the buffers which originate from past optimizations, last one received being obtained in optimization at time instant k_i; see Section 5.1.4 for more details. The rest of elements come from Assumption 5.4, namely

$$\tilde{u}_{\mathfrak{p}}^{r(k_{i+1})}(k_{i+1}+j) := \kappa_{r(k_{i+1})}(\tilde{x}_{\mathfrak{p}}(k_{i+1}+j)),\ \forall j \in \{\Delta k - \Delta k_i - 1, \ldots, \Delta k - 1\} \tag{5.74}$$

where

$$\tilde{x}_{\mathfrak{p}}^{+} = f_{\mathfrak{p}}(\tilde{x}_{\mathfrak{p}}, \kappa_{r(k_{i+1})}, 0^{q_{\mathfrak{p}}}) \tag{5.75}$$

with

$$\tilde{x}_{\mathfrak{p}} = \phi_{f_{\mathfrak{p}}}(\Delta k, x, \{w_{\mathfrak{n}}\}_{k_i}^{k_i+\Delta k-1}, \{0, \ldots, 0\}). \tag{5.76}$$

Note that the sequence given in (5.73) is a feasible one. Now, direct application of this sequence in the corresponding cost function produces the following

$$\begin{aligned}
&J(\phi_f(\Delta k_i + 1, x, \{w_{\mathfrak{n}}\}_{k_i}^{k_i+\Delta k_i}), \{\tilde{u}_{\mathfrak{p}_{r(k_{i+1})}}\}_{k_{i+1}}^{k_{i+1}+\Delta k-1}, \{0, \ldots, 0\})) \\
&\quad = V(x) - \sum_{j=k_i}^{k_i+\Delta k_i} l(\phi_{f_{\mathfrak{p}}}(j-k_i, x, \{w_{\mathfrak{n}}\}_{k_i}^{k_i+\Delta k_i}, \{0, \ldots, 0\}), u_{\mathfrak{p}}^{r^*(k_i)}(j)) \\
&\qquad + \sum_{j=k_i+\Delta k}^{k_{i+1}=\Delta k-1} \Big\{ g(f_{\mathfrak{p}}(\tilde{x}(j), \tilde{u}_{\mathfrak{p}}(j), 0^{q_{\mathfrak{p}}})) - g(\tilde{x}(j)) + l(\tilde{x}(j), \tilde{u}_{\mathfrak{p}}(j)) \Big\} \\
&\quad \leq V(x) - \sum_{j=k_i}^{k_i+\Delta k_i} l(\phi_{f_{\mathfrak{p}}}(j-k_i, x, \{w_{\mathfrak{n}}\}_{k_i}^{k_i+\Delta k_i}, \{0, \ldots, 0\}), u_{\mathfrak{p}}^{r^*(k_i)}(j))
\end{aligned} \tag{5.77}$$

where $\{\tilde{u}_{\mathfrak{p}_{r(k_{i+1})}}\}_{k_{i+1}}^{k_{i+1}+\Delta k-1}$ is a sequence of corresponding control values for node $r(k_{i+1})$ from a feasible control sequence given in (5.73); also, we write $\{u_{\mathfrak{p}}^{r^*}\}_{k_i}^{k_i+\Delta k_i}$ instead of $\{u_{\mathfrak{p}}^{r^*(k_i)}\}_{k_i}^{k_i+\Delta k_i}$ to simplify notation. Now, due to optimality, we have

$$\begin{aligned}
&V(\phi_f(\Delta k_i + 1, x, \{w_{\mathfrak{n}}\}_{k_i}^{k_i+\Delta k_i}, \{0, \ldots, 0\})) \\
&\qquad \leq J(\phi_f(\Delta k_i + 1, x, \{w_{\mathfrak{n}}\}_{k_i}^{k_i+\Delta k_i}, \{0, \ldots, 0\}), \{\tilde{u}_{\mathfrak{p}_{r(k_{i+1})}}\}_{k_{i+1}}^{k_{i+1}+\Delta k-1}),
\end{aligned} \tag{5.78}$$

thus, application of Assumption 5.3 yields

$$\begin{aligned}\Delta V(x(k_i)) &:= V(\phi_f(\Delta k_i+1, x, \{w_{\mathfrak{n}}\}_{k_i}^{k_i+\Delta k_i}, \{0,\dots,0\})) - V(x)\\ &\le \sum_{j=k_i}^{k_i+\Delta k_i} l(\phi_{f_{\mathfrak{p}}}(j-k_i, x, \{w_{\mathfrak{n}}\}_{k_i}^{k_i+\Delta k_i}, \{0,\dots,0\}), u_{\mathfrak{p}}^{r^*(k_i)}(j))\\ &\le -\sum_{j=k_i}^{k_i+\Delta k_i} \alpha_{x_{\mathfrak{p}}}(|\phi_{f_{\mathfrak{p}}}(j-k_i, x, \{w_{\mathfrak{n}}\}_{k_i}^{k_i+\Delta k_i}, \{0,\dots,0\})|),\end{aligned} \tag{5.79}$$

as desired.

2. $\Delta k_i = \Delta k - 1$: The only difference is that now, we consider a *feasible* control sequence $\{\tilde{u}_{\mathfrak{p}}^{r(k_{i+1})}(k_{i+1}), \dots, \tilde{u}_{\mathfrak{p}}^{r(k_{i+1})}(k_{i+1}+\Delta k-1)\}$, whose values come from Assumption 3.3.

- $V(\phi_f(\Delta k_i+1, x, \{w_{\mathfrak{n}}\}_{k_i}^{k_i+\Delta k_i}, \{w_{\mathfrak{p}}\}_{k_i}^{k_i+\Delta k_i})) - V(\phi_f(\Delta k_i+1, x, \{w_{\mathfrak{n}}\}_{k_i}^{k_i+\Delta k_i}, \{0,\dots,0\}))$: First, note that from Assumption 5.5 it follows that,

$$\begin{aligned}&V(\phi_f(\Delta k_i+1, x, \{w_{\mathfrak{n}}\}_{k_i}^{k_i+\Delta k_i}, \{w_{\mathfrak{p}}\}_{k_i}^{k_i+\Delta k_i})) - V(\phi_f(\Delta k_i+1, x, \{w_{\mathfrak{n}}\}_{k_i}^{k_i+\Delta k_i}, \{0,\dots,0\}))\\ &\quad\le \gamma_V(|\phi_{f_{\mathfrak{p}}}(\Delta k_i+1, x, \{w_{\mathfrak{n}}\}_{k_i}^{k_i+\Delta k_i}, \{w_{\mathfrak{p}}\}_{k_i}^{k_i+\Delta k_i})\\ &\qquad\qquad - \phi_{f_{\mathfrak{p}}}(\Delta k_i+1, x, \{w_{\mathfrak{n}}\}_{k_i}^{k_i+\Delta k_i}, \{0,\dots,0\})|)\end{aligned} \tag{5.80}$$

It might be useful to recall that $\phi_f(\Delta k_i+1, x, \{w_{\mathfrak{n}}\}_{k_i}^{k_i+\Delta k_i}, \{w_{\mathfrak{p}}\}_{k_i}^{k_i+\Delta k_i})$ and $\phi_f(\Delta k_i+1, x, \{w_{\mathfrak{n}}\}_{k_i}^{k_i+\Delta k_i}, \{0,\dots,0\})$ have the same buffer contents and that $\{w_{\mathfrak{n}}\}_{k_i}^{k_i+\Delta k_i} = \{1,0,\dots,0\}$. Applying Assumption 5.1 and the property of solution mapping, namely

$$\phi_{f_\mathfrak{p}}(j-k_i, x, \{w_\mathfrak{n}\}_{k_i}^{j-1}, \{w_\mathfrak{p}\}_{k_i}^{j-1}) = \begin{cases} x_\mathfrak{p}(k_i) & \text{if } j = k_i \\ f_\mathfrak{p}(\phi_{f_\mathfrak{p}}(j-k_i-1, x, \{w_\mathfrak{n}\}_{k_i}^{j-2}, \{w_\mathfrak{p}\}_{k_i}^{j-2}), u_\mathfrak{p}(j-1), w_\mathfrak{p}(j-1)) & \text{if } j \in \{k_i+1, \dots, k_i+\mathfrak{h}\} \end{cases} \tag{5.81}$$

one arrives at

$$\begin{aligned} &|\phi_{f_\mathfrak{p}}(j-k_i, x, \{w_\mathfrak{n}\}_{k_i}^{j-1}, \{w_\mathfrak{p}\}_{k_i}^{j-1}) - \phi_{f_\mathfrak{p}}(j-k_i, x, \{w_\mathfrak{n}\}_{k_i}^{j-1}, \{0, \dots, 0\})| \\ &\quad = |f_\mathfrak{p}(\phi_{f_\mathfrak{p}}(j-k_i-1, x, \{w_\mathfrak{n}\}_{k_i}^{j-2}, \{w_\mathfrak{p}\}_{k_i}^{j-2}), u_\mathfrak{p}(j-1), w_\mathfrak{p}(j-1)) \\ &\qquad\qquad - f_\mathfrak{p}(\phi_{f_\mathfrak{p}}(j-k_i-1, x, \{w_\mathfrak{n}\}_{k_i}^{j-2}, \{0, \dots, 0\}), u_\mathfrak{p}(j-1), 0)| \\ &\quad \leq \alpha_{x_\mathfrak{p}}(|\phi_{f_\mathfrak{p}}(j-k_i-1, x, \{w_\mathfrak{n}\}_{k_i}^{j-2}, \{w_\mathfrak{p}\}_{k_i}^{j-2}) \\ &\qquad\qquad - \phi_{f_\mathfrak{p}}(j-k_i-1, x, \{w_\mathfrak{n}\}_{k_i}^{j-2}, \{0, \dots, 0\})|) + \alpha_{w_\mathfrak{p}}(|w_\mathfrak{p}(j-1)|) \end{aligned} \tag{5.82}$$

which holds for any $j \in \{k_i, \dots, k_{i+1}\}$ with $|x_\mathfrak{p}(k_i) - x_\mathfrak{p}(k_i)| = 0$ for $j = k_i$. Using the latter inequality one can easily show that

$$\begin{aligned} &|\phi_{f_\mathfrak{p}}(j-k_i, x, \{w_\mathfrak{n}\}_{k_i}^{j-1}, \{w_\mathfrak{p}\}_{k_i}^{j-1}) - \phi_{f_\mathfrak{p}}(j-k_i, x, \{w_\mathfrak{n}\}_{k_i}^{j-1}, \{0, \dots, 0\})| \\ &\quad \leq \sum_{l=1}^{j-k_i} \bar{\alpha}_l(|w_\mathfrak{p}(j-l)|) \end{aligned} \tag{5.83}$$

holds for any $j \in \{k_i, \dots, k_{i+1}\}$ where $\bar{\alpha}_{l+1} := \alpha_{x_\mathfrak{p}} \circ 2 \cdot \mathrm{Id} \circ \bar{\alpha}_l$ is a class-$\mathcal{K}$ function and $\bar{\alpha}_1 = \alpha_{w_\mathfrak{p}}$. Now, note that due to Assumption 5.2 $k_{i+1} - k_i \leq \mathfrak{h} = \Delta k$, and note that since $\alpha_{x_\mathfrak{p}}$ and $\alpha_{w_\mathfrak{p}}$ are a class-$\mathcal{K}$ functions one can always bound them with new class-$\mathcal{K}$ functions so that $\bar{\alpha}_{l+1} > \bar{\alpha}_l$. Thus, it follows $\bar{\alpha}_\mathfrak{h} = \max_{l \in \{1, \dots, \mathfrak{h}\}} \bar{\alpha}_l$. Then, from latter inequality we have

$$
\begin{aligned}
&|\phi_{f_\mathfrak{p}}(j-k_i, x, \{w_\mathfrak{n}\}_{k_i}^{j-1}, \{w_\mathfrak{p}\}_{k_i}^{j-1}) - \phi_{f_\mathfrak{p}}(j-k_i, x, \{w_\mathfrak{n}\}_{k_i}^{j-1}, \{0,\dots,0\})| \\
&\quad \leq \sum_{l=k_i}^{j-1} \bar{\alpha}_\mathfrak{h}(|w_\mathfrak{p}(l)|)
\end{aligned} \tag{5.84}
$$

for any $j \in \{k_i, \dots, k_{i+1}\}$. Setting $j = k_i + \Delta k_i + 1$ $(= k_{i+1})$ results in

$$
\begin{aligned}
&|\phi_{f_\mathfrak{p}}(\Delta k_i+1, x, \{w_\mathfrak{n}\}_{k_i}^{k_i+\Delta k_i}, \{w_\mathfrak{p}\}_{k_i}^{k_i+\Delta k_i}) - \phi_{f_\mathfrak{p}}(j, x, \{w_\mathfrak{n}\}_{k_i}^{k_i+\Delta k_i}, \{0,\dots,0\})| \\
&\quad \leq \sum_{l=k_i}^{k_i+\Delta k_i} \bar{\alpha}_\mathfrak{h}(|w_\mathfrak{p}(l)|)
\end{aligned} \tag{5.85}
$$

Now, recall that due to Assumption 5.3, 5.4 and 5.8 $V(\phi_f(\Delta k_i+1, x, \{w_\mathfrak{n}\}_{k_i}^{k_i+\Delta k_i}, \{w_\mathfrak{p}\}_{k_i}^{k_i+\Delta k_i}))$ is bounded. Further, using the second inequality from (1.8) yields

$$
\begin{aligned}
&V(\phi_f(\Delta k_i+1, x, \{w_\mathfrak{n}\}_{k_i}^{k_i+\Delta k_i}, \{w_\mathfrak{p}\}_{k_i}^{k_i+\Delta k_i})) \\
&\qquad - V(\phi_f(\Delta k_i+1, x, \{w_\mathfrak{n}\}_{k_i}^{k_i+\Delta k_i}, \{0,\dots,0\})) \\
&\qquad\qquad \leq \alpha_V\left(\sum_{l=k_i}^{k_i+\Delta k_i} \bar{\alpha}_\mathfrak{h}(|w_\mathfrak{p}(l)|)\right) \\
&\qquad\qquad \leq \sum_{l=k_i}^{k_i+\Delta k_i} \alpha_V(2^{\Delta k_i}\bar{\alpha}_\mathfrak{h}(|w_\mathfrak{p}(l)|)) \\
&\qquad\qquad \leq \sum_{l=k_i}^{k_i+\Delta k_i} \alpha_V(2^{\Delta k-1}\bar{\alpha}_\mathfrak{h}(|w_\mathfrak{p}(l)|))
\end{aligned} \tag{5.86}
$$

Towards combining the latter inequality with the inequality (5.79) to establish the desired dissipation inequality we will first add and subtract $|\phi_{f_\mathfrak{p}}(j-k_i, x, \{w_\mathfrak{n}\}_{k_i}^{k_i+\Delta k_i}, \{w_\mathfrak{p}\}_{k_i}^{k_i+\Delta k_i})|$ to the argument of the $\alpha_{x_\mathfrak{p}}$ function on the right hand side of the inequality (5.79). Then, we will use first inequality from set of inequalities given in (1.8) and the triangle inequality (e.g., $|x| - |y| \leq |x-y|$, $\forall x \in \mathbb{R}^n, y \in \mathbb{R}^n$, $n \in \mathbb{N}$) to obtain

$$
\begin{aligned}
\Delta V(x(k_i)) &:= V(\phi_f(\Delta k_i+1, x, \{w_{\mathfrak{n}}\}_{k_i}^{k_i+\Delta k_i}, \{0,\dots,0\})) - V(x) \\
&\le -\sum_{j=k_i}^{k_i+\Delta k_i} \alpha_{x_{\mathfrak{p}}}(|\phi_{f_{\mathfrak{p}}}(j-k_i, x, \{w_{\mathfrak{n}}\}_{k_i}^{k_i+\Delta k_i}, \{0,\dots,0\})| \\
&\quad + |\phi_{f_{\mathfrak{p}}}(j-k_i, x, \{w_{\mathfrak{n}}\}_{k_i}^{k_i+\Delta k_i}, \{w_{\mathfrak{p}}\}_{k_i}^{k_i+\Delta k_i})| \\
&\quad - |\phi_{f_{\mathfrak{p}}}(j-k_i, x, \{w_{\mathfrak{n}}\}_{k_i}^{k_i+\Delta k_i}, \{w_{\mathfrak{p}}\}_{k_i}^{k_i+\Delta k_i})|) \\
&\le -\sum_{j=k_i}^{k_i+\Delta k_i} \frac{\alpha_{x_{\mathfrak{p}}}(|\phi_{f_{\mathfrak{p}}}(j-k_i, x, \{w_{\mathfrak{n}}\}_{k_i}^{k_i+\Delta k_i}, \{w_{\mathfrak{p}}\}_{k_i}^{k_i+\Delta k_i})|)}{2} \\
&\quad + \sum_{j=k_i}^{k_i+\Delta k_i} \frac{1}{2}\alpha_{x_{\mathfrak{p}}}(|\phi_{f_{\mathfrak{p}}}(j-k_i, x, \{w_{\mathfrak{n}}\}_{k_i}^{k_i+\Delta k_i}, \{w_{\mathfrak{p}}\}_{k_i}^{k_i+\Delta k_i})| \\
&\qquad - |\phi_{f_{\mathfrak{p}}}(j-k_i, x, \{w_{\mathfrak{n}}\}_{k_i}^{k_i+\Delta k_i}, \{0,\dots,0\}|)
\end{aligned}
\tag{5.87}
$$

Moreover, by using inequality (5.83) we can bound second term from the latter inequality as

$$
\begin{aligned}
&\sum_{j=k_i}^{k_i+\Delta k_i} \frac{1}{2}\alpha_{x_{\mathfrak{p}}}(|\phi_{f_{\mathfrak{p}}}(j-k_i, x, \{w_{\mathfrak{n}}\}_{k_i}^{k_i+\Delta k_i}, \{w_{\mathfrak{p}}\}_{k_i}^{k_i+\Delta k_i})| - |\phi_{f_{\mathfrak{p}}}(j-k_i, x, \{w_{\mathfrak{n}}\}_{k_i}^{k_i+\Delta k_i}, \{0,\dots,0\}|) \\
&\qquad \le \sum_{j=k_i}^{k_i+\Delta k_i} \frac{1}{2}\alpha_{x_{\mathfrak{p}}}\left(\sum_{l=k_i}^{j-1} \bar{\alpha}_{\mathfrak{h}}(|w_{\mathfrak{n}}(l)|)\right).
\end{aligned}
\tag{5.88}
$$

Now, we will use convention that whenever the lower index in the sum is greater than the upper one the sum is equal to zero. Thus, for $= k_i$ the inner sum in latter inequality will be equal to zero. Now, applying third inequality from set of inequalities given in (1.8) we arrive at

$$
\begin{aligned}
\sum_{j=k_i}^{k_i+\Delta k_i} \frac{1}{2}\alpha_{x_{\mathfrak{p}}}\left(\sum_{l=k_i}^{j-1} \bar{\alpha}_{\mathfrak{h}}(|w_{\mathfrak{n}}(l)|)\right) &\le (\Delta k_i+1) \sum_{l=k_i}^{k_i+\Delta k_i} \frac{1}{2}\alpha_{x_{\mathfrak{p}}}(2^{\Delta k_i}\bar{\alpha}_{\mathfrak{h}}(|w_{\mathfrak{n}}(l)|)) \\
&\le \Delta k \sum_{l=k_i}^{k_i+\Delta k_i} \frac{1}{2}\alpha_{x_{\mathfrak{p}}}(2^{\Delta k-1}\bar{\alpha}_{\mathfrak{h}}(|w_{\mathfrak{n}}(l)|)).
\end{aligned}
\tag{5.89}
$$

Finally, first we replace j with l in the first sum of the last inequality of (5.87). Then we use inequalities (5.89) and (5.88) in inequality (5.87) and combine the new inequality with (5.86). Then we define functions

$$\left.\begin{aligned} \alpha_1 &:= \tfrac{1}{2} \cdot \mathrm{Id} \circ \alpha \in \mathcal{K}_\infty, \\ \alpha_2 &:= \max\{\tfrac{\mathfrak{h}}{2} \cdot \mathrm{Id} \circ \alpha \circ 2^{\mathfrak{h}-1} \cdot \mathrm{Id} \circ \bar{\alpha}_{\mathfrak{h}}, \gamma_V \circ 2^{\mathfrak{h}-1} \cdot \mathrm{Id} \circ \bar{\alpha}_{\mathfrak{h}}\} \in \mathcal{K}, \end{aligned}\right\} \tag{5.90}$$

which completes the proof. □

Proof of Lemma 5.3. Let the conditions of Lemma 5.3 be satisfied. We first consider time interval $\{k_0, \ldots, k_i\}$ by applying Lemma 5.2 i times which results in i inequalities which cover the time interval of interest. Adding these inequalities results in

$$\begin{aligned} V(x(k_i)) - V(x(k_0)) &\leq -\sum_{j=0}^{i-1} \sum_{l=k_j}^{k_{j+1}-1} \alpha_1(|\phi_{f_{\mathfrak{p}}}(l-k_0, x, \{w_{\mathfrak{n}}\}_{k_0}^{l-1}, \{w_{\mathfrak{p}}\}_{k_0}^{l-1})|) \\ &\quad + \sum_{j=0}^{i-1} \sum_{l=k_j}^{k_{j+1}-1} \alpha_2(|w_{\mathfrak{p}}(l)|) \\ &= -\sum_{l=k_0}^{k_i-1} \alpha_1(|\phi_{f_{\mathfrak{p}}}(l-k_0, x, \{w_{\mathfrak{n}}\}_{k_0}^{l-1}, \{w_{\mathfrak{p}}\}_{k_0}^{l-1})|) \\ &\quad + \sum_{l=k_0}^{k_i-1} \alpha_2(|w_{\mathfrak{p}}(l)|). \end{aligned} \tag{5.91}$$

Moreover, using Assumption 5.6 yields

$$\begin{aligned} V(x(k_i)) &+ \sum_{l=k_0}^{k_i-1} \alpha_1(|\phi_{f_{\mathfrak{p}}}(l-k_0, x, \{w_{\mathfrak{n}}\}_{k_0}^{l-1}, \{w_{\mathfrak{p}}\}_{k_0}^{l-1})|) \\ &\leq V(x(k_0)) + \sum_{l=k_0}^{k_i-1} \alpha_2(|w_{\mathfrak{p}}(l)|) \\ &\leq \gamma_{\bar{V}}(|x(k_0)|) + \sum_{l=k_0}^{k_i-1} \alpha_2(|w_{\mathfrak{p}}(l)|). \end{aligned} \tag{5.92}$$

Now, we examine time interval between k_i and $k \leq k_{i+1}$. First note

$$
\begin{aligned}
V(x(k_i)) &= J(x, \{u^*_{\mathfrak{p}_{r^*}}\}_{k_i}^{k_i+\Delta k-1}) \\
&= \sum_{l=k_i}^{k_i+\Delta k-1} l(\phi_{f_\mathfrak{p}}(l-k_i, x, \{w_\mathfrak{n}\}_{k_i}^{l-1}, \{0,\dots,0\}), u^{r^*}_\mathfrak{p}(l)) \\
&\quad + g(\phi_{f_\mathfrak{p}}(\Delta k, x, \{w_\mathfrak{n}\}_{k_i}^{k_i+\Delta k-1}, \{0,\dots,0\}) \\
&\geq \sum_{l=k_i}^{k} l(\phi_{f_\mathfrak{p}}(l-k_i, x, \{w_\mathfrak{n}\}_{k_i}^{l-1}, \{0,\dots,0\}), u^{r^*}_\mathfrak{p}(l)) \\
&\geq \sum_{l=k_i}^{k} \alpha(|\phi_{f_\mathfrak{p}}(l-k_i, x, \{w_\mathfrak{n}\}_{k_i}^{l-1}, \{0,\dots,0\})|)
\end{aligned}
\tag{5.93}
$$

where $k_i \leq k \leq k_{i+1}-1$. Adding and subtracting term $|\phi_{f_\mathfrak{p}}(l-k_i, x, \{w_\mathfrak{n}\}_{k_i}^{l-1}, \{w_\mathfrak{p}\}_{k_i}^{l-1})|$ to the argument of the α function in latter inequality and using the lower bound from inequality (1.7) yields

$$
\begin{aligned}
V(x(k_i)) \geq \sum_{l=k_i}^{k} \Big(&\frac{1}{2}\alpha(|\phi_{f_\mathfrak{p}}(l-k_i, x, \{w_\mathfrak{n}\}_{k_i}^{l-1}, \{w_\mathfrak{p}\}_{k_i}^{l-1})| \\
&- \frac{1}{2}\alpha(|\phi_{f_\mathfrak{p}}(l-k_i, x, \{w_\mathfrak{n}\}_{k_i}^{l-1}, \{0,\dots,0\})| - |\phi_{f_\mathfrak{p}}(l-k_i, x, \{w_\mathfrak{n}\}_{k_i}^{l-1}, \{w_\mathfrak{p}\}_{k_i}^{l-1})|)\Big) \\
\geq \sum_{l=k_i}^{k} &\underbrace{\frac{1}{2}\alpha}_{\alpha_1}(|\phi_{f_\mathfrak{p}}(l-k_i, x, \{w_\mathfrak{n}\}_{k_i}^{l-1}, \{w_\mathfrak{p}\}_{k_i}^{l-1})|) \\
- \sum_{l=k_i}^{k} &\underbrace{\frac{1}{2}\alpha}_{\alpha_1}(|\phi_{f_\mathfrak{p}}(l-k_i, x, \{w_\mathfrak{n}\}_{k_i}^{l-1}, \{0,\dots,0\})| - |\phi_{f_\mathfrak{p}}(l-k_i, x, \{w_\mathfrak{n}\}_{k_i}^{l-1}, \{w_\mathfrak{p}\}_{k_i}^{l-1})|),
\end{aligned}
\tag{5.94}
$$

which when used in inequality (5.92) results in

$$\sum_{l=k_0}^{k_i-1}\alpha_1(|\phi_{f_\mathfrak{p}}(l-k_0,x,\{w_\mathfrak{n}\}_{k_0}^{l-1},\{w_\mathfrak{p}\}_{k_0}^{l-1})|)+\sum_{l=k_i}^{k}\alpha_1(|\phi_{f_\mathfrak{p}}(l-k_i,x,\{w_\mathfrak{n}\}_{k_i}^{l-1},\{w_\mathfrak{p}\}_{k_i}^{l-1})|)$$
$$\leq \gamma_{\bar{V}}(|x(k_0)|)+\sum_{l=k_0}^{k_i-1}\alpha_2(|w_\mathfrak{p}(l)|)$$
$$+\sum_{l=k_i}^{k}\alpha_1(|\phi_{f_\mathfrak{p}}(l-k_i,x,\{w_\mathfrak{n}\}_{k_i}^{l-1},\{0,\dots,0\})|-|\phi_{f_\mathfrak{p}}(l-k_i,x,\{w_\mathfrak{n}\}_{k_i}^{l-1},\{w_\mathfrak{p}\}_{k_i}^{l-1})|). \tag{5.95}$$

Finally using the inequalities (5.88) and (5.89) results in

$$\sum_{l=k_0}^{k-1}\alpha_1(|\phi_{f_\mathfrak{p}}(l-k_0,x,\{w_\mathfrak{n}\}_{k_0}^{l-1},\{w_\mathfrak{p}\}_{k_0}^{l-1})|)\leq \gamma_{\bar{V}}(|x(k_0)|)+\sum_{l=k_0}^{k_i-1}\alpha_2(|w_\mathfrak{p}(l)|)$$
$$+\sum_{l=k_i}^{k-1}\frac{\mathfrak{h}}{2}\alpha(2^{\mathfrak{h}-1}\bar{\alpha}_\mathfrak{h}(|w_\mathfrak{p}(l)|)) \tag{5.96}$$

which after defining $\alpha_3 := \max\{\alpha_2, \frac{\mathfrak{h}}{2}\cdot \mathrm{Id}\circ\alpha\circ 2^{\mathfrak{h}-1}\cdot\mathrm{Id}\circ\bar{\alpha}_\mathfrak{h}\}\in\mathcal{K}$ provides desired inequality. □

Proof of Lemma 5.4. Let the Assumptions 5.2 and 5.7 be satisfied. According to Assumption 5.7 it follows

$$|\phi_f(1,x,\{w_\mathfrak{n}\}_0^0,\{w_\mathfrak{p}\}_0^0)|\leq \alpha_x(|x(0)|)+\alpha_{\bar{w}_\mathfrak{p}}(|w_\mathfrak{p}(0)|). \tag{5.97}$$

Continuing like this until time first successful transmission k_0 one arrives at

$$|\phi_f(j,x,\{w_\mathfrak{n}\}_0^{j-1},\{w_\mathfrak{p}\}_0^{j-1})|\leq \bar{\alpha}_{x_j}(|x(0)|)+\sum_{l=1}^{j}\bar{\alpha}_{\bar{w}_{\mathfrak{p}_l}}(|w_\mathfrak{p}(j-l)|). \tag{5.98}$$

where $j\in\{0,\dots,k_0\}$ and

$$\left.\begin{aligned}\bar{\alpha}_{x_{l+1}}&=\alpha_x\circ 2\cdot\mathrm{Id}\circ\bar{\alpha}_{x_l}\in\mathcal{K}, &\quad \bar{\alpha}_{x_0}&=\mathrm{Id},\ \bar{\alpha}_{x_1}=\alpha_x,\\ \bar{\alpha}_{\bar{w}_{\mathfrak{p}_{l+1}}}&=\alpha_x\circ 2\cdot\mathrm{Id}\circ\bar{\alpha}_{\bar{w}_{\mathfrak{p}_l}}\in\mathcal{K}, &\quad \bar{\alpha}_{\bar{w}_{\mathfrak{p}_1}}&=\alpha_{\bar{w}_\mathfrak{p}}.\end{aligned}\right\} \tag{5.99}$$

Now, note that according to Assumption 5.2 we have that $k_0\leq\Delta k$, thus, similarly

as in the proof of Lemma 5.2, we can always ensure that $\bar{\alpha}_{x_{l+1}} > \bar{\alpha}_{x_l}$ and $\bar{\alpha}_{\bar{w}_{\mathfrak{p}_{l+1}}} > \bar{\alpha}_{\bar{w}_{\mathfrak{p}_l}}$ yielding

$$\left.\begin{aligned} \hat{\alpha}_{x_{\Delta k}} &= \max_{l\in\{1,\dots,\Delta k\}} \bar{\alpha}_{x_l} \in \mathcal{K}, \\ \hat{\alpha}_{\bar{w}_{\mathfrak{p}_{\Delta k}}} &= \max_{l\in\{1,\dots,\Delta k\}} \bar{\alpha}_{\bar{w}_{\mathfrak{p}_l}} \in \mathcal{K}, \end{aligned}\right\} \tag{5.100}$$

which provides

$$|\phi_f(j, x, \{w_{\mathfrak{n}}\}_0^{j-1}, \{w_{\mathfrak{p}}\}_0^{j-1})| \leq \hat{\alpha}_{x_{\Delta k}}(|x(0)|) + \sum_{l=0}^{j-1} \hat{\alpha}_{\bar{w}_{\mathfrak{p}_{\Delta k}}}(|w_{\mathfrak{p}}(l)|). \tag{5.101}$$

Now, by applying function $\gamma_{\bar{V}}$ from Assumption 5.6 to inequality (5.98) and summing the resulting inequalities from $j = 0$ to $j = k_0$ with the application of third inequality from (1.8) results in

$$\begin{aligned} &\sum_{j=0}^{k_0} \gamma_{\bar{V}}(|\phi_f(j, x, \{w_{\mathfrak{n}}\}_0^{j-1}, \{w_{\mathfrak{p}}\}_0^{j-1})|) \\ &\quad \leq (k_0+1)\gamma_{\bar{V}}(\hat{\alpha}_{x_{\Delta k}}(|x(0)|)) + \sum_{j=0}^{k_0} \gamma_{\bar{V}}\Big(\sum_{l=0}^{j-1} \hat{\alpha}_{\bar{w}_{\mathfrak{p}_{\Delta k}}}(|w_{\mathfrak{p}}(l)|)\Big) \\ &\quad \leq (\Delta k+1)\gamma_{\bar{V}}(\hat{\alpha}_{x_{\Delta k}}(|x(0)|)) + k_0 \sum_{l=0}^{k_0-1} \gamma_{\bar{V}}\Big(2^{k_0-1}\hat{\alpha}_{\bar{w}_{\mathfrak{p}_{\Delta k}}}(|w_{\mathfrak{p}}(l)|)\Big) \\ &\quad \leq \alpha_4(|x(0)|) + \sum_{l=0}^{k_0-1} \Delta k \gamma_{\bar{V}}\Big(2^{\Delta k-1}\hat{\alpha}_{\bar{w}_{\mathfrak{p}_{\Delta k}}}(|w_{\mathfrak{p}}(l)|)\Big) \end{aligned} \tag{5.102}$$

where $\alpha_4 := (\Delta k + 1) \cdot \mathrm{Id} \circ \gamma_{\bar{V}} \circ \hat{\alpha}_{x_{\Delta k}} \in \mathcal{K}$. Finally, noting that due to definitions of the overall NCS state, see (5.16), and Euclidean norm it follows that $|\phi_f(\cdot)| = |(\phi_{f_{\mathfrak{p}}}(\cdot), \phi_{f_{\mathfrak{b}}}(\cdot))| \geq |\phi_{f_{\mathfrak{p}}}(\cdot)|$, thus

$$\begin{aligned} &\sum_{j=0}^{k_0-1} \gamma_{\bar{V}}(|\phi_{f_{\mathfrak{p}}}(j, x, \{w_{\mathfrak{n}}\}_0^{j-1}, \{w_{\mathfrak{p}}\}_0^{j-1})|) + \underbrace{\gamma_{\bar{V}}(|\phi_{f_{\mathfrak{p}}}(k_0, x, \{w_{\mathfrak{n}}\}_0^{k_0-1}, \{w_{\mathfrak{p}}\}_0^{k_0-1})|)}_{\gamma_{\bar{V}}(|x(k_0)|)} \\ &\quad \leq \alpha_4(|x(0)|) + \sum_{l=0}^{k_0-1} \alpha_5(|w_{\mathfrak{p}}(l)|) \end{aligned} \tag{5.103}$$

wehre $\alpha_5 := \Delta k \cdot \mathrm{Id} \circ \gamma_{\bar{V}} \circ 2^{\Delta k-1} \cdot \mathrm{Id} \circ \hat{\alpha}_{\bar{w}_{\mathfrak{p}_{\Delta k}}} \in \mathcal{K}$ which after replacing j with l provides desired inequality. □

Proof of Theorem 5.1. Let the Assumptions 5.1–5.8 be satisfied. Let us consider some arbitrary $x(0) \in \mathbb{X}$, $\{w_{\mathfrak{n}}\}_0^\infty \in \mathcal{S}^{\mathcal{D}}_{\mathbb{K}_{\Delta k}}$ and $\{w_{\mathfrak{p}}\}_0^\infty \in \mathcal{S}^{\mathrm{W}}$. Then, according to Assumption 5.8 and the conclusion of Lemma 5.1 $x(k) = \phi_f(k, x, \{w_{\mathfrak{n}}\}_0^{k-1}, \{w_{\mathfrak{p}}\}_0^{k-1}) \in \mathbb{X}_{\mathfrak{p}_{\mathfrak{f}}} \times \mathbb{U}^{\mathfrak{h}}_{\mathfrak{p}}$ for all $k > 0$. Note that if $0 < k < k_0 \leq \Delta k$, then from Assumption 5.8 (see Eq. 5.35) it follows $x(k) \in \mathbb{X}_{\mathfrak{p}_{\mathfrak{f}}} \times \mathbb{U}^{\mathfrak{h}}_{\mathfrak{p}}$, while otherwise, if $k \in \{k_i, \ldots, k_{i+1}\}$ then according Assumption 5.8 (see Eq. 5.36) and conclusion of Lemma 5.1 $x(k) \in \mathbb{X}_{\mathfrak{p}_{\mathfrak{f}}} \times \mathbb{U}^{\mathfrak{h}}_{\mathfrak{p}}$. Next, we use the conclusion of Lemma 5.4 to bound term $\gamma_{\bar{V}}$ in conclusion of Lemma 5.3 yielding

$$\begin{aligned}\sum_{l=0}^{k_0-1} \gamma_{\bar{V}}(|\phi_{f_{\mathfrak{p}}}(l, x, \{w_{\mathfrak{n}}\}_0^{l-1}, \{w_{\mathfrak{p}}\}_0^{l-1})|) + \sum_{l=k_0}^{k-1} \alpha_1(|\phi_{f_{\mathfrak{p}}}(l-k_0, x, \{w_{\mathfrak{n}}\}_{k_0}^{l-1}, \{w_{\mathfrak{p}}\}_{k_0}^{l-1})|) \\ \leq \alpha_4(|x(0)|) + \sum_{l=0}^{k_0-1} \alpha_5(|w_{\mathfrak{p}}(l)|) + \sum_{l=0}^{k-1} \alpha_3(|w_{\mathfrak{p}}(l)|),\end{aligned} \tag{5.104}$$

where after defining

$$\left.\begin{aligned}\gamma_1 &:= \min\{\gamma_{\bar{V}}, \alpha_1\}, \\ \gamma_2 &:= \alpha_4, \\ \gamma_3 &:= \mathrm{Id}, \\ \gamma_4 &:= \max\{\alpha_3, \alpha_5\},\end{aligned}\right\} \tag{5.105}$$

provides the desired property defined in Definition 5.1. □

Proof of Corollary 5.1. The proof follows the same lines as the proof of Theorem 5.1, thus, we merely present the corresponding functions. These are as follows

$$\left.\begin{aligned}
\alpha_1(s) &:= \tfrac{1}{2}s^2,\\
\alpha_2(s) &:= \max\{\tfrac{\Delta k}{2}, c_V\}2^{4(\Delta k-1)}c_{x_\mathfrak{p}}^{2(\Delta k-1)}c_{w_\mathfrak{p}}^2 s^2,\\
\alpha_3(s) &:= \alpha_2(s),\\
\alpha_4(s) &:= (\Delta k-1)c_{\bar{V}}(2c_x)^{2(\Delta k-1)}c_x^2 s^2,\\
\alpha_5(s) &:= \Delta k c_{\bar{V}} 2^{4(\Delta k-1)}c_{\bar{V}}^{2(\Delta k-1)}c_{\bar{w}_\mathfrak{p}}^2 s^2,\\
\gamma_1(s) &:= \min\{c_{\bar{V}}, \tfrac{1}{2}\}s^2,\\
\gamma_2(s) &:= \alpha_4(s),\\
\gamma_3(s) &:= s,\\
\gamma_4(s) &:= \max\{c_x^{2(\Delta k-1)}c_{w_\mathfrak{p}}^2, \Delta k c_{\bar{V}} c_x^{2(\Delta k-1)}c_{\bar{w}_\mathfrak{p}}^2\}2^{4(\Delta k-1)}s^2.
\end{aligned}\right\} \tag{5.106}$$

□

Proof of Corollary 5.2. Immediately, from the inequality (5.26) it follows

$$\sum_{l=0}^{k}\gamma_1(|\phi_{f_\mathfrak{p}}(l, x, \{w_\mathfrak{n}\}_0^{l-1}, \{0,\dots,0\})|) \le \gamma_2(|x(0)|) \tag{5.107}$$

for any $k \in \mathbb{N}$, $x(0) \in \mathbb{X}_{\mathfrak{p}_\mathfrak{f}} \times \mathbb{U}_\mathfrak{p}^\mathfrak{h}$, $\{w_\mathfrak{n}\}_0^{l-1} \subset \{w_\mathfrak{n}\}_0^\infty \in \mathcal{S}_{\mathbb{K}_{\Delta k}}^{\mathcal{D}}$ and $\{0,\dots,0\} \subset \{w_\mathfrak{p}\}_0^\infty \in \mathcal{S}^{\mathbb{W}}$. Since the sum in latter inequality is bounded by a finite number it follows

$$\lim_{k\to\infty}\gamma_1(|\phi_{f_\mathfrak{p}}(k, x, \{w_\mathfrak{n}\}_0^{k-1}, \{0,\dots,0\})|) = 0 \Rightarrow \lim_{k\to\infty}|\phi_{f_\mathfrak{p}}(k, x, \{w_\mathfrak{n}\}_0^{k-1}, \{0,\dots,0\})| = 0, \tag{5.108}$$

which recovers the main result from [30]. Note that another way to conclude the latter implication is to use dissipation inequality (5.79) as in [30]. □

Proof of Corollary 5.3. The proof of Corollary 5.3 is very similar to the proof of Theorem 5.1. Thus, correspondingly, we will point to the differences rather than repeating the steps already written. First note that inequality (5.51) implies $V(x_\mathfrak{p}) \le \gamma_V|x_\mathfrak{p}|$ for any $x \in \mathbb{X}_{\mathfrak{p}_\mathfrak{f}} \times \mathbb{U}_\mathfrak{p}^\mathfrak{h}$. Furthermore, due to definitions of the overall NCS state, see (5.16), and Euclidean norm it follows $V(x_\mathfrak{p}) \le \gamma_V|x_\mathfrak{p}| \le \gamma_V|x|$ for any $x \in \mathbb{X}_{\mathfrak{p}_\mathfrak{f}} \times \mathbb{U}_\mathfrak{p}^\mathfrak{h}$. As mentioned above, the proof of Corollary 5.3 is very similar to the proof of Theorem 5.1, e.g., it can be seen as case where $\mathrm{R} = 1$. Hence, next we present the changes in the conclusions of

Lemmas used in the proof of Theorem 5.1 (due to slightly different assumptions) which completes the proof. The changes are as follows:

1. Lemma 5.1: Stays the same,

2. Lemma 5.2: The dissipation inequality takes the following form

$$\begin{aligned} &V(x_{\mathfrak{p}}(k_{i+1})) - V(x_{\mathfrak{p}}(k_i)) \\ &\quad \leq -\sum_{j=k_i}^{k_{i+1}-1} \alpha_1(|\phi_{f_{\mathfrak{p}}}(j-k_i, x, \{w_{\mathfrak{n}}\}_{k_i}^{j-1}, \{w_{\mathfrak{p}}\}_{k_i}^{j-1})|) + \sum_{j=k_i}^{k_{i+1}-1} \alpha_2(|w_{\mathfrak{p}}(j)|), \end{aligned} \tag{5.109}$$

   for any $k_i \in \mathbb{K}_{\Delta k}$, $k_{i+1} \in \mathbb{K}_{\Delta k}$, $x_{\mathfrak{p}}(k_i) \in \mathbb{X}_{\mathfrak{p}_{\mathfrak{f}}} \times \mathbb{U}_{\mathfrak{p}}^{\mathfrak{h}}$ and $x_{\mathfrak{p}}(k_{i+1}) \in \mathbb{X}_{\mathfrak{p}_{\mathfrak{f}}} \times \mathbb{U}_{\mathfrak{p}}^{\mathfrak{h}}$,

3. Lemma 5.3: Due to the inequality 5.51 $V(x_{\mathfrak{p}}) \leq \gamma_V |x_{\mathfrak{p}}| \leq \gamma_V |x|$ which introduces the following change

$$\sum_{j=k_0}^{k-1} \alpha_1(|\phi_{f_{\mathfrak{p}}}(j-k_0, x, \{w_{\mathfrak{n}}\}_{k_i}^{j-1}, \{w_{\mathfrak{p}}\}_{k_i}^{j-1})|) \leq \gamma_V(|x(k_0)|) + \sum_{j=k_0}^{k-1} \alpha_3(|w_{\mathfrak{p}}(j)|) \tag{5.110}$$

   for any $x(0) \in \mathbb{X}_{\mathfrak{p}_{\mathfrak{f}}} \times \mathbb{U}_{\mathfrak{p}}^{\mathfrak{h}}$,

4. Lemma 5.4: Finally, in the same manner as in later inequality due to the inequality (5.51) we have the following

$$\gamma_V(|x(k_0)|) \leq \alpha_4(|x(0)|) + \sum_{j=0}^{k_0-1} \alpha_5(|w_{\mathfrak{p}}(j)|) - \sum_{j=0}^{k_0-1} \gamma_V(|\phi_{f_{\mathfrak{p}}}(j, x, \{w_{\mathfrak{n}}\}_0^{j-1}, \{w_{\mathfrak{p}}\}_0^{j-1})|). \tag{5.111}$$

□

This page intentionally left blank.

Chapter 6

Controllability with respect to Scheduling

Controllability is the last control system property we consider. Similarly as properties of stability and robustness, the controllability property also provides a specific insight about the corresponding system which in turn aids in its deeper understanding. However, unlike in previous chapters, here the network only induces scheduling; packet dropouts are left for future work.

In particular we investigate if and how the controllability of a plant can be preserved once a network which imposes the scheduling issue is introduced into the system. Our approach consists of exploiting network processing capabilities and/or the architectural flexibility that NCSs possess. Namely, similarly as in previous chapters, we add buffers to the corresponding NCS architecture, if needed.

Controllability has been addressed in NCS society and some recent references that document this are [52,53,55] and [54]. In [52], the authors consider static scheduling protocols and focus on linear plants with zero-order blocks in front of plant inputs. They use a *lifting technique* to establish sufficient conditions for controllability (and observability) of the corresponding NCS. In [55], the authors consider a NCS with a SISO linear plant and investigate the effects of the so called *blind* periods in communication on controllability. Finally, the last two, relatively-related, references to our results are [53] and [54] in which the authors consider Multi-Hop Control Networks with MIMO and SISO linear plants and present conditions for controllability, respectively.

The contributions of this chapter are as follows. First we define a model of the corresponding NCS which is interesting and novel in its own right. More precisely, the interest

and novelty associated with the NCS model come from the way we capture the evolution of system variables. Namely, we use two indices to capture the evolution of system variables. In particular, the first index corresponds to discrete time while the second index represents a counter which keeps track of transmission and processing instants. Moreover, transitioning from this model back to a model that uses only discrete time is quite easy and we do that to establish the corresponding controllability results. To establish our controllability results we rely on the assumption that the corresponding network has appropriate processing capabilities (we call these *additional processing capabilities* (APCs)) which enable us to manipulate the corresponding communication protocols. Using that we establish two controllability results for NCSs which have general nonlinear plants. These results are rather general. On the other hand, for NCSs which have linear plants we first extend a controllability result form [52] by showing the existence of an *admissible communication sequence* so that the corresponding controllability result holds. Then we use this extension and the resulting NCS architecture to establish our controllability result. In [55], even though the authors do not consider explicitly the scheduling issue (since the plant is SISO) the corresponding result can be viewed as a special case of our result when there is no scheduling.

This chapter is organized as follows. In Section 6.1 we present the corresponding model while in Section 6.2 we present the analysis and collect all results. Finally, in Section 6.3 we provide the corresponding proofs.

6.1 NCS architecture

Similarly as in the previous chapters, we will present the considered NCS architecture, see Fig. 6.1, by presenting each part separately in a more detail. However, before we depart to the specifics of the corresponding NCS architecture let us spend few words on assumptions we use that make the resulting model interesting and novel. Namely, we assume that the transmission of the data through the network is *instantaneous* as well as the processing in devices $\mu_{\mathfrak{p}}$ and $\mu_{\mathfrak{c}}$. The corresponding effects of *instantaneous* transmission and processing can be described as jumps in the dynamical model of the NCS. This

motivates us to adopt a discrete-time approach as documented in [47, 48] and use two indices to capture time evolution in system variables. Moreover, notice that our model can be seen to some extent as a discretization of a model from [47, 48]. More precisely, the first index refers to discrete time while the second index is a counter which refers to transmission *plus* processing instants. The indices do not evolve independently. Both are incremented alternatively and can only be incremented by 1. For example, consider a sequence of index pairs

$$\{(0,0),(0,1),(1,1),(1,2),\dots\}.$$

Whenever the second index is larger than the first, it indicates that transmission and processing has just occurred. Using this convention we define the corresponding NCS model. We believe that the model is novel and useful in its own right and maybe useful for other problems as well. Let us now proceed with the presentation of the parts that form the considered NCS architecture.

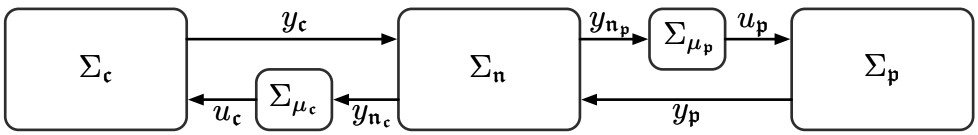


Figure 6.1: A NCS architecture for addressing controllability.

6.1.1 Plant

The plant model is represented as

$$\Sigma_{\mathfrak{p}} : \begin{cases} x_{\mathfrak{p}}(i+1,i+2) = f_{\mathfrak{p}}(x_{\mathfrak{p}}(i,i+1), u_{\mathfrak{p}}(i,i+1)), \\ y_{\mathfrak{p}}(i,i+1) = h_{\mathfrak{p}}(x_{\mathfrak{p}}(i,i+1)), \end{cases} \tag{6.1}$$

where $x_{\mathfrak{p}}(i,i+1) \in \mathbb{R}^{n_{\mathfrak{p}}}$ is the plant state, $u_{\mathfrak{p}}(i,i+1) \in \mathbb{R}^{m_{\mathfrak{p}}}$ is the plant input and $y(i,i+1) \in \mathbb{R}^{p_{\mathfrak{p}}}$ is the plant output after transmission and processing have occurred. The mappings $f_{\mathfrak{p}} : \mathbb{R}^{n_{\mathfrak{p}}} \times \mathbb{R}^{m_{\mathfrak{p}}} \to \mathbb{R}^{n_{\mathfrak{p}}}$ and $h_{\mathfrak{p}} : \mathbb{R}^{n_{\mathfrak{p}}} \to \mathbb{R}^{p_{\mathfrak{p}}}$ are assumed nonlinear where $n_{\mathfrak{p}}, m_{\mathfrak{p}}$ and $p_{\mathfrak{p}}$ are positive integers. Moreover, we assume that during transmission and

processing, captured with our notation with $(i,i) \rightarrow (i,i+1)$, the plant state does not change, i.e.,

$$x_{\mathfrak{p}}(i,i+1) = x_{\mathfrak{p}}(i,i),\ \forall i \in \mathbb{N}. \tag{6.2}$$

Finally, as in the previous chapters, the plant control input is *partitioned* according to

$$u_{\mathfrak{p}} := (u_{\mathfrak{p}_1}, \ldots, u_{\mathfrak{p}_{\mathrm{R}}}) \tag{6.3}$$

where

$$\left.\begin{array}{c} u_{\mathfrak{p}_r} \in \mathbb{R}^{m_{\mathfrak{p}_r}},\ m_{\mathfrak{p}_r} \in \mathbb{N},\ \forall r \in \mathcal{R}, \\ \sum_{r=1}^{\mathrm{R}} m_{\mathfrak{p}_r} = m_{\mathfrak{p}}, \\ \mathcal{R} := \{1, \ldots, \mathrm{R}\}. \end{array}\right\} \tag{6.4}$$

6.1.2 Network protocols

The considered network is assumed to impose *only* the scheduling issue. However, to simplify analysis, we focus only on scheduling of plant inputs, i.e., we place a network only between controller output and plant inputs. Consequently, there is no scheduling of plant outputs. In this case, the corresponding NCS architecture can be depicted as in Fig. 6.2.

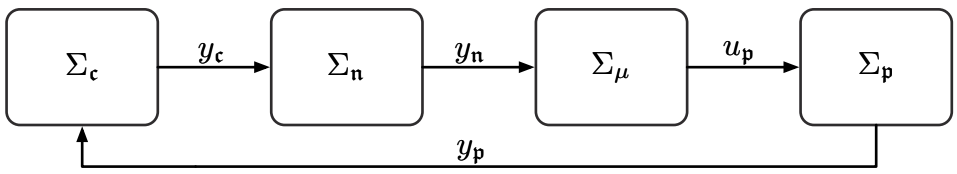


Figure 6.2: A NCS architecture for addressing controllability - a simplified representation.

In order to address this scheduling issues we characterize the network with a protocol which governs the medium access of each node; e.g., see [47, 48, 172]. We focus on dynamic protocols which compare the data addressed to a node with network state, e.g.,

the corresponding buffer contents before transmission. In our case, data to be transmitted are controller outputs and the quantity used by the protocol is

$$e(i,i) := x_{\mathfrak{n}}(i,i) - y_{\mathfrak{c}}(i,i). \tag{6.5}$$

In the latter equation, $y_{\mathfrak{c}}(i,i)$ is the controller output and $x_{\mathfrak{n}}(i,i)$ is the network state defined as

$$x_{\mathfrak{n}}(i,i) := f_{\mathfrak{n}}(y_{\mathfrak{c}}(i,i), y_{\mathfrak{p}}(i,i)) \tag{6.6}$$

where $f_{\mathfrak{n}} : \mathbb{R}^{p_{\mathfrak{c}}} \times \mathbb{R}^{p_{\mathfrak{p}}} \to \mathbb{R}^{p_{\mathfrak{c}}}$. Mapping $f_{\mathfrak{n}}$ captures the fact that some networks can have *additional processing capabilities* (APC). For instance, via appropriate $f_{\mathfrak{n}}$ one can manipulate which node will be picked, i.e., we can manipulate the protocol; this will be shown later in the chapter.

Further, a protocol can be described by diagonal matrices $\Psi(\cdot,\cdot)$ which contain zeros and ones on its diagonal; see equations (14), (16) and (17) in [47] for different types of protocols. In this manuscript we will focus on the so-called Try Once Discard protocol (TOD).

Definition 6.1 (TOD protocol, [48])**.** *Suppose that there are* $\mathrm{R} \in \mathbb{Z}_{\geq 2}$ *nodes competing for access to the network. Correspondingly, the error vector is partitioned as* $e = (e_1, \ldots, e_{\mathrm{R}})$*. The node* $r \in \mathcal{R}$ *with the greatest weighted error at instant* $(i,i), i \in \mathbb{N}_0$ *will be granted access. (It is assumed that the weights are already incorporated into the model.) If a data packet fails to win access to the network, it is discarded and new data is used at the next transmission time. If two or more nodes have equal priority, a pre-specified ordering of the nodes is used to resolve the collision. More precisely, the diagonal matrix* $\Psi(\cdot,\cdot)$ *is given as*

$$\Psi\big((i,i), e(i,i)\big) = diag\big(\psi_1\big(e(i,i)\big) I_{m_{\mathfrak{p}_1}}, \ldots, \psi_{\mathrm{R}}\big(e(i,i)\big) I_{m_{\mathfrak{p}_{\mathrm{R}}}}\big) \tag{6.7}$$

where $i \in \mathbb{N}_0$ *and* $\sum_{r=1}^{\mathrm{R}} m_{\mathfrak{p}_r} = m_{\mathfrak{p}}$ *and where*

$$\psi_r\big(e(i,i)\big) = \begin{cases} 1, & \textit{if } r = \min\big(\arg\max_{r\in\mathcal{R}} |e_r(i,i)|\big), \\ 0, & \textit{otherwise}. \end{cases} \tag{6.8}$$

for each $r \in \mathcal{R}$. □

6.1.3 Processing devices

As stated in the introduction of this chapter, in order to mitigate the undesirable effects of the network one might exploit the flexibility of a NCS architecture and resort to distributed computing. In our case, this translates to the introduction of extra devices which have simple processing capabilities and memory. We concentrate on two types.

One type of a device will just apply the received value for the addressed node and zeros to the remaining nodes. Correspondingly, the output of this device is mathematically described as

$$y_{\mu_{\mathfrak{p}}^s}(i,i+1) = h_{\mu_{\mathfrak{p}}^s}\big(y_{\mathfrak{n}}(i,i+1)\big), \tag{6.9}$$

for each $i \in \mathbb{N}_0$, where s alludes to static since this device does not need any memory, e.g., see Fig. 6.3 for a conceptual abstraction of a static devices for a plant with three inputs; e.g., it behaves as a switch.

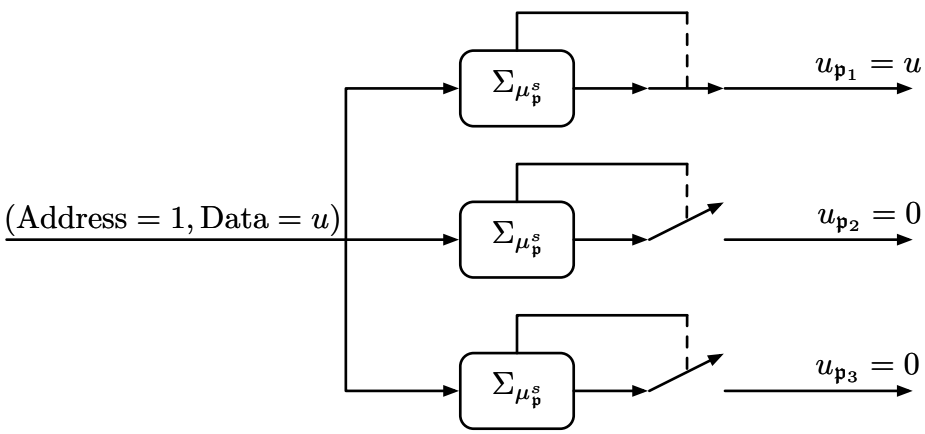


Figure 6.3: Conceptual abstraction of static devices for a plant with three inputs.

The other type of device will be dynamic. Its output is defined as follows:

$$y_{\mu_{\mathfrak{p}}^d}(i,i+1) = h_{\mu_{\mathfrak{p}}^d}\big(y_{\mathfrak{n}}(i,i+1), x_{\mu_{\mathfrak{p}}^d}(i-1,i)\big) \tag{6.10}$$

for all $i \in \mathbb{N}$, where d alludes to dynamic.

Next, we proceed with the necessary preliminaries needed for stating the NCS architectures determined by the devices used. First, we assume that states in $\mu_{\mathfrak{p}}^d$ hold their values until new data arrives, namely

$$x_{\mu_{\mathfrak{p}}^d}(i,i+1) = x_{\mu_{\mathfrak{p}}^d}(i+1,i+1) \tag{6.11}$$

for each $i \in \mathbb{N}_0$. Similarly, we assume that the value of controller's output $y_{\mathfrak{c}}$ does not change during transmission and processing, that is

$$y_{\mathfrak{c}}(i,i) = y_{\mathfrak{c}}(i,i+1) \tag{6.12}$$

for each $i \in \mathbb{N}_0$.

Remark: Note that the equation (6.11) could be replaced with more complex processing; e.g., see [31, 61, 83, 173, 174]. □

Now, note that the network output can be defined as

$$y_{\mathfrak{n}}(i,i+1) = \Psi\big(i, e(i,i)\big) y_{\mathfrak{c}}(i,i) \tag{6.13}$$

for each $i \in \mathbb{N}_0$. Using the latter equation, the output of the processing unit $\mu_{\mathfrak{p}}^d$ is then given as

$$y_{\mu_{\mathfrak{p}}^d}(i,i+1) = \Psi\big(i, e(i,i)\big) y_{\mathfrak{c}}(i,i) + \big(I - \Psi\big(i, e(i,i)\big)\big) x_{\mu_{\mathfrak{p}}^d}(i,i) \tag{6.14}$$

for each $i \in \mathbb{N}_0$. Note also that according to (6.9) we have

$$y_{\mu_{\mathfrak{p}}^s}(i,i+1) = h_{\mu_{\mathfrak{p}}^s}\big(y_{\mathfrak{n}}(i,i+1)\big). \tag{6.15}$$

As depicted in Fig. 6.2, we have that $u_{\mathfrak{p}}(\cdot,\cdot) = y_{\mu_{\mathfrak{p}}^*}(\cdot,\cdot)$ where $* \in \{s,d\}$. Moreover, for the

case when $\mu_{\mathfrak{p}}^{d}$ is used $x_{\mu_{\mathfrak{p}}^{d}}(\cdot,\cdot) = y_{\mu_{\mathfrak{p}}^{d}}(\cdot,\cdot) = u_{\mathfrak{p}}(\cdot,\cdot)$. Now, given the above, the equation (6.14) becomes

$$u_{\mathfrak{p}}(i,i+1) = \Psi\big(i,e(i,i)\big)y_{\mathfrak{c}}(i,i+1) + \big(I - \Psi\big(i,e(i,i)\big)\big)x_{\mu_{\mathfrak{p}}^{d}}(i-1,i) \tag{6.16}$$

whereas, assuming $x_{\mathfrak{n}}(\cdot) = x_{\mu_{\mathfrak{p}}^{d}}(\cdot)$, $e(i,i)$ (see (6.5)) satisfies

$$e(i,i) = x_{\mathfrak{n}}(i-1,i) - y_{\mathfrak{c}}(i,i+1). \tag{6.17}$$

6.1.4 NCS architecture with dynamic devices

Manipulating equations (6.1), (6.2), (6.10)–(6.14), (6.16) and (6.17) yields

$$\left.\begin{aligned} x_{\mathfrak{p}}(i+1,i+2) &= f_{\mathfrak{p}}\big(x_{\mathfrak{p}}(i,i+1), \big(I - \Psi\big(i,e(i,i)\big)\big)e(i,i) + y_{\mathfrak{c}}(i,i)\big), \\ e(i,i+1) &= \big(I - \Psi\big(i,e(i,i)\big)\big)e(i,i), \\ e(i,i) &= e(i-1,i) + y_{\mathfrak{c}}(i-1,i) - y_{\mathfrak{c}}(i,i+1). \end{aligned}\right\} \tag{6.18}$$

In order to simplify the upcoming analysis, we will write k for (i,i) and $k+1$ for $(i+1,i+1)$, see Fig. 6.4.

Using the latter convention, gives us the following model

$$\left.\begin{aligned} x_{\mathfrak{p}}(k+1) &= f_{\mathfrak{p}}\big(x_{\mathfrak{p}}(k), \big(I - \Psi\big(k,e(k)\big)\big)e(k) + y_{\mathfrak{c}}(k)\big), \\ e(k+1) &= \big(I - \Psi\big(k,e(k)\big)\big)e(k) + y_{\mathfrak{c}}(k) - y_{\mathfrak{c}}(k+1). \end{aligned}\right\} \tag{6.19}$$

6.1.5 NCS architecture with static devices

Simple manipulations of equations (6.1), (6.2), (6.15) and (6.17) with assumptions that $x_{\mathfrak{n}}(\cdot,\cdot) = y_{\mu_{\mathfrak{p}}^{s}}(\cdot,\cdot)$ and $x_{\mathfrak{n}}(i,i+1) = x_{\mathfrak{n}}(i+1,i+1)$ yields

$$\left.\begin{aligned} x_{\mathfrak{p}}(i+1,i+2) &= f_{\mathfrak{p}}\big(x_{\mathfrak{p}}(i,i+1), y_{\mu_{\mathfrak{p}}^{s}}(i,i+1)\big), \\ e(i,i+1) &= \big(I - \Psi\big(i,e(i,i)\big)\big)e(i,i), \\ e(i+1,i+1) &= y_{\mu_{\mathfrak{p}}^{s}}(i-1,i) - y_{\mathfrak{c}}(i+1,i+2). \end{aligned}\right\} \tag{6.20}$$

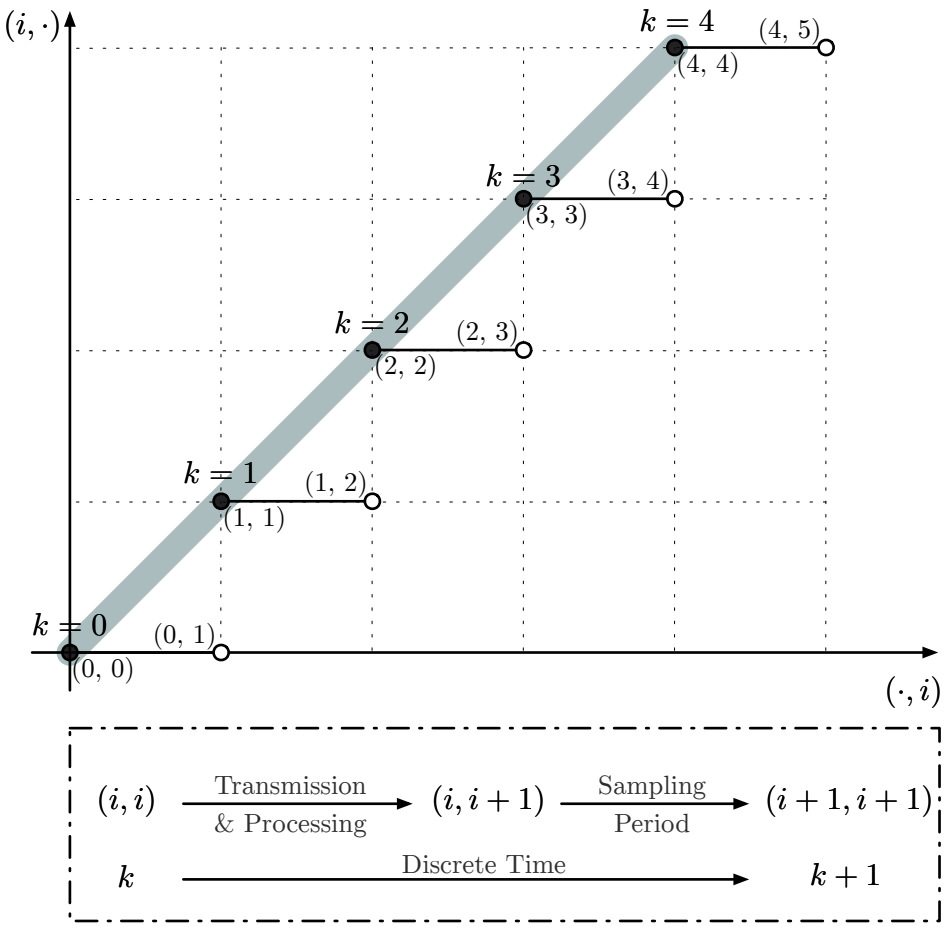


Figure 6.4: A convention for "transitioning" form hybrid time to discrete time.

Using again the same convention for "turning" hybrid time into a discrete time (see Fig. 6.4) results in

$$\left.\begin{aligned} x_{\mathfrak{p}}(k+1) &= f_{\mathfrak{p}}\big(x_{\mathfrak{p}}(k), y_{\mu^s_{\mathfrak{p}}}(k)\big), \\ e(k+1) &= y_{\mu^s_{\mathfrak{p}}}(k) - y_{\mathfrak{c}}(k+1). \end{aligned}\right\} \tag{6.21}$$

6.2 Results

The question we are interested in answering is how the scheduling issue (induced by the network) affects the controllability of the plant. Let us begin by adopting the following

notion of controllability.

Definition 6.2 (Controllability)**.** *The system*

$$\Sigma_x : x(k+1) = f\big(x(k), u(k)\big),\ k \in \mathbb{N}_0 \tag{6.22}$$

where $x \in \mathbb{R}^n$ is the state, $u \in \mathbb{R}^m$ is the input, $n \in \mathbb{N}$ and $m \in \mathbb{N}$, is said to be asymptotically controllable to the origin*, if there exists $\beta \in \mathcal{KL}$, such that for any initial condition x, there exists a nonempty* set *of semi-infinite length control sequences $\mathcal{S}^{\mathbb{R}^m}(x)$ such that for all $u_0^\infty = \big\{u(0), u(1), \dots\big\} \in \mathcal{S}^{\mathbb{R}^m}(x)$ the following inequality holds*

$$\big|\phi(k, x, u_0^\infty)\big| \leq \beta\big(|x|, k\big),\ \forall k \in \mathbb{N}_0. \tag{6.23}$$

Moreover, if $\beta\big(|x|, k\big)$ can be chosen as $\beta\big(|x|, k\big) = M \exp(-k\lambda)|x|$ for some $(M, \lambda) \in \mathbb{R}_{>0} \times \mathbb{R}_{>0}$, then Σ_x is said to be exponentially controllable to the origin. □

Applying the latter definition to the plant makes explicit the fact that for a given $x_{\mathfrak{p}}$ there may exist more than one control sequence which drives the plant state to the origin satisfying the desired bound. This allows us to study the controllability property of the same plant when the corresponding inputs are accessed through a network. The study is done by examining whether for any $x_{\mathfrak{p}} \in \mathbb{R}^{n_{\mathfrak{p}}}$, the network allows for *realization* of at least one sequence in $\mathcal{S}^{\mathbb{R}^m}(x_{\mathfrak{p}})$.

By realization we mean on some sort of pre-processing ($\Sigma_{\mu_{\text{pre}}}$) and/or post-processing ($\Sigma_{\mu_{\text{post}}}$) and/or exploitation of the network capabilities to manipulate the corresponding control values so that they remain the same once transmitted via network and received by the plant input; see Fig. 6.5 .

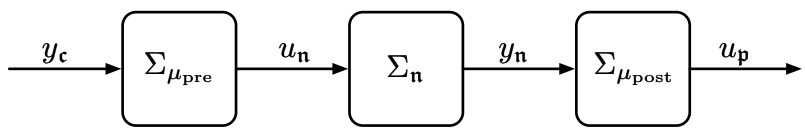


Figure 6.5: A block diagram abstraction of the concept of the realization of control sequences over a network.

Remark: Recall that we begin this chapter with the NCS architecture which communicates all signals through a network, see Fig. 6.1. However, due to our interest if for a given $x_{\mathfrak{p}}$ the network allows for *realization* of at least one sequence in $\mathcal{S}^{\mathbb{R}^m}(x_{\mathfrak{p}})$, we do not send plant outputs through a network since in that case what the controller receives might be different from the real $x_{\mathfrak{p}}$ (due to scheduling). This results in a simpler NCS, see Fig. 6.2. □

We proceed by presenting how to mitigate the scheduling issue effects on the controllability of the nonlinear and linear plants, separately.

6.2.1 Controllability: nonlinear plants

First, with $\mathcal{S}_{\mathrm{S}}^{\mathbb{R}^m}(x_{\mathfrak{p}}) \subset \mathcal{S}^{\mathbb{R}^m}(x_{\mathfrak{p}})$ we will denote a set that consists of semi-infinite length control sequences from $\mathcal{S}^{\mathbb{R}^m}(x_{\mathfrak{p}})$ which can be realized by exploiting network processing capabilities, network protocol and an appropriate processing unit $\mu_{\mathfrak{p}}^*$; the notation $\mathcal{S}_{\mathrm{S}}^{\mathbb{R}^m}(x_{\mathfrak{p}})$ alludes to a subset due to the *scheduling*.

Note that unlike static protocols, dynamic protocols need not have a predefined schedule. In fact, values to be sent through a network have to satisfy a certain criterion defined by the protocol which is usually fixed, see (6.8).

Now, let us assume that the network allows for *manipulating* this criterion. This manipulation we will call *additional processing capability* (APC); what it means exactly we will define in the sequel. Important to note is that having the ability to manipulate the criterion, e.g., see (6.8), enables us to trick the corresponding protocol and force which node gets the access. Note that having ability to force (choose) which node gets the access might be just enough to realize some sequences and preserve controllability. In fact, on a high level, the former is exactly what we do. However, in order to be more precise, we proceed with the needed definitions. We start by defining rigorously the ability to manipulate the criterion used by the protocol do provide an access to a node, i.e., we define APC next.

Definition 6.3 (APC)**.** *Suppose that there are* $\mathrm{R} \in \mathbb{Z}_{\geq 2}$ *nodes competing for access to the network. Correspondingly, the network state vector is partitioned as* $x_{\mathfrak{n}} = (x_{\mathfrak{n}_1}, \ldots, x_{\mathfrak{n}_{\mathrm{R}}})$*. Then,* additional

processing capability (APC) *means that there exist* $\delta > 0$ *such that provided* $\bar{r} \in \{1, \ldots, \mathrm{R}\}$, $x_{\mathfrak{n}_r} := u_{\mathfrak{p}_r} + \delta, \forall r \in \{1, \ldots, \mathrm{R}\}$, $r \neq \bar{r}$ *and* $x_{\mathfrak{n}_{\bar{r}}} := u_{\mathfrak{p}_{\bar{r}}}$. □

Note that in Definition 6.3 we do not explain how $\bar{r}$ is provided; this is done in the sequel. Also note that by adding $\delta > 0$ to all nodes except to $\bar{r}$ we effectively are tricking the protocol to choose node $\bar{r}$, see (6.5) and (6.8) for the case of TOD protocol. In fact, next we provide a lemma that captures precisely what tricking TOD means.

Lemma 6.1 (Tricking TOD)**.** *Consider any* $u_{\mathfrak{p}}(k) \in \{u_{\mathfrak{p}}\}_0^\infty \in \mathcal{S}_{\mathrm{S}}^{\mathbb{R}^m}(x_{\mathfrak{p}})$, $k \in \mathbb{N}_0$ *for some* $x_{\mathfrak{p}} \in \mathbb{R}^{n_{\mathfrak{p}}}$. *Let the network be governed by the TOD protocol and let it have* APC. *Then, provided* $\bar{r} \in \{1, \ldots, \mathrm{R}\}$, $u_{\mathfrak{p}_{\bar{r}}}(k)$ *is chosen for transmission.* □

As promised above, next we discuss how $\bar{r} \in \{1, \ldots, \mathrm{R}\}$ is provided. First, we concentrate on a special subset of $\mathcal{S}_{\mathrm{S}}^{\mathbb{R}^m}(x_{\mathfrak{p}})$ which is additionally accompanied with the appropriate processing unit $\mu_{\mathfrak{p}}^*$. More precisely, let us consider set $\mathcal{S}_{\mathrm{S}_0}^{\mathbb{R}^m}(x_{\mathfrak{p}}) \subset \mathcal{S}_{\mathrm{S}}^{\mathbb{R}^m}(x_{\mathfrak{p}})$ which consists of control sequences where at each time instant a member of the corresponding sequence has at most one nonzero element. Namely, let $\{u^0\}_0^\infty = \{u^0(0), u^0(1), \ldots, u^0(k), \ldots\} \in \mathcal{S}_{\mathrm{S}_0}^{\mathbb{R}^m}(x_{\mathfrak{p}})$, $k \in \mathbb{N}_0$. Then, for each k it follows $u^0(k) = \big(0, \ldots, 0, u_r^0(k), 0, \ldots, 0\big)$, $u_r^0(k) \in \mathbb{R}^{m_r}$, where $r \in \{1, \ldots, \mathrm{R}\}$. Furthermore, let the processing unit be $\mu_{\mathfrak{p}}^s$; see (6.9).

Now, for the sake of simplicity and precision, we modify Definition 6.3 with respect to set $\mathcal{S}_{\mathrm{S}_0}^{\mathbb{R}^m}(x_{\mathfrak{p}})$. Namely, we extend the corresponding definition to additionally extract the index of nonzero element or to provide any index if all elements are equal to zero; see also Fig. 6.6.

Definition 6.4 (APC$_0$)**.** *Consider* $\mathcal{S}_{\mathrm{S}_0}^{\mathbb{R}^m}(x_{\mathfrak{p}}) \subset \mathcal{S}_{\mathrm{S}}^{\mathbb{R}^m}(x_{\mathfrak{p}})$ *and suppose that there are* $\mathrm{R} \in \mathbb{Z}_{\geq 2}$ *nodes competing for network access. Correspondingly, the network state vector is partitioned as* $x_{\mathfrak{n}} = (x_{\mathfrak{n}_1}, \ldots, x_{\mathfrak{n}_{\mathrm{R}}})$. *Then,* APC$_0$ *means that there exist* $\delta > 0$ *and* $\bar{r} \in \{1, \ldots, \mathrm{R}\}$ *is the index of the nonzero element of the corresponding control vector at time k or it is any index otherwise, and* $x_{\mathfrak{n}_r}(k) := u_{\mathfrak{p}_r}^0(k) + \delta, \forall r \in \{1, \ldots, \mathrm{R}\}$, $r \neq \bar{r}$ *and* $x_{\mathfrak{n}_{\bar{r}}} := u_{\mathfrak{p}_{\bar{r}}}^0$. □

It follows that with Definition 6.4 and processing unit $\mu_{\mathfrak{p}}^s$, we can state the following lemma related to realization of sequences from set $\mathcal{S}_{\mathrm{S}_0}^{\mathbb{R}^m}(x_{\mathfrak{p}})$, i.e., $u_{\mathfrak{p}}(k) = u^0(k), \forall k \in \mathbb{N}_0$; see (6.9).

Lemma 6.2 (Realizing sequences from $\mathcal{S}_{\mathrm{S}_0}^{\mathbb{R}^m}(x_{\mathfrak{p}})$)**.** *Consider set $\mathcal{S}_{\mathrm{S}_0}^{\mathbb{R}^m}(x_{\mathfrak{p}}) \subset \mathcal{S}_{\mathrm{S}}^{\mathbb{R}^m}(x_{\mathfrak{p}})$ and the processing unit $\mu_{\mathfrak{p}}^s$. Let the network be governed by TOD protocol and let it have* APC_0*. Then, the sequences from $\mathcal{S}_{\mathrm{S}_0}^{\mathbb{R}^m}(x_{\mathfrak{p}})$ are realizable .* □

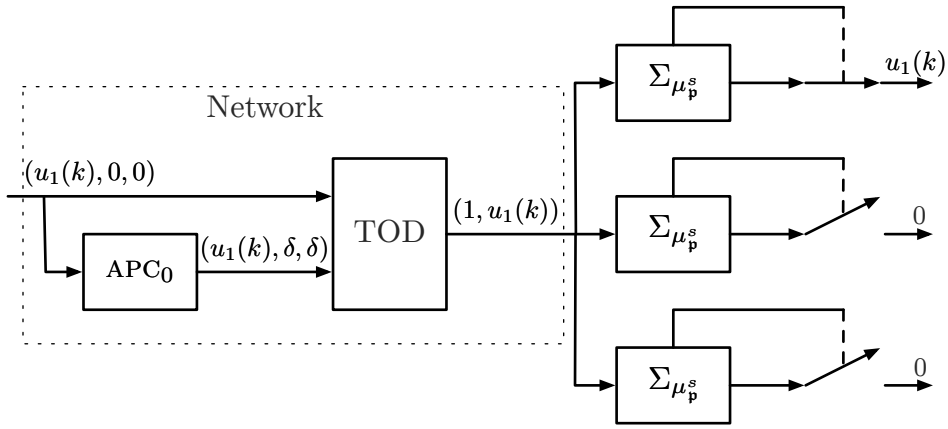


Figure 6.6: Illustration of tricking TOD with APC_0 for a plant with three inputs.

We proceed by focusing on the $\mathcal{S}_{\mathrm{S}_\delta}^{\mathbb{R}^m}(x_{\mathfrak{p}}) \subset \mathcal{S}_{\mathrm{S}}^{\mathbb{R}^m}(x_{\mathfrak{p}})$ which consists of control sequences where at each two consecutive time instances the corresponding members of the corresponding sequence differ in at most one element.

Namely, let $\{u^\delta\}_0^\infty = \{u^\delta(0), u^\delta(1), \ldots, u^\delta(k), \ldots\} \in \mathcal{S}_{\mathrm{S}_\delta}^{\mathbb{R}^m}(x_{\mathfrak{p}})$, $k \in \mathbb{N}_0$, then for each k it follows $\left|u^\delta(k) - u^\delta(k+1)\right| = \big(0, \ldots, 0, |u_r^\delta(k) - u_r^\delta(k+1)|, 0, \ldots, 0\big) = \big(0, \ldots, 0, \delta_r(k), 0, \ldots, 0\big)$, $r \in \{1, \ldots, \mathrm{R}\}$, $\delta_r(k) \geq 0$. Furthermore, let the processing unit be $\mu_{\mathfrak{p}}^d$, see (6.14).

Again, we modify Definition 6.3 but now with respect to $\mathcal{S}_{\mathrm{S}_\delta}^{\mathbb{R}^m}(x_{\mathfrak{p}})$. Similarly as above, we extend the corresponding definition to additionally extract the index of differing element or to provide any index if all elements are the same.

Definition 6.5 (APC_δ)**.** *Let the processing unit $\mu_{\mathfrak{p}}^d$ be used. Consider set $\mathcal{S}_{\mathrm{S}_\delta}^{\mathbb{R}^m}(x_{\mathfrak{p}})$ and suppose that there are $\mathrm{R} \in \mathbb{Z}_{\geq 2}$ nodes competing for access to the network. The network state vector is partitioned as $x_{\mathfrak{n}} = (x_{\mathfrak{n}_1}, \ldots, x_{\mathfrak{n}_\mathrm{R}})$. Then,* APC_δ *means that there exist $\delta > 0$ and by comparing $u^\delta(k)$ and $x_{\mathfrak{n}}(k-1) = x_{\mu_{\mathfrak{p}}^d}(k-1), k \in \mathbb{N}$, index $\bar{r} \in \{1, \ldots, \mathrm{R}\}$ is the index of the differing element or it is any index if $u^\delta(k) = x_{\mathfrak{n}}(k)$. Finally, $x_{\mathfrak{n}_r}(k) := u_r^\delta(k) + \delta, \forall r \in \{1, \ldots, \mathrm{R}\}$, $r \neq \bar{r}$ and $x_{\mathfrak{n}_{\bar{r}}} := u_{\mathfrak{p}_{\bar{r}}}^\delta$.* □

Remark: It is important to note that it is possible that $u^\delta(0)$ and $x_{\mu_\mathfrak{p}^d}(0)$ differ in more than one element. Thus, let us take a short detour to discuss how realizations of sequences from the set $\mathcal{S}_{\mathrm{S}_\delta}^{\mathbb{R}^m}(x_\mathfrak{p})$ impose restrictions. If $x_{\mu_\mathfrak{p}^d}(0)$ and $u^\delta(0)$ differ in more than one element, then determination of an index for element to be sent would have to be specified by some rule. However, by the time $u_\mathfrak{p}(k) = u^\delta(k)$, the plant state might diverge from the trajectory that leads to the origin. Hence, without imposing constraint that at time when $u_\mathfrak{p}(k) = u^\delta(k)$ we have $\phi_{f_\mathfrak{p}}(k, x_\mathfrak{p}, \{u^\delta\}_0^\infty) = \phi_{f_\mathfrak{p}}(k, x_\mathfrak{p}, \{\tilde{u}^\delta\}_0^\infty)$,[1] where $k \geq m$, we cannot guarantee controllability. □

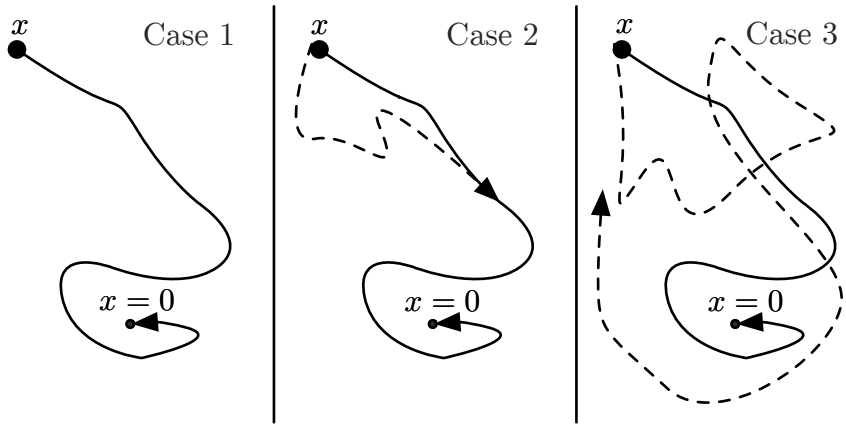


Figure 6.7: Case 1: $u^\delta(0) = x_{\mu_\mathfrak{p}^d}(0)$; Case 2: $u^\delta(0) \neq x_{\mu_\mathfrak{p}^d}(0)$ but when $u_\mathfrak{p}(k) = u^\delta(k)$ we have $\phi_{f_\mathfrak{p}}(k, x_\mathfrak{p}, \{u^\delta\}_0^\infty) = \phi_{f_\mathfrak{p}}(k, x_\mathfrak{p}, \{\tilde{u}^\delta\}_0^\infty)$ where $k \geq m$; Case 3: $u^\delta(0) \neq x_{\mu_\mathfrak{p}^d}(0)$ but when $u_\mathfrak{p}(k) = u^\delta(k)$ we have $\phi_{f_\mathfrak{p}}(k, x_\mathfrak{p}, \{u^\delta\}_0^\infty) \neq \phi_{f_\mathfrak{p}}(k, x_\mathfrak{p}, \{\tilde{u}^\delta\}_0^\infty)$

Now we are ready to state the following lemma which is related to realization of sequences from $\mathcal{S}_{\mathrm{S}_\delta}^{\mathbb{R}^m}(x_\mathfrak{p})$, i.e., $u_\mathfrak{p}(k) = u^\delta(k), \forall k \in \mathbb{N}_0$.

Lemma 6.3 (Realizing sequences from $\mathcal{S}_{\mathrm{S}_\delta}^{\mathbb{R}^m}(x_\mathfrak{p})$)**.** *Consider set $\mathcal{S}_{\mathrm{S}_\delta}^{\mathbb{R}^m}(x_\mathfrak{p})$ and the processing unit $\mu_\mathfrak{p}^d$. Let the corresponding network be governed by TOD protocol and let it have* APC_δ*. If $x_{\mu_\mathfrak{p}^d}(0)$ and $u^\delta(0) \in \{u^\delta\}_0^\infty \in \mathcal{S}_{\mathrm{S}_\delta}^{\mathbb{R}^m}(x_\mathfrak{p})$ differ in at most one element, then $\{u^\delta\}_0^\infty$ is realizable.*□

The results above are rather general. Moreover, due to its requirement, Lemma 6.3 is more restrictive than Lemma 6.2.

[1] $\{\tilde{u}^\delta\}_0^\infty = \{x_{\mu_\mathfrak{p}^d}(0), x_{\mu_\mathfrak{p}^d}(1), \ldots, x_{\mu_\mathfrak{p}^d}(k-1), u^\delta(k), u^\delta(k+1), \ldots\}$

6.2.2 Controllability: linear plants

In this section we consider a special case of (6.1), namely

$$\Sigma_{\mathfrak{p}} : \begin{cases} x_{\mathfrak{p}}(k+1) = Ax_{\mathfrak{p}}(k) + Bu_{\mathfrak{p}}(k), \\ y_{\mathfrak{p}}(k) = Cx_{\mathfrak{p}}(k) \end{cases} \tag{6.24}$$

where $A \in \mathbb{R}^{n_{\mathfrak{p}} \times n_{\mathfrak{p}}}$, $B \in \mathbb{R}^{n_{\mathfrak{p}} \times m_{\mathfrak{p}}}$ and $C \in \mathbb{R}^{p_{\mathfrak{p}} \times n_{\mathfrak{p}}}$.

NCS architecture with linear plants and dynamic devices

Sufficient conditions for controllability of a NCS with a linear plant (6.24), a processing unit $\mu_{\mathfrak{p}}^{d}$ and a network which imposes *periodic* scheduling are documented in [52]. We use this result in the sequel.

For the sake of clarity and self-containedness, let us begin by providing some additional notation from [52]. A periodic transmission sequences is denoted by $\sigma_\tau = \{\sigma(0), \sigma(1), \ldots, \sigma(\tau-1)\}$ where $\tau \in \mathbb{N}$ denotes the period. Further, for each $i \in \{0, \ldots, \tau-1\}$ we introduce a vector $\sigma(i) \in \{0,1\}^{m_{\mathfrak{p}}}$ with 0 denoting corresponding node not to be updated and 1 denoting corresponding node to be updated. For simplicity we refer to periodic transmission sequences as communication sequences.

Next, we define admissible communication sequences, which means that during period τ every node will be updated at least once.

Definition 6.6 ([52]). *Let the maximum number of nodes which can be addressed be a $\tilde{m} < m_{\mathfrak{p}}$. If for any period $\tau \in \mathbb{N}$ the following is satisfied:*

- *for each $i \in \{0, \ldots, \tau-1\}$, $\left|\sigma_\tau(i)\right| \leq \tilde{m}$;*
- *$span\big(\sigma(0), \ldots, \sigma(\tau-1)\big) = \mathbb{R}^{m_{\mathfrak{p}}}$,*

then the communication sequence σ_τ is admissible. □

Notice that an admissible sequence remains admissible if an element that already exists in the sequence or the element consisting only of zero values is added to it. We introduce the communication sequence matrix

$$\mathcal{E}(k,i) = \prod_{j=i}^{k} \big(I - \mathrm{diag}\big(\sigma(j)\big)\big),\ k \geq i. \tag{6.25}$$

The communication sequence matrix polynomial is defined as

$$\mathcal{G}(\mu) = \sum_{l=0}^{\tau-1} \big(\big(\mu^{\tau-1} I + \mu^{\tau-2}\mathcal{E}(l+1,l+1) + \cdots + \mathcal{E}(l+\tau-1,l+1)\big)\mathrm{diag}\big(\sigma(l)\big)\big) \tag{6.26}$$

with the indeterminate μ. The communication sequence characteristic polynomial is defined as

$$g(\mu) = \det\big(\mathcal{G}(\mu)\big). \tag{6.27}$$

Finally, the theorem that establishes controllability of a NCS with a linear plant (6.24), a processing unit $\mu_{\mathfrak{p}}^{d}$ and a network which imposes *periodic* scheduling is stated next.

Theorem 6.1 ([52]-Theorem 1)**.** *Consider a NCS with linear plant, see Eq. 6.24 and a processing unit $\mu_{\mathfrak{p}}^{d}$. Let the corresponding network impose periodic scheduling with period $\tau \in \mathbb{N}$. If*

- *A communication sequence σ_τ is admissible,*
- *The nonzero eigenvalues of matrix A do not coincide with the zeros of the communication sequence characteristic polynomial $g(\mu)$,*
- *The pair (A, B) is controllable,*

then the NCS is controllable. □

Notice once again that Theorem 6.1 establishes that if we have a controllable plant, a processing unit $\mu_{\mathfrak{p}}^{d}$ and a suitable admissible *periodic* communication sequence, then the resulting NCS will be controllable as well. However, the theorem does not address the existence of such a sequence. Answering this question is one of the results of this chapter.

We start by noticing that if an admissible sequence is extended so that it remains admissible, then the order of the resulting polynomials in (6.27) increases accordingly. Next,

we concentrate on the effects to the roots of the corresponding polynomials. However, before we provide some insight, notice that simple calculations yield

$$\mathcal{G}(\mu) = \mathtt{diag}\big(\mathcal{P}_1(\tau,\mu),\ldots,\mathcal{P}_{m_{\mathfrak{p}}}(\tau,\mu)\big)$$

where

$$\mathcal{P}_i(\tau,\mu) = \mu^{\tau-1}\sum_{l=0}^{\tau-1}\sigma_i(l) + \mu^{\tau-2}\sum_{l=0}^{\tau-1}\sigma_i(l)\prod_{j=l+1}^{l+1}(1-\sigma_i(j)) + \cdots$$
$$\cdots + \mu\sum_{l=0}^{\tau-1}\sigma_i(l)\prod_{j=l+1}^{l+\tau-2}(1-\sigma_i(j)) + \sum_{l=0}^{\tau-1}\sigma_i(l)\prod_{j=l+1}^{l+\tau-1}(1-\sigma_i(j))$$

for each $i \in \{1,\ldots,m_{\mathfrak{p}}\}$. Furthermore, note that the "minimum-length" communication sequence is the standard basis for $\mathbb{R}^{m_{\mathfrak{p}}}$.

Before we proceed, let us we define the nth root of unity, where $n \in \mathbb{N}$, as a $z \in \mathbb{C}$ such that $z^n = 1$; which is *primitive* if it is not kth root of unity for any $k \in \{1,\ldots,n-1\}$. Now, to gain some insight into the effects of enlarging the length of an admissible communication sequence in the way described above, we provide Table 6.1. One should notice that the "minimum-length" communication sequence (first row) and communication sequences formed by adding elements consisting only of zero values to the "minimum-length" communication sequence (second, fourth and seventh row) have roots on a unit circle. In fact, by adding $(0,\ldots,0)$'s to the "minimum-length" communication sequence, for each $i \in \{1,\ldots,m_{\mathfrak{p}}\}$, we generate polynomials $\mathcal{P}_i(\tau,\mu) = \sum_{j=0}^{\tau-1}\mu^j$. Such polynomials have *all* roots on a unit circle. However, if n is an odd number, then the corresponding polynomial will always have one root at -1. On the other hand, in the theory on cyclotomic polynomials, it is a well known fact that if $n = p-1$ where p is an *odd* prime number (any prime number other than 2 which is the unique *even* prime), then the roots of the corresponding polynomial will correspond to the pth *primitive* roots of the *unity* (see [175], page 306). More precisely $\sum_{j=0}^{p-1}\mu^j = \frac{\mu^p-1}{\mu-1}$ with all roots being distinct for each such p.

Now, before stating our first extension of Theorem 6.1 we modify Definition 6.3 so

that it periodically provides indices provided in admissible communication sequence.

Definition 6.7 (APC_{σ_τ})**.** *Consider* $\mathrm{R} \in \mathbb{Z}_{\geq 2}$ *nodes competing for access to the network and the network state vector* $x_{\mathfrak{n}} = (x_{\mathfrak{n}_1}, \ldots, x_{\mathfrak{n}_\mathrm{R}})$*. Given an admissible communication sequence* σ_τ*, then:*

- *There exist* $\delta > 0$*,*
- $\bar{r} \in \{1, \ldots, \mathrm{R}\}$ *is an index of a nonzero element from* σ_τ *at time instant* $l \cdot k$ *where* $l \in \{0, \ldots, \tau\}$ *and* $k \in \mathbb{N}_0$*,*
- $x_{\mathfrak{n}_r}(k) := y_{\mathfrak{c}_r}(k) + \delta, \forall r \in \{1, \ldots, \mathrm{R}\},\ r \neq \bar{r}$*.*

□

Theorem 6.2. *Consider a NCS with a linear plant* $\Sigma_{\mathfrak{p}}$*, see (6.24), and a processing unit* $\mu^d_{\mathfrak{p}}$*. Let the network be governed by the TOD protocol and let it have* APC_{σ_τ}*. If the pair* (A, B) *is controllable, then the NCS is controllable.* □

Remark: Results in [52] implicitly require that condition from Lemma 6.3 is satisfied (see (4) in [52]). This appears very restrictive unless special control sequences are considered for which appropriate devices exist; see Subsection 6.2.1. □

NCS architecture with linear plants and static devices

We recall that, at each time instant, processing unit $\mu^s_{\mathfrak{p}}$ applies the received value to the addressed node and zero values to the remaining nodes. Thus, we are considering control sequences from the set $\mathcal{S}^{\mathbb{R}^m}_{S_0}(x_{\mathfrak{p}}), x_{\mathfrak{p}} \in \mathbb{R}^{n_{\mathfrak{p}}}$; see Subsection 6.2.1. This means that the requirement as in Lemma 6.3 is not needed, making the forthcoming results less restrictive than Theorems 6.1 and 6.2.

Corollary 6.1. *Consider a NCS with a linear plant* $\Sigma_{\mathfrak{p}}$*, see (6.24), and a processing unit* $\mu^s_{\mathfrak{p}}$*. Let the corresponding network impose periodic scheduling with a given period* τ*. If*

- *A communication sequence* σ_τ *is admissible,*
- *The nonzero eigenvalues of matrix* A *do not coincide with the zeros of the communication sequence characteristic polynomial* $g(\mu)$*,*

- *The pair* (A, B) *is controllable,*

then the NCS remains controllable. □

Using the same ideas as in Theorem 6.2 we state

Corollary 6.2. *Consider a NCS with a linear plant* $\Sigma_{\mathfrak{p}}$*, see (6.24), and a processing unit* $\mu^{\mathrm{s}}_{\mathfrak{p}}$*. Let the corresponding network be governed by the TOD protocol and let it have* $\mathrm{APC}_{\sigma_\tau}$ *If the pair* (A, B) *is controllable, then the NCS is controllable.* □

6.3 Proofs

Proof of Lemma 6.1. Let $\bar{r} \in \{1, \ldots, \mathrm{R}\}$ be given. Then, according to APC, for all $r \in \{1, \ldots, \mathrm{R}\}, r \neq \bar{r}$, $x_{\mathfrak{n}_r}(k) := u_{\mathfrak{p}_r}(k) + \delta$. It follows that the corresponding error vector (see (6.5)) $e(k) = (\delta, \ldots, \delta, e_{\bar{r}}(k), \delta, \ldots, \delta)$ with $e_{\bar{r}}(k) = 0$. Hence, according to TOD protocol $u_{\mathfrak{p}_r}(k)$ is chosen for transmission. □

Proof of Lemma 6.2. Consider any sequence from $\mathcal{S}^{\mathbb{R}^m}_{\mathrm{S}_0}(x_{\mathfrak{p}})$ and an element of the corresponding sequence at time instant $k \in \mathbb{N}_0$. According to APC_0, an index $\bar{r} \in \{1, \ldots, \mathrm{R}\}$ of the nonzero element from the corresponding control vector $u^0(k)$ or any index otherwise is picked and $x_{\mathfrak{n}_r}(k) := u^0_r(k) + \delta, \delta > 0, \forall r \in \{1, \ldots, \mathrm{R}\}$, $r \neq \bar{r}$. Correspondingly, the error vector $e(k) = (\delta, \ldots, \delta, e_{\bar{r}}(k), \delta, \ldots, \delta)$ with $e_{\bar{r}}(k) = 0$; for simplicity let $\bar{r}$ be also the index if all elements of the $u^0(k)$ are equal to zero. Hence, according to TOD $u^0_{\bar{r}}(k)$ is chosen. Effectively, processing unit $\mu^{\mathrm{s}}_{\mathfrak{p}}$ receives $u^0(k)$ and according to Fig. 6.2 and (6.9) $u_{\mathfrak{p}}(k) = y_{\mu^{\mathrm{s}}_{\mathfrak{p}}}(k) = u^0(k)$, as desired. □

Proof of Lemma 6.3. Consider $\{u^\delta\}^\infty_0 \in \mathcal{S}^{\mathbb{R}^m}_{\mathrm{S}_\delta}(x_{\mathfrak{p}})$ with $x_{\mu^d_{\mathfrak{p}}}(0) = u^\delta(0)$. Next, consider $u^\delta(k), k \in \mathbb{N}$. According to APC_δ an index $\bar{r} \in \{1, \ldots, \mathrm{R}\}$ of differing element between $u^\delta(k)$ and $u^\delta(k-1)$ or any index otherwise is picked and $x_{\mathfrak{n}_r}(k) := u^\delta_r(k) + \delta, \delta > 0, \forall r \in \{1, \ldots, \mathrm{R}\}$, $r \neq \bar{r}$. Correspondingly, the error vector $e(k) = (\delta, \ldots, \delta, e_{\bar{r}}(k), \delta, \ldots, \delta)$ with $e_{\bar{r}}(k) = 0$; for simplicity let $\bar{r}$ be also the index if $u^\delta(k) = u^\delta(k-1)$. Hence, according to TOD protocol $u^\delta_{\bar{r}}(k)$ is chosen for transmission. Finally, according to Fig. 6.2 and (6.16) $u_{\mathfrak{p}}(k) = y_{\mu^d_{\mathfrak{p}}}(k) = u^\delta(k)$ as desired. □

Proof of Theorem 6.2. Consider $A \in \mathbb{R}^{n_{\mathfrak{p}} \times n_{\mathfrak{p}}}$ and recall that for a given $\tau \in \mathbb{N}$, the polynomial $\sum_{j=0}^{\tau-1} \mu^j$, indeterminate μ, has *all* roots on the unit circle. The corresponding matrix either has no eigenvalues on the unit circle or finitely many. If A has no eigenvalues on the unit circle, then $\tau \geq n_{\mathfrak{p}}$. Otherwise, there exists a finite number of *odd* prime numbers for which roots of the corresponding polynomial coincide with the eigenvalues of A. However, since there are infinitely many *odd* prime numbers, there exists an *odd* prime number τ for which roots of the corresponding polynomial do not coincide with the eigenvalues of A. We proceed with adding $(0, \ldots, 0)$ elements to the "minimum-length" communication sequence so that the resulting length of the new admissible communication sequence equals to τ. We denote this new sequence with σ_τ. It follows that all conditions of Theorem 6.1 are satisfied. Hence, the NCS is controllable. Now, according to APC_{σ_τ}, at time instant $l \cdot k$ where $l \in \{0, \ldots, \tau\}$ and $k \in \mathbb{N}_0$, the index $\bar{r} \in \{1, \ldots, \mathrm{R}\}$ is an index of a nonzero element from σ_τ, and $x_{\mathfrak{n}_r}(l \cdot k) := y_{\mathfrak{c}_r}(lk) + \delta, \delta > 0, \forall r \in \{1, \ldots, \mathrm{R}\}$, $r \neq \bar{r}$. Correspondingly, the error vector $e(l \cdot k) = (\delta, \ldots, \delta, e_{\bar{r}}(lk), \delta, \ldots, \delta)$ with $e_{\bar{r}}(l \cdot k) = 0$. Hence, according to TOD, $y_{\mathfrak{c}_{\bar{r}}}(l \cdot k)$ is chosen for transmission. Finally, according to Fig. 6.2 and (6.16), $u_{\mathfrak{p}}(l \cdot k) = y_{\mu^d_{\mathfrak{p}}}(l \cdot k) = y_{\mathfrak{c}}(l \cdot k)$, as desired. □

Proof of Corollary 6.1. The proof follows the same lines of the proof of Theorem 6.1 in [52]. Note that $\mathbf{u}(t) = \mathtt{diag}(\sigma_c(t))\mathbf{u}_\ell(t)$ which impacts equations from (9) to (40) in the following way $D_c(\cdot, \star) := \mathrm{diag}\big(\sigma_c(\cdot)\big)$. □

Proof of Corollary 6.2. The proof differs from the proof of Theorem 6.2 only in the fact that instead of Theorem 6.1, Corollary 6.1 is used and in the last line instead of $y_{\mu^d_{\mathfrak{p}}}$, $y_{\mu^s_{\mathfrak{p}}}$ is used. □

τ	σ_w	$\mathcal{G}(\mu)$	$\{\mu : g(\mu) = 0\}$
2	$\{(0,1),(1,0)\}$	$\begin{bmatrix} \mu+1 & 0 \\ 0 & \mu+1 \end{bmatrix}$	$\{-1\}$
3	$\{(0,1),(0,0),(1,0)\}$	$\begin{bmatrix} \mu^2+\mu+1 & 0 \\ 0 & \mu^2+\mu+1 \end{bmatrix}$	$\{-0.5 \pm i0.86\}$
3	$\{(0,1),(0,1),(1,0)\}$	$\begin{bmatrix} \mu^2+\mu+1 & 0 \\ 0 & 2\mu^2+\mu \end{bmatrix}$	$\{-0.5, 0, -0.5 \pm i0.86\}$
4	$\{(0,1),(0,0),(0,0),(1,0)\}$	$\begin{bmatrix} \mu^3+\mu^2+\mu+1 & 0 \\ 0 & \mu^3+\mu^2+\mu+1 \end{bmatrix}$	$\{-1, \pm i\}$
4	$\{(0,1),(0,1),(0,0),(1,0)\}$	$\begin{bmatrix} \mu^3+\mu^2+\mu+1 & 0 \\ 0 & 2\mu^3+\mu^2+\mu \end{bmatrix}$	$\{-1, 0, \pm i, -0.25 \pm i0.66\}$
4	$\{(0,1),(0,1),(0,1),(1,0)\}$	$\begin{bmatrix} \mu^3+\mu^2+\mu+1 & 0 \\ 0 & 3\mu^3+\mu^2 \end{bmatrix}$	$\{-1, -0.33, 0, \pm i\}$
5	$\{(0,1),(0,0),(0,0),(0,0),(1,0)\}$	$\begin{bmatrix} \mu^4+\mu^3+\mu^2+\mu+1 & 0 \\ 0 & \mu^4+\mu^3+\mu^2+\mu+1 \end{bmatrix}$	$\{-0.81 \pm i0.56, 0.31 \pm i0.95, -0.62 \pm i0.51, 0.37 \pm i0.8\}$
5	$\{(0,1),(0,1),(0,0),(0,0),(1,0)\}$	$\begin{bmatrix} \mu^4+\mu^3+\mu^2+\mu+1 & 0 \\ 0 & 2\mu^4+\mu^3+\mu^2+\mu \end{bmatrix}$	$\{0, -0.74, 0.12 \pm i0.81, -0.62 \pm i0.5, 0.37 \pm i0.8\}$
5	$\{(0,1),(0,1),(0,1),(0,0),(1,0)\}$	$\begin{bmatrix} \mu^4+\mu^3+\mu^2+\mu+1 & 0 \\ 0 & 3\mu^4+\mu^3+\mu^2 \end{bmatrix}$	$\{0, -0.81 \pm i0.59, 0.31 \pm i0.95, -0.16 \pm i0.55\}$
5	$\{(0,1),(0,1),(0,1),(0,1),(1,0)\}$	$\begin{bmatrix} \mu^4+\mu^3+\mu^2+\mu+1 & 0 \\ 0 & 4\mu^4+\mu^3 \end{bmatrix}$	$\{0, -0.25, -0.81 \pm i0.59, 0.31 \pm i0.96\}$

Table 6.1: Communication sequence matrix polynomials for different periodic sequences and roots of the corresponding polynomials for a second ordered system; permutation of added elements, adding $(1,0)$ instead of $(0,1)$, or adding both where possible, does not generate new polynomials.

This page intentionally left blank.

Chapter 7
Implementation

Results we establish rely on the corresponding theoretical models and assumptions. In order to demonstrate that our theoretical results hold in applications which involve real hardware and also to provide a better insight into the theory through linking models with the real NCS we implement the considered framework.

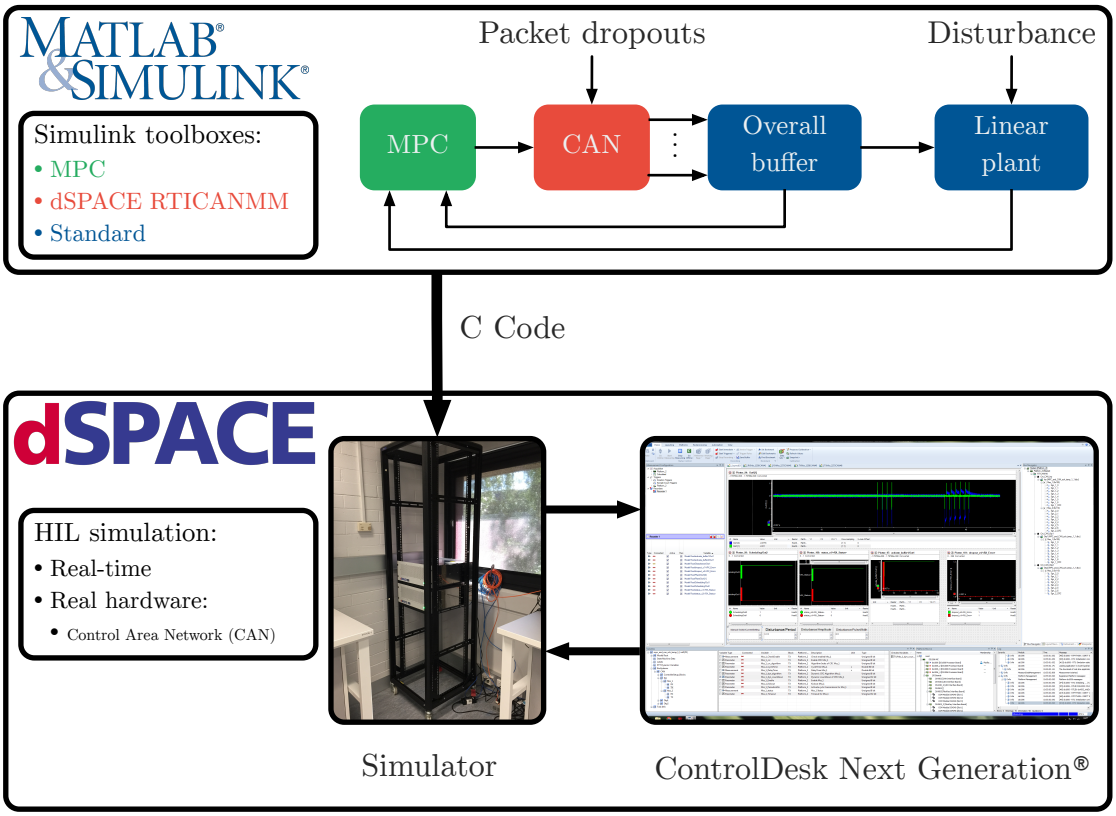


Figure 7.1: Implementation as a Hardware-In-the-Loop (HIL) simulation.

More precisely, we implement the NCS architecture considered in Chapter 3 – Chap-

ter 5 as the so-called Hardware-In-the-Loop (HIL) simulation using dSPACE® Simulator, see Fig. 7.1. HIL simulation, as the name suggests, is a (real-time) simulation which involves real hardware. In our case, the real hardware is a Control Area Network (CAN). Furthermore, for all simulated parts, e.g., controller, buffer(s) and plant, the corresponding C code is generated and downloaded to dSPACE® Simulator processor which communicates with CAN bus directly.

Besides validating theoretical results, the corresponding implementation has many other benefits. For instance, many interesting problems can arise during this process which can lead to interesting and important extensions of the theoretical questions. Furthermore, this helps to deeper understand the corresponding intrinsic processes which in turn can correct and/or change the initial theoretical approach or even lead to new questions.

A widespread approach in the development of complex system, especially control systems, such as an airplane, is to implement the corresponding system (or parts of it) as a HIL simulation before the final production. Roughly speaking, HIL simulation is a class of real-time simulations[1] which involves real hardware, see [176]. There are many reasons why HIL simulations are so omnipresent, spanning many fields such as the development time, cost and safety. These simulations are the standard in industry and many manufacturers are using them when developing their products, e.g., see [176] and references therein.

In particular, our HIL simulation is implemented by using MATLAB® and Simulink® software together with dSPACE® hardware - simulator (real CAN bus) and software - ControlDesk Next Generation®, see Fig. 7.1. This implementation alone is one contribution of this chapter. Additionally, some interesting extensions and/or changes, such as closing the whole loop via network and/or changing a network (to, say, FlexRay), are relatively easy to do and the corresponding development time is relatively short. Another contribution of this chapter is the verification of the considered protocol and controller co-design and the concept of using predictions within MPC framework to account for packet dropouts and scheduling. More precisely, we demonstrate that our stability

[1]Notice that real-time simulations are a class of simulations in which the *simulated* system generates the *same*, time-dependent, input and output signal values as the real system.

result from Chapter 3 and our robustness result from Chapter 5 hold, albeit with deteriorated performance. Namely, during the process of implementation we encountered other communication issues such as quantization and delay which revealed that some of our assumptions might be unrealistic or strong for certain types of networks such as CAN bus.

The rest of the chapter is organized as follows. First, in Section 7.1, we introduce the considered NCS architecture. Then, in Section 7.2, we check if our stability and robustness assumptions are satisfied. Due to the amount of the corresponding information in Section 7.3 we present a conceptual outline of the implementation; the missing details can be found in Appendix A. Finally, in Section 7.4 we present implementation results.

7.1 NCS architecture

The considered NCS architecture is introduced in Chapter 3, see Fig. 3.1 and in Chapter 5, see Fig. 5.1. For the sake of a more fluid presentation, we depict the corresponding NCS architecture (with slight modifications) once again, namely, see Fig. 7.2.

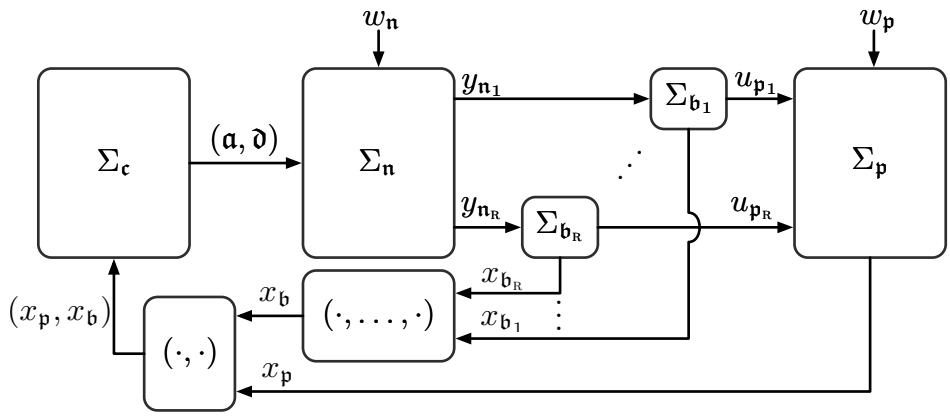


Figure 7.2: Implemented NCS architecture.

Notice, that unlike in Fig. 3.1 and Fig. 5.1, where we were general when it comes to how the controller knows the contents of the buffers[2], in Fig. 7.2 we explicitly illustrate

[2]Namely, we used ACK in the feedback to denote several possibilities, one being the usage of acknowledgment of receipt.

that we send buffer measurements together with plant measurements; notice that since there is no network between plant/buffer(s) output and controller input, for implementation purposes, it was much simpler to send buffer(s) and plant measurements together.

In a similar fashion as in previous chapters, we proceed with the presentation of the parts of the considered NCS architecture.

7.1.1 Plant

We restrict our attention to linear plants since MATLAB® offers an MPC toolbox which we use to obtain an optimal node and its sequence of optimally predicted control values. Moreover, we restrict our attention to second-order linear plants with two inputs since two inputs are sufficient to show scheduling. Thus, the considered discrete-time plant is given as

$$\Sigma_{\mathfrak{p}} : x_{\mathfrak{p}}^{+} = f_{\mathfrak{p}}(x_{\mathfrak{p}}, u_{\mathfrak{p}}, w_{\mathfrak{p}}) := Ax_{\mathfrak{p}} + Bu_{\mathfrak{p}} + Ww_{\mathfrak{p}}. \tag{7.1}$$

where $x_{\mathfrak{p}} \in \mathbb{R}^2$ is the state, $u_{\mathfrak{p}} \in \mathbb{R}^2$ is the input and $w_{\mathfrak{p}} \in \mathbb{R}^2$ is the exogenous disturbance of the plant while $A \in \mathbb{R}^{2\times 2}$, $B \in \mathbb{R}^{2\times 2}$ and $W \in \mathbb{R}^{2\times 2}$. In particular we consider

$$\left.\begin{aligned} A &= \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}, \\ B &= \begin{bmatrix} 0 & \frac{1}{2} \\ 1 & \frac{1}{2} \end{bmatrix}, \\ W &= \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}. \end{aligned}\right\} \tag{7.2}$$

7.1.2 Network

The model of the network we use in Chapter 3 – Chapter 5 focuses only on two communication issues, namely, packet dropouts and scheduling. It is simple, general and it can correspond to many types of networks (where other communication issues can be ignored);

for instance, it can correspond to a network which uses all seven layers of OSI/ISO communication model (see Fig. A.2 in Appendix A) and it can also correspond to a network which uses only three layers such as a CAN bus (see Fig. A.4 in Appendix A). However, in practical applications, usually other communication issues cannot be ignored. In our case, CAN bus imposed delay and quantization which we did not addressed in our analysis and design. Thus, one of the reasons for the implementation was to test our analysis and design (based on a general and yet simple network mode) an a real network.

As mentioned above we use CAN bus as a network in our NCS; please see Section A.2 in Appendix A for basic information about CAN that is relevant to the implementation. As indicated in the previous paragraph, our network model is very simple and general but it ignores some intrinsic network phenomena such as quantization, see Fig. 7.3, and delay, see Fig. 7.4, which are present on CAN bus.

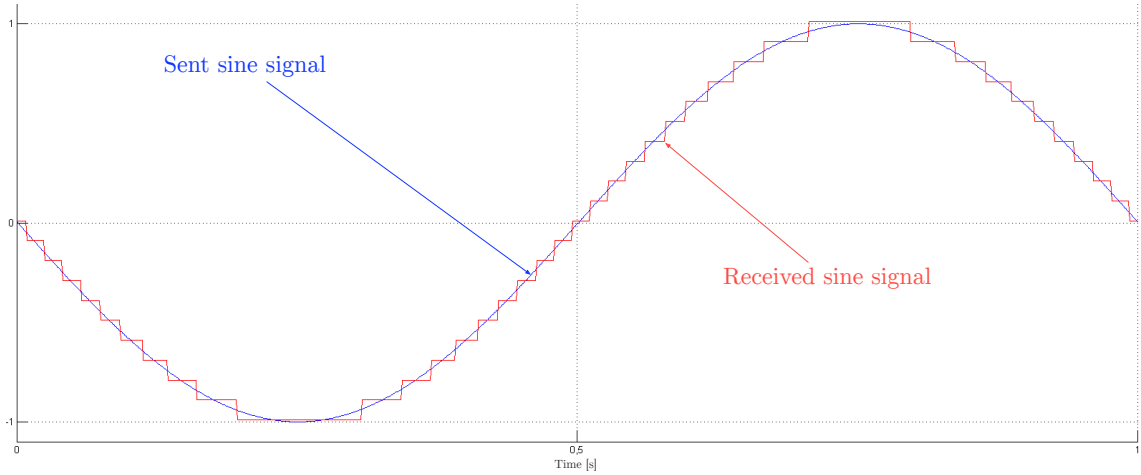


Figure 7.3: Demonstrating quantization issues on CAN bus; here we use only one byte to transmit the values of sine signal since we have to use one byte to transmit the values of each control element from the sequence of the optimally predicted controls.

Due to the fact that our network model does not account for these communication issues there are some consequences. Namely, the issue of quantization (see Fig. 7.3) results in deteriorated performance (e.g., practical stability) as documented in Section 7.4 or it can lead to instability due to control saturation, see Section A.3.2 in Appendix A. On the other hand, the issue of delay (Fig. 7.4) was easily and successfully addressed within MPC toolbox; namely input delays (recall that we only have network between

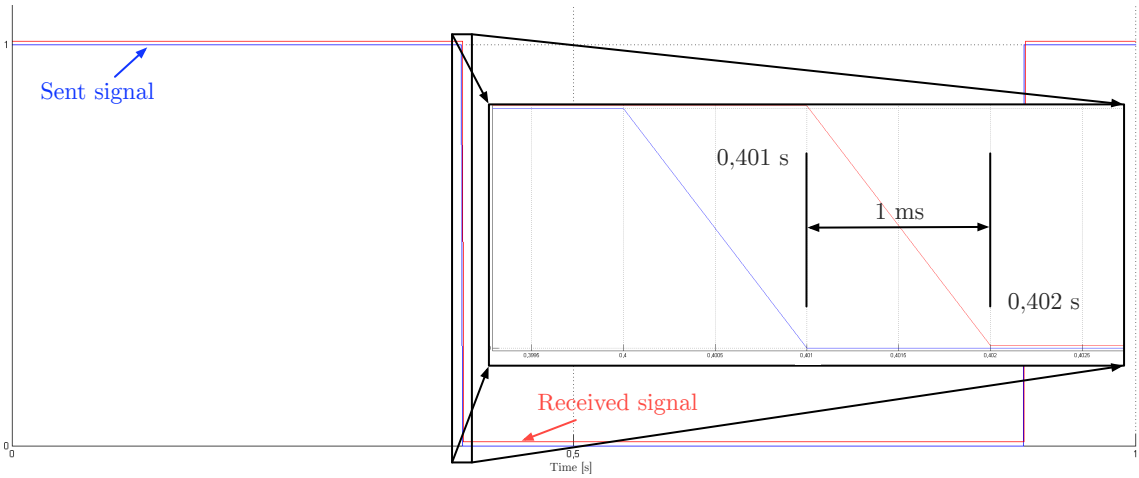


Figure 7.4: Demonstrating delay on CAN bus.

controller's output and plant's inputs) for linear systems can be addressed by enlarging the dimension of plant dynamics.

On the other hand, communication on CAN bus is a communication on demand, which is based on event-driven bus access methods such as the CSMA/CA method (Carrier Sense Multiple Access with Collision Avoidance). Thus, as our network model ensures, we can manipulate which node gets the access which is needed for the used protocol and controller co-design to work. Further, data communication on CAN bus is performed with the so-called frames, see Fig. A.3, and only eight bytes are allocated for useful data. Notice that in our case this will put a restriction on the length of the prediction horizon, see Section A.3.2, and result in quantization issue, see Fig. 7.3.

We note that packet dropouts are not common on CAN bus due to its design. Namely, the risk of message collisions (e.g., packet dropouts) is countered with the priority-driven CSMA/CA method, see Section A.2. Thus, we need to generate errors in communication which can result in packet dropouts. We do this in two ways. Both ways are used in the automotive industry. One approach is to write a customer-specific "faulty" CRC (Cyclic Redundancy Check), see Listing A.1 and Fig. A.14. This sequence, when received will cause an error which will stop an ongoing communication. It should be noted that even though this error is induced "artificially", everything is still occurring on the real hardware (i.e., on the real CAN bus) and all corresponding responses are real, i.e., the abortion

of an ongoing communication. Another, more straightforward approach is that during experiment run-time we block a sender node for a certain amount of time (see Fig. A.14) which results in packets not being received, i.e., effectively a packet dropout(s). For the corresponding respond in the case of "artificially" induced packet dropout (as described above) see Fig 7.5

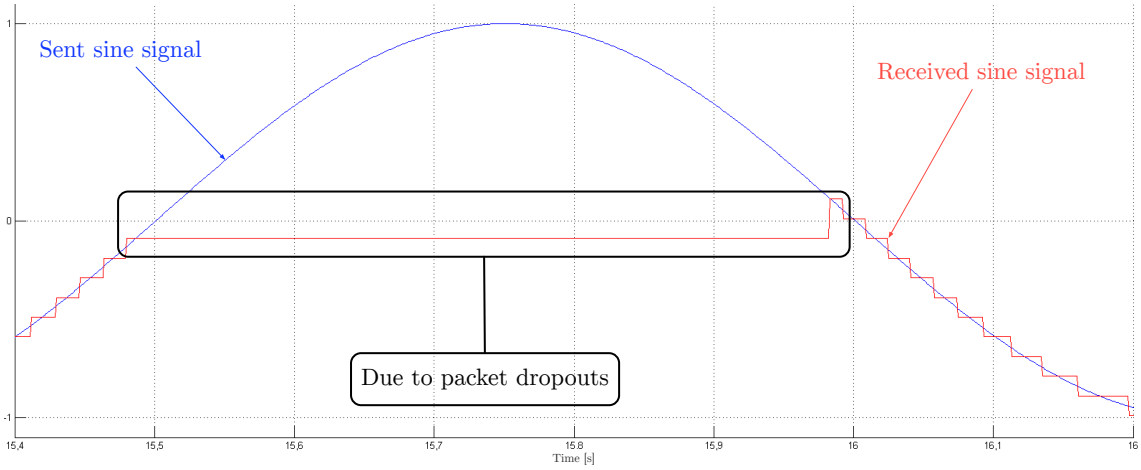


Figure 7.5: The effects of generated packet dropouts.

Indeed, our results are more applicable to the cases where networks are more prone to packet dropouts such as Ethernet or a wireless network and in future we will perform experiments on these networks as well. However, as it will be shown in the sequel, the results we already have are a good indication of what to expect and moreover we additionally have computer simulations which make the same predictions, see Section 5.2.3.

7.1.3 Buffer

The dynamics of the buffer remains the same, see Section 3.1.3 and/or Section 5.1.3. However, due to the fact that CAN frame has only eight bytes for useful data we set the length of each individual buffer (there are two) to seven, i.e., $L = 7$; see a paragraph after equation (3.12) and also see Section A.3.2. Also, notice that we leave one byte for CRC check-sum computed in Listing A.1, which will be used to generate packet dropouts.

7.1.4 Controller

As mentioned above, we chose linear plant (7.1) for implementation since MATLAB® provides an MPC toolbox for linear plants. Recall that due to scheduling, effectively, we need two MPC controllers, one for each plant input, see Section 3.1.4. Also, recall that the length of the buffers corresponds to the length of horizon, hence, $\mathfrak{h} = \mathrm{L} = 7$, see Section A.3.2. Now, following the contents from Section 3.1, the corresponding derivation for linear plant (7.1) is as follows.

From (3.25) it follows

$$\Sigma_{\mathfrak{p}}^{\mathfrak{m}} : \begin{cases} \tilde{x}_{\mathfrak{p}}(k+i+1) := A\tilde{x}_{\mathfrak{p}}(k+i) + B\tilde{u}_{\mathfrak{p}}^{r}(k+i), \\ \tilde{x}_{\mathfrak{p}}(k) = x_{\mathfrak{p}}(k),\ k \in \mathbb{N}_0,\ i \in \{0, \ldots, \mathfrak{h}-1\}. \end{cases} \tag{7.3}$$

The overall shift matrix (see (3.21)) remains the same, namely

$$S := \operatorname{diag}(S_{m_{\mathfrak{p}_1}}, \ldots, S_{m_{\mathfrak{p}_{\mathrm{R}}}}), \tag{7.4}$$

Furthermore, from (3.26) it follows

$$\tilde{u}_{\mathfrak{p}}^{r} := \begin{bmatrix} \Gamma_{m_1}\tilde{x}_{\mathfrak{b}} \\ \vdots \\ \Gamma_{m_{r-1}}\tilde{x}_{\mathfrak{b}} \\ \tilde{u}_{\mathfrak{p}_r} \\ \Gamma_{m_{r+1}}\tilde{x}_{\mathfrak{b}} \\ \vdots \\ \Gamma_{m_{\mathrm{R}}}\tilde{x}_{\mathfrak{b}} \end{bmatrix} = Z_r\tilde{x}_{\mathfrak{b}} + R_r\tilde{u}_{\mathfrak{p}}, \tag{7.5}$$

where

$$Z_r = \begin{bmatrix} \Gamma_{m_1} \\ 0_{m_2} \\ \vdots \\ 0_{m_{\mathrm{R}}} \end{bmatrix} + \cdots + \begin{bmatrix} 0_{m_1} \\ \vdots \\ 0_{m_{r-2}} \\ \Gamma_{m_{r-1}} \\ 0_{m_r} \\ \vdots \\ 0_{m_{\mathrm{R}}} \end{bmatrix} + \begin{bmatrix} 0_{m_1} \\ \vdots \\ 0_{m_{\mathrm{R}}} \end{bmatrix} + \begin{bmatrix} 0_{m_1} \\ \vdots \\ 0_{m_r} \\ \Gamma_{m_{r+1}} \\ 0_{m_{r+2}} \\ \vdots \\ 0_{m_{\mathrm{R}}} \end{bmatrix} + \cdots + \begin{bmatrix} 0_{m_1} \\ \vdots \\ 0_{m_{\mathrm{R}-1}} \\ \Gamma_{m_{\mathrm{R}}} \end{bmatrix}, \tag{7.6}$$

and

$$R_r = \begin{bmatrix} 0_{m_1} & \cdots & 0_{m_{r-1}} & 0_{m_r} & 0_{m_{r+1}} & \cdots & 0_{m_{\mathrm{R}}} \\ & \vdots & & & & \vdots & \\ 0_{m_1} & \cdots & 0_{m_{r-1}} & I_{m_r} & 0_{m_{r+1}} & \cdots & 0_{m_{\mathrm{R}}} \\ & \vdots & & & & \vdots & \\ 0_{m_1} & \cdots & 0_{m_{r-1}} & 0_{m_r} & 0_{m_{r+1}} & \cdots & 0_{m_{\mathrm{R}}} \end{bmatrix}. \tag{7.7}$$

Buffer model (see (3.27)) is given as

$$\Sigma_{\mathfrak{b}}^{\mathfrak{m}} : \begin{cases} \tilde{x}_{\mathfrak{b}}^{+} & := S\tilde{x}_{\mathfrak{b}}, \\ \tilde{x}_{\mathfrak{b}} & = x_{\mathfrak{b}}. \end{cases} \tag{7.8}$$

Finally, the overall NCS model used in optimization (see (3.31)) becomes

$$\begin{aligned} \tilde{x}^{+} &= \begin{bmatrix} A & 0 \\ 0 & S \end{bmatrix} \begin{bmatrix} \tilde{x}_{\mathfrak{p}} \\ \tilde{x}_{\mathfrak{b}} \end{bmatrix} + \begin{bmatrix} B \\ 0 \end{bmatrix} \tilde{u}_{\mathfrak{p}} \\ &= \begin{bmatrix} A & 0 \\ 0 & S \end{bmatrix} \begin{bmatrix} \tilde{x}_{\mathfrak{p}} \\ \tilde{x}_{\mathfrak{b}} \end{bmatrix} + \begin{bmatrix} BZ_r \\ 0 \end{bmatrix} \tilde{x}_{\mathfrak{b}} + \begin{bmatrix} BR_r \\ 0 \end{bmatrix} \tilde{u}_{\mathfrak{p}} \\ &= \begin{bmatrix} A & BZ_r \\ 0 & S \end{bmatrix} \begin{bmatrix} \tilde{x}_{\mathfrak{p}} \\ \tilde{x}_{\mathfrak{b}} \end{bmatrix} + \begin{bmatrix} BR_r \\ 0 \end{bmatrix} \tilde{u}_{\mathfrak{p}}. \end{aligned} \tag{7.9}$$

(Note that in our experiment $\mathrm{R} = 2$.) The corresponding MATLAB® script is given in

Listing A.2.

7.2 Assumptions

In what follows we provide some comments on the assumptions we made to establish our stability and robustness results. Recall that our stability (UGAS) result, i.e., Lemma 3.1, relies on Assumption 3.1 – Assumption 3.4, while our robustness (partial linear gain ℓ_2 stability) result, i.e., Corollary 5.1, relies on Assumption 5.1 – Assumption 5.8.

7.2.1 Stability – UGAS

Assumption 3.1 – Bound on the number of consecutive packet dropouts

As discussed in Section A.2, CAN bus is very reliable when it comes to packet dropouts. This is due to the fact that it uses the priority-driven CSMA/CA method to address the risk of message collisions (e.g., packet dropouts). In fact, packet dropout usually occurs to low-priority packet (messages) when CAN bus is very loaded. Thus, this assumption is satisfied. In order to test our results with respect to packet dropouts we create conditions, as described in Section 7.1.2, which induce packet dropout(s). In fact, in Section 7.4 we violate this assumption to demonstrate that our results hold even in this scenario.

Assumption 3.2 – Semi-positive definiteness and lower bounds on stage cost function

The cost function in MATLAB® MPC tool box is quadratic, thus, this assumption is satisfied.

Assumption 3.3 – Terminal control law

Due to linear plant dynamics of the corresponding MPC model, quadratic cost and unconstrained optimization problem this assumption is satisfied; e.g., see [107].

Assumption 3.4 – Class-$\mathcal{K}$ bound on the optimal value function

Similarly as for Assumption 3.2, since the cost function is quadratic, Assumption 3.4 is satisfied.

7.2.2 Robustness – Partial linear gain ℓ_2 stability

Assumption 5.1 – Continuity

Due to linear dynamics this assumption is satisfied.

Assumption 5.2 – Bound on the number of consecutive packet dropouts

The same comments apply as for Assumption 3.1 made above.

Assumption 5.3 – Semi-positive definiteness and lower bound on stage cost function

The same comments apply as for Assumption 3.2 made above.

Assumption 5.4 – Terminal control law

The same comments apply as for Assumption 3.3 made above.

Assumption 5.5 – Continuity of the optimal value function

Since plant dynamics is linear, cost function is quadratic and sets where state, input and disturbance evolve are compact, this assumptions is satisfied.

Assumption 5.6 – Class-$\mathcal{K}$ bound on the optimal value function

The same comments apply as for Assumption 3.1 made above.

Assumption 5.7 – Bound on plant trajectories until fist successful transmission

Due to linear dynamics and Assumption 5.1, Assumption 5.7 is satisfied.

Assumption 5.8 – Positive invariance (recursive feasibility)

Recursive feasibility is a peculiar property and its explicit consideration is left for future work. However, for the implementation purposes, following the discussion given in [107] after Assumption 3.9, we kept the disturbance small enough so that our Assumption 5.8 is satisfied; notice that the dynamics of our plant is linear, the cost function in MATLAB® MPC tool box is quadratic and there are no constraints.

7.3 Implementation

As mentioned in the introduction above, due to the amount of the corresponding information, we only provide a conceptual outline of the implementation and move all the missing details to Appendix A. However, notice that even in Appendix A we do not include all the details since the complete amount of information is vast. Namely, in order to include all the relevant details we would have to include extensive "how-to" information from several manuals and tutorials. Nonetheless, we provide the corresponding references or, if possible, we filter the necessary amount of information. Succinctly, the setup procedure includes the following steps:

1. Setting up dSPACE® simulator and establish working connection with the corresponding workstation (e.g., a PC),
2. Choosing the plant you wish to control – see Section A.3.1,
3. Designing CAN structure using Vector Informatik® CANdb++ editor – see Section A.3.2,
4. Making the corresponding physical connections that resemble the designed CAN structure – see Section A.3.3,
5. Designing MPC controller using MATLAB® MPC toolbox – see Section A.3.4,
6. Designing the corresponding NCS model using Simulink®, generating the corresponding C-code and downloading it onto dSPACE® simulator hardware – see Section A.3.5,

7. Designing the layouts in dSPACE® Control Desk New Generation which will capture the corresponding test results – see Section A.3.6 and Section 7.4 for the corresponding results.

As mentioned in the introduction above, the implementation is done as a HIL simulation. In particular, we use MATLAB® and Simulink® software together with dSPACE® hardware - simulator (real CAN bus) and software - ControlDesk Next Generation® to achieve this, i.e., see Fig. 7.1. Theoretically, the considered NCS is depicted in Fig. 7.1 while a conceptual MATLAB®-like counterpart, which takes into account the fact that our plant has two inputs is depicted in Fig. 7.6.

We consider linear plant (7.1) which has two inputs. This means we will need two MPC controllers and two buffers, each for one plant input. Each MPC controller will compute a sequence of optimal control values for its plant input and provide this sequence to the CAN sender node along with the corresponding value of the optimal cost function. Then, the values of the two optimal value functions are compared to determine the minimal one and the corresponding signal (i.e., true or false) is provided to CAN sender node. CAN sender node will use this signal to decide which sequence of optimal control values to send; i.e., the one that corresponds to the optimal value function with the minimal value. If transmission was successful (no dropout), the corresponding CAN receiver node will provide this information along with the information that new data arrived and the new data (i.e., the corresponding sequence of optimal control values). These ingredients are further used to update the corresponding buffer with new data. On the other hand, the other CAN receiver node will provide information that no new data arrived which is further used to inform buffer to keep using its current content. Similarly, in case of a packet dropout, the corresponding CAN receiver node would provide this information which would inform buffer to keep using its current content. Finally, buffer values are applied to plant input and plant state along with buffer state is sent directly to the controller(s).

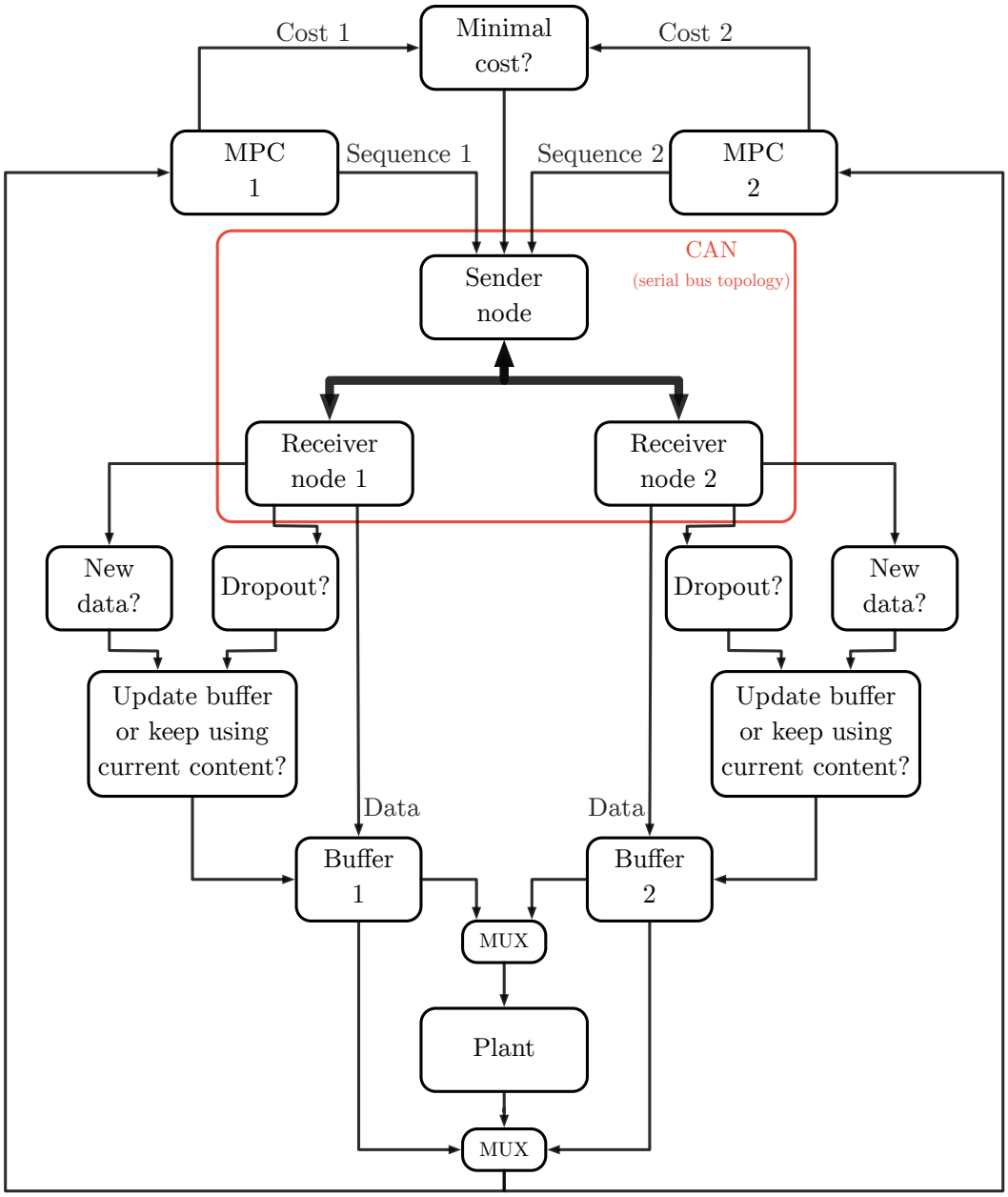


Figure 7.6: Conceptual illustration of the considered NCS in Simulink®.

7.4 Results

We use dSPACE® ControlDesk Next Generation® to run our HIL simulation and to capture the corresponding results, see Section A.3.6. First, we present the results when the

plant is not affected with disturbances, i.e., our UGAS result (see Lemma 3.1). Then, we present the results when the plant is affected with disturbances, i.e, our partial linear gain ℓ_2 stability (see Corollary 5.1).

Stability – UGAS

We first present the scheduling of inputs $u_{\mathfrak{p}_1}$ and $u_{\mathfrak{p}_2}$. As documented in Fig. 7.7 and Fig. 7.8, the dynamics is such that scheduling only occurs in, roughly, first two seconds and then input $u_{\mathfrak{p}_2}$ takes over. Next, in Fig. 7.9, we present Euclidean norm of the plant state, i.e.,

$$||x_{\mathfrak{p}}||^2_{\mathcal{L}_2} := |x_{\mathfrak{p}_1}|^2 + |x_{\mathfrak{p}_2}|^2.$$

As indicated with Fig. 7.7 and Fig. 7.8, input $u_{\mathfrak{p}_2}$ takes over. As mentioned earlier, we generate packet dropouts as described in Section A.3.6. Recall, that if a packet dropout occurs the corresponding buffer has to use the content from its memory. We capture this in Fig. 7.10 and we show the corresponding Euclidean norm of plant state in Fig. 7.11. As stated above, we are violating Assumption 3.1. This combined with the quantization issues can be a reason why there is a considerable increase in the state norm for the second set of packet dropouts in Fig. 7.11. Additionally, notice also the performance deterioration in Fig. 7.9 due to quantization issues; see Section A.3.2.

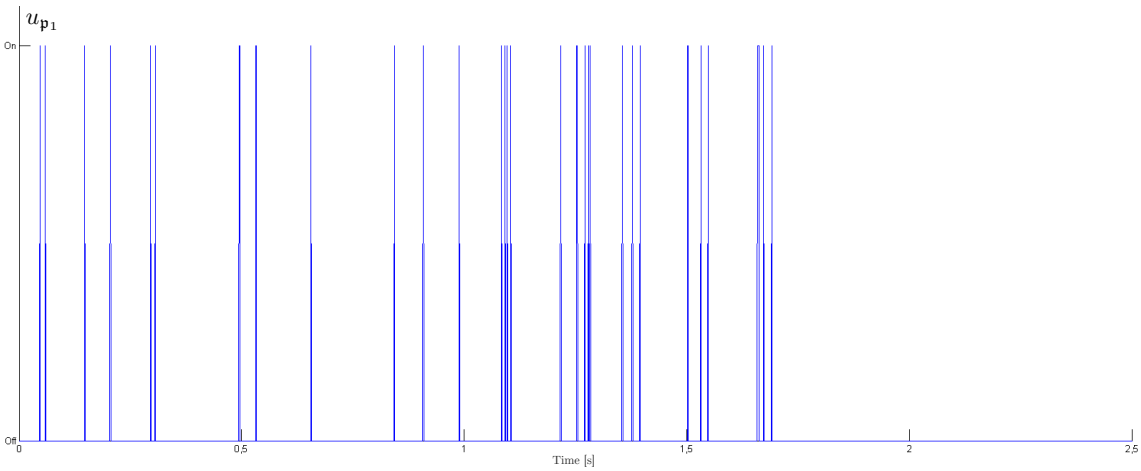


Figure 7.7: Stability: scheduling of input $u_{\mathfrak{p}_1}$

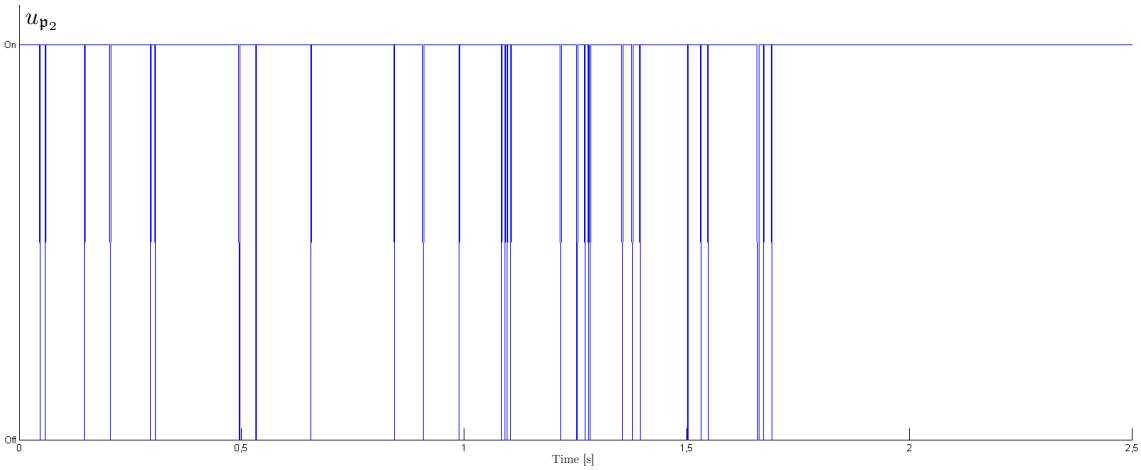


Figure 7.8: Stability: scheduling of input $u_{\mathfrak{p}_2}$

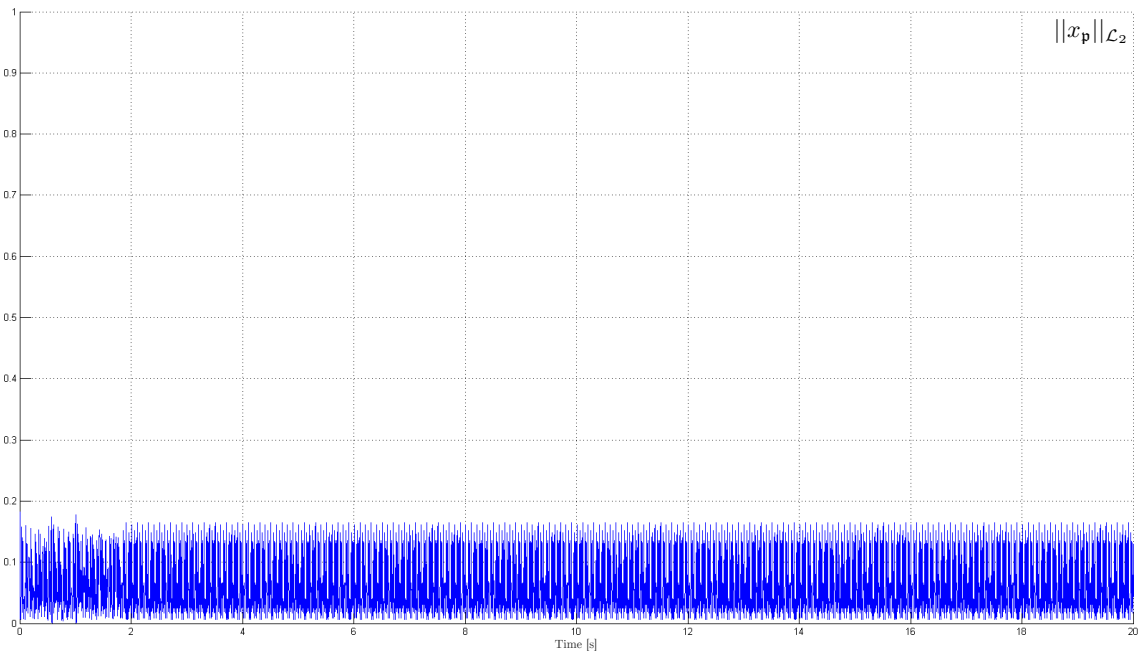


Figure 7.9: Stability: Euclidean norm of plant state

Robustness – Partial linear gain ℓ_2 stability

Unlike for stability results, here we do not include scheduling results since they are almost identical in a sense that scheduling only occurs in first two seconds. Namely, again, input $u_{\mathfrak{p}_2}$ takes over. Here, in Fig. 7.12 we present Euclidean norm of plant disturbance which is followed with an indication of packet dropouts through updating contents of

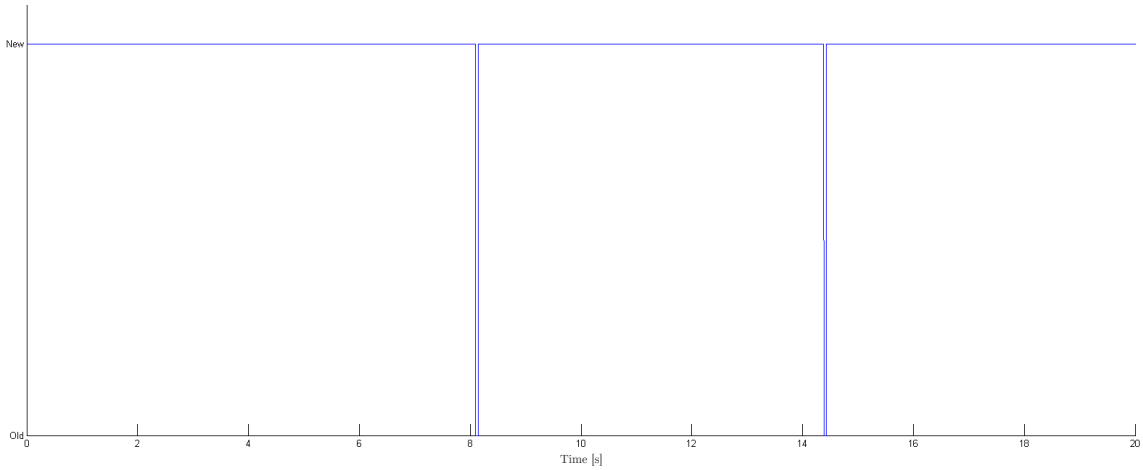


Figure 7.10: Stability: indication of packet dropouts through updating contents of buffer located before input $u_{\mathfrak{p}_2}$

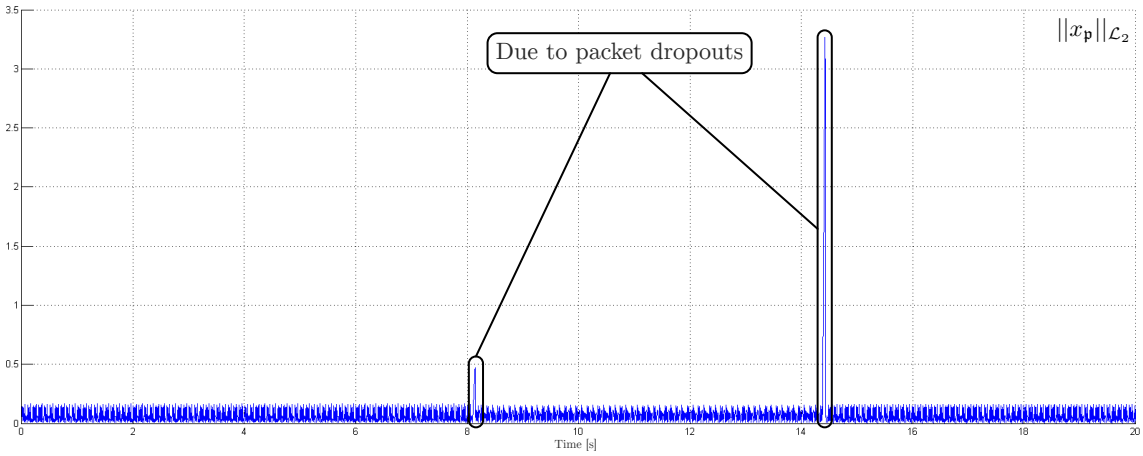


Figure 7.11: Stability: Euclidean norm of plant state due to packet dropouts

buffer located before input $u_{\mathfrak{p}_2}$ in Fig. 7.13. Finally, in Fig. 7.14 we show the corresponding plant response.

Similarly, as for stability results, and as indicated in Fig. 7.14, the performance is slightly deteriorated due to quantization issues, see Section A.3.2.

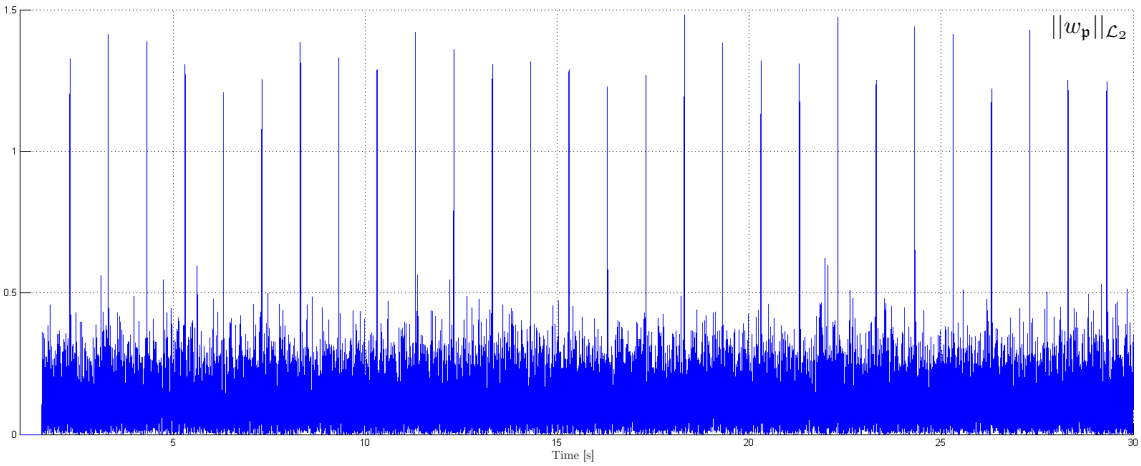


Figure 7.12: Robustness: Euclidean norm of plant disturbance

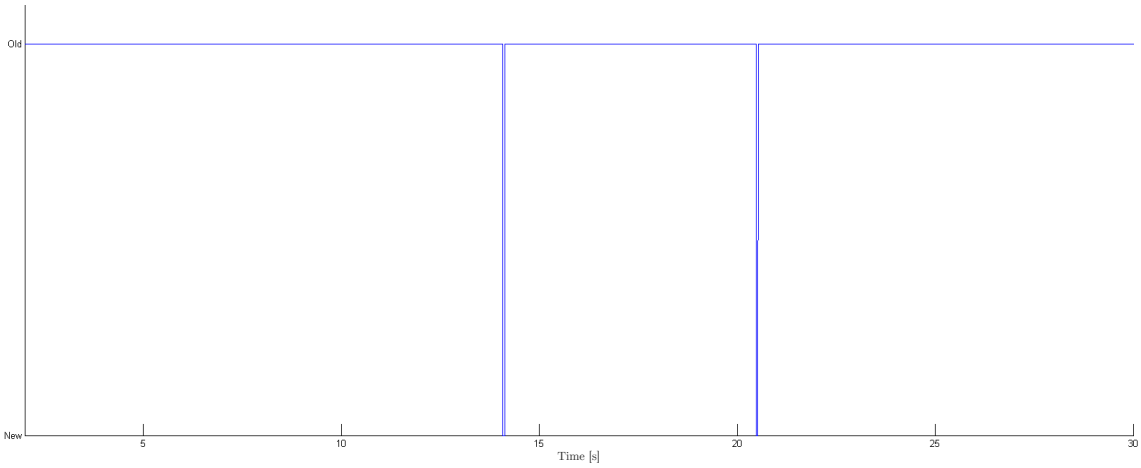


Figure 7.13: Robustness: indication of packet dropouts through updating contents of buffer located before input $u_{\mathfrak{p}_2}$

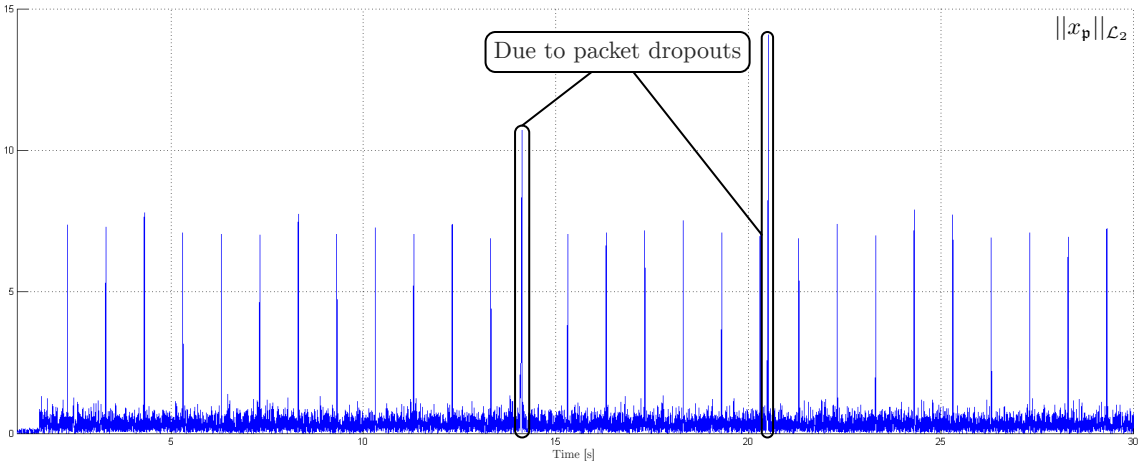


Figure 7.14: Robustness: Euclidean norm of plant state

This page intentionally left blank.

Chapter 8

Conclusion and future work

We focused on a specific NCS architecture. In particular, we considered only the case where the network is located between the controller output and plant input, e.g., see Fig. 2.5. Furthermore, we focused only on two network induced communication issues, namely, the issue of scheduling and the issue of packet dropouts. We addressed these communication constraints by carrying out a protocol and controller co-design, e.g., see [30]. This co-design entails exploitation of the flexible architecture of NCSs which allows for distributed computation and the usage of MPC framework.

We first investigated the stability property, more precisely, UGAS of the augmented state of the plant and buffer state. This result is then extended to the case where the corresponding NCS is governed with an Economic MPC. The following investigation deals with the robustness property where we establish several results. Namely, we establish partial nonlinear and linear gain ℓ_2 stability, we recover and strengthen the main result from [30], we provide an alternative robustness characterization of the NCS considered in [31] and we establish ISS of the augmented state of the plant and buffer state. This is followed by considering the controllability property where we introduce an interesting model and we establish several results. Finally, we finish with the implementation in which we confirm our stability and robustness expectations.

There are several very interesting future directions outlined in the sequel.

8.1 NCS architecture

The most interesting future direction is to close the whole loop over the network, e.g., sending plant measurements over a network as well. Indeed, this would render the current analysis and design much harder. For instance, notice that in order for an MPC controller to operate "well" it needs to have access to current plant measurements. Additionally placing a network between plant's output and controller's input would definitely affect that. Thus, perhaps a different approach might have to be considered. Results and insights documented in [21, 94] would definitely be used in the pursuit of extending our results to a more general NCSs.

It would be interesting to replace buffers with devices with more processing power, e.g., simple predictors. This would lead to true distributed computing which in the case of closing the whole loop over the network, might be sufficient to still use MPC framework.

Another future direction is consideration of NCS architecture in which a network is *only* located between plant's output and controller's input. This architecture, if investigated properly, can help in the case of closing the whole loop over the network. Moreover, notice that even in this simpler case we would still encounter the issue of the availability of the current plant measurements for an MPC controller.

8.2 Network

Another direction would be the consideration of richer network models. In particular, inclusion of the network delays and the quantization issues in the presented framework. As shown in the implementation section these two issues can not be avoided, at least in the case of CAN bus. Indeed, input delays for linear systems can be addressed by enlarging the dimension of plant dynamics and quantization can lead to practical stability. However, an explicit investigation of these two additional issues within proposed framework would lead to a more applicable result.

8.3 Analysis and design

Here we have several directions. First direction could be establishing all results with respect to output measurements (e.g., not full state measurements). Another one is explicit investigation of recursive feasibility. At the moment we are assuming the necessary assumptions that ensure it; e.g., see Assumption 5.8. Addressing this directly would make the presented framework more attractive.

Applying stochastic tools to the presented framework for stability with respect to packet dropouts and scheduling for standard MPC can be another direction. Recall that dropouts are random and incorporating that via dropout averages instead of a finite upper bounds in the present analysis would be very interesting. Moreover, this would result in a deeper understanding of the corresponding NCSs.

Another direction would be the relaxation of the assumptions presented in the robustness analysis. Note that we are using continuity assumption for the optimal value function and relaxing that would be very useful since it would apply to a larger class of systems.

Further, extending our stability results to Economic MPC with periodic terminal constraints and average constraints is another interesting directions. This is in fact very likely to be materialized soon since we have done already a first step of extending our results to the basic Economic MPC.

Finally, with respect to our controllability results, one future direction would consist of including packet dropouts into the system.

8.4 Implementation

One direction would consist of developing necessary solvers for the corresponding optimization problems coming from MPC in the case of nonlinear plant dynamics. Note that at the moment we are using an MPC toolbox provided by the MATLAB® which either requires linear systems or linearizations of nonlinear ones. Another direction is testing our stability and robustness results over FlexRay network which has much larger payload which can lead to improved performance because of smaller impact of quanti-

zation. Finally, testing our results over Ethernet and/or wireless network is yet another future direction.

Appendix A
Implementation details

The necessary information related to implementation is provided. We begin by expanding on HIL simulations mentioned in the introduction of Chapter 7. Then, we provide more information about CAN bus which is followed with a more detailed simulation setup description. Finally, we finish by providing specific settings of CAN-related Simulink® blocks, e.g., RTICANMM blocks.

A.1 Hardware-in-the-loop simulation

HIL simulations belong to a class of the *real-time* simulations, e.g., see [176]. Real-time simulations are a class of simulations in which the *simulated* system generates the *same*, time-dependent, input and output signal values as the real system.

Furthermore, real-time simulations usually imply that at least one part of the (overall) simulated system is actually real. More precisely, it is the actual hardware. In fact, this is why the other parts of the (overall) simulated systems must be simulated in real-time; e.g., their input and output, time-dependent, signal values must correspond to their real counterparts. Indeed, it is possible to completely simulate a system in the real-time, and this is done in certain occasions.

Thus, HIL simulation is a real-time simulation in which at least one part of the (overall) simulated system is *real*, i.e., the actual hardware. In control engineering practice this is usually the controller. However, there is really no strict rule which part of the overall control system is real and which one is simulated in real-time. For instance if a control system consists of a controller, a network, an actuator, a plant and a sensor, then, there

are 32 possible scenarios of which part is real and which one is simulated in real-time. Of course, the cases where all parts are simulated or real, are not "true" HIL simulations. The former one is "software-in-the-loop" simulation while the latter one is the actual realization of the corresponding system.

HIL simulations have became a necessity when a complex system is being developed; especially in control related areas such as transport (cars, areoplanes and ships), military applications, space exploration and many others. There are many reasons why HIL simulations are so omnipresent, spanning many fields such as development time, cost and safety. Some specific reasons are provided in the sequel:

- Design and testing of the control hardware and software without the need to operate the real system,
- Testing of the control hardware and software under extreme environmental conditions,
- Testing of the effects of faults and failures of system's components,
- Operating and testing of extreme and dangerous operating conditions,
- Reproducible experiments,
- Easy operation with different man-machine interfaces,
- Saving of cost and development time (one of the most important reasons).

For a more detailed exposition on HIL simulations we refer the reader to [176] and references therein.

A.2 Control Area Network

Control Area Network (CAN) is a type of a much larger class of communication networks, namely, the serial bus. In particular, the serial bus is a communication channel

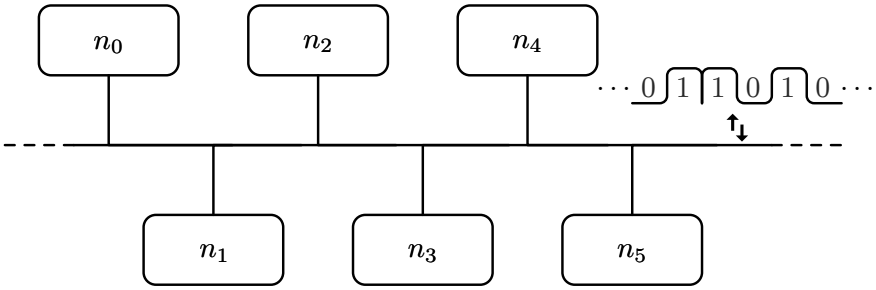


Figure A.1: Serial bus networking; node $n_{\cdot}$ is an abstraction for a communication participant, e.g., a plant.

on which an information is communicated in a form of bit-serial exchange of the corresponding data; e.g, this is a process of sequentially transmitting the corresponding data *one* bit at a time, see Fig. A.1.

There is an extensive literature on CAN, for instance, see [177, 178] and references therein or on-line resources, such as the ones provided by Vector Informatik®; cross-referencing Vector Informatik with CAN in Google® search engine will result in the corresponding resources. In fact, in the sequel, we use the corresponding on-line resources (`https://elearning.vector.com/vl_index_en.html`) to provide the basic information about CAN. So, let us begin.

In the year 1983, the International Standardization Organization (ISO) standardized the implementation of a serial data communication process with the Open System Interconnection (OSI) communication model, see Fig. A.2. We notice that including all detail about the OSI 7 layer communication model is not necessary for our purposes; however, those interested in the details can start by consulting [177, 178] and references therein. For our purposes, we borrow an illustration provided by Vector Informatik® which succinctly captures the basic principles of communication between two 7-layer nodes (in a Peer-to-Peer communication fashion).

As illustrated in Fig. A.3, layer 2 - layer 7, in the sender node, adds an extra information (e.g., PCI - x, $x \in \{2, \ldots, 7\}$) to the actual data (i.e., Payload), which forms the data frame that is sent over a physical transmission medium. Notice that addition of this extra data enables a layer in the sender node to communicate with the corresponding layer in

Layer	Data unit	Function
7. Application	Data	High-level application programming interface, including resource sharing, remote file access, directory services and virtual terminals...
6. Presentation		Translation of data between a networking service and an application; e.g., data compression, encryption/decryption...
5. Session		Managing communication sessions, i.e. continuous exchange of information in the form of multiple back-and-forth transmissions...
4. Transport	Segments	Reliable transmission of data segments between points on a network, including segmentation, acknowledgement and multiplexing...
3. Network	Packet/ Datagram	Structuring and managing a multi-node network, including addressing, routing and traffic control...
2. Data link	Bit/Frame	Reliable transmission of data frames between two nodes connected by a physical layer...
1. Physical	Bit	Transmission and reception of raw bit streams over a physical medium...

Figure A.2: OSI 7 layer communication model.

the receiver node; e.g., layer 3 in sender node communicates with layer 3 in receiver node. A detailed description of the latter procedure can be found, for instance, in [177, 178].

Since we concentrate on CAN, we need mention that it does not uses all 7 layers to communicate but 3. In particular, it uses Data Link Layer and the Physical Layer while the functions of unconsidered layers are usually assigned to Application Layer, see Fig. A.4.

Before presenting relevant information about CAN let us acknowledge that it was a result of the research initiated by Bosch® in the year 1983 to develop a communication system tailored for the automotive industry. Now, the most important resource in serial bus communication is the access to the bus. In CAN, all nodes have equal access to the bus rights or the corresponding interaction between the nodes is said to be the

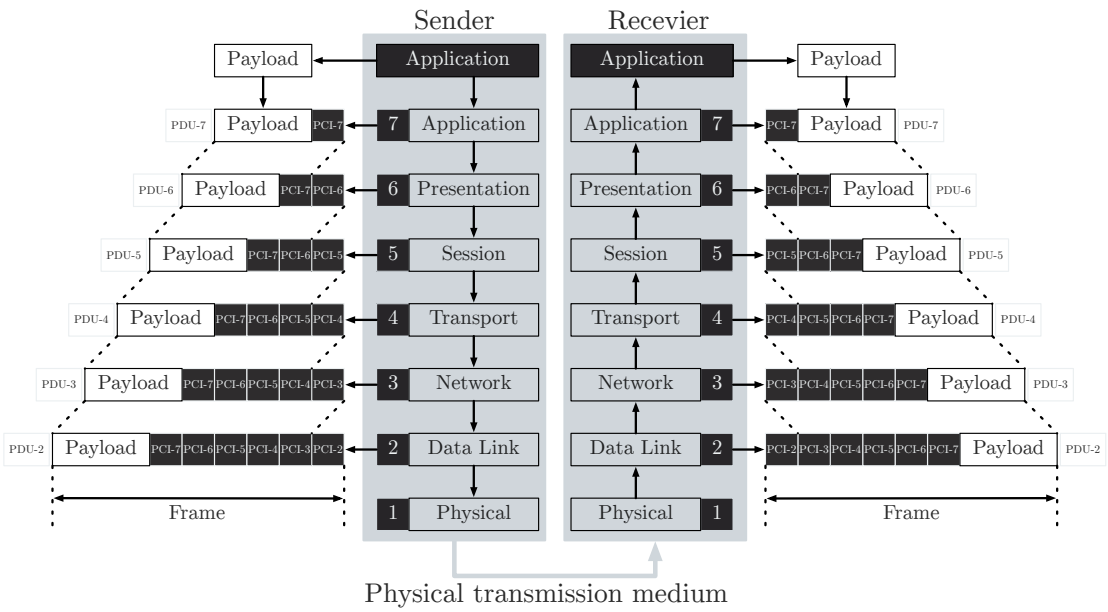


Figure A.3: Basic communication principles in OSI 7 layer communication model; PCI - Protocol Control Information, PDU - Protocol Data Unit.

Layer	Function
7. Application	Functions of unconsidered layers...
2. Data Link	Addressing, message building (Framing), bus access, synchronisation, error detection and error correction...
1. Physical	Description of the physical bus interface and conventions for physical signal transmission...

Figure A.4: Three Layer model.

Multi-Master Interaction; e.g., each node is a master as opposed to a slave in the Master-Slave Interaction which needs the permission from a master to access the communication medium. This is prerequisite for implementing communication on demand which is based on event-driven bus access methods such as the CSMA/CA method (Carrier Sense Multiple Access with Collision Avoidance). The central aspect of serial bus systems with demand-based interaction is that any bus node can access the common transmission medium at any time. Unfortunately, with increasing bus loads the risk of message collisions increases. In CAN this risk is countered by the priority-driven CSMA/CA method. However, this method does not prevent delays in transmission of low priority messages with increasing bus loads, and in the worst case scenario they may be blocked.

Physically, all communication participants (e.g., bus nodes) are connected passively to a common transmission medium (bus). This means that all data reaches all bus nodes, making a serial bus topology implicitly a diffusion network or a broadcast system. Significant advantage of this topology is that it permits any desired logical interaction structure but, unfortunately, its bus length and number of bus nodes are limited. Moreover, long electrical lines must be terminated by a so-called "characteristic impedance" to prevent signal reflections at the line ends. Also, note that a break in the transmission medium prevents further communication; see Fig. A.1.

Data communication in a serial bus system is performed by so-called frames. As indicated in Fig. A.3, the frame (message), besides containing the actual useful data (Payload), also contains other information (e.g., PCI - ·). This exta information is used to: produce a unique assignment between the useful data and the bus node (e.g., addressing); provide communication partners with information for synchronization; and provide information for protecting the useful data. Let us borrow the exposition from Vector Informatik® on-line resources on a data frame that is used for data transmission in CAN, see Fig. A.5. Namely, the data frame begins with a synchronization bit, the so-called start bit (Start Of Frame - SOF). This is followed by the frame's Identifier. The next bit, the RTR bit (Remote Transmission Request), indicates the frame type (Data or Remote frame). The sender uses the IDE bit (Identifier Extension) that follows to indicate the frame format (standard or extended format); in standard format the identifier comprises 11 bits, while

in extended format it is made up of 29 bits. This is followed by the DLC (Data Length Code), which indicates the number of useful bytes. Only then does the data field begin, which comprises a maximum of *eight* useful bytes. Next there is the CRC (Cyclic Redundancy Check) sequence that is used to protect all information and useful data. Before the terminating EOF (End Of Frame) symbol there is the ACK field (Acknowledgement). Before proceeding with the information on bus access let us emphasize that there is only *eight* useful bytes for the actual (useful) data; in our case this will lead to quantization issues since we are sending a sequence of control values.

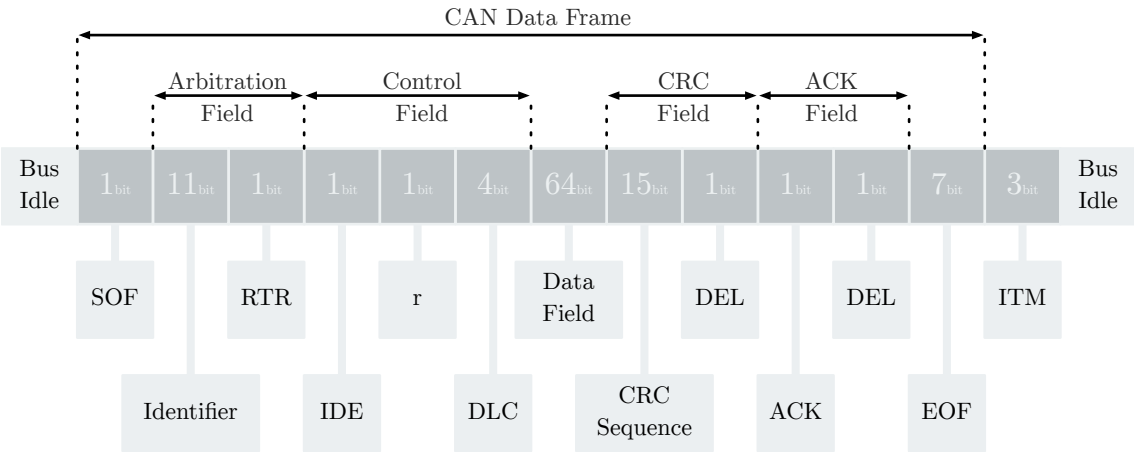
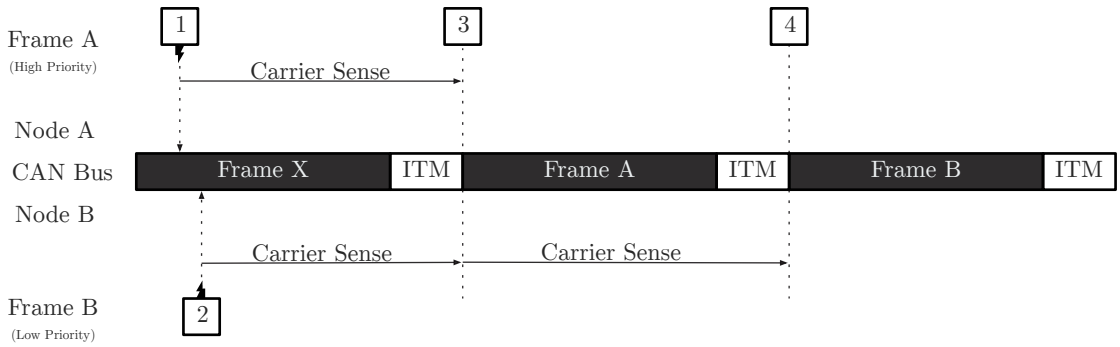


Figure A.5: CAN data framing.

As mentioned above, all CAN nodes have equal rights. Thus, the right to bus access in CAN is not pre-determined which makes possible to grant each and every bus node access to the bus as needed. This induces the probabilistic nature to the corresponding bus access method. The unpredictability of bus access increases the risk of collisions with increasing bus load. Such problems may be alleviated by various approaches such as the CSMA/CA method. To prevent an ongoing data transmission from being destroyed, bus nodes wishing to send must sample the bus before accessing it (Carrier Sense). A node may only send if the bus is idle.

Bus access by the CSMA/CA method enables very quick reactions to events. However, it still suffers the fundamental risk that multiple bus nodes might wish to access the bus at the same time. This on the other hand can result in collisions. To prevent



Event	Description
1	Node A wants to access the Bus but because it is busy Node A has to wait for ITM to expire, hence, Node A monitors (Carrier Sense) the Bus to detect this.
2	Node B wants to access the Bus but because it is busy Node B has to wait for ITM to expire, hence, Node B monitors (Carrier Sense) the Bus to detect this.
3	After ITM expires the Bus becomes idle and Node A is granted the access since its message (Frame A) has higher priority; Node B waits for the Bus to be idle again.
4	After ITM expires the Bus becomes idle and Node B is granted the access to transmit its message (Frame B).

Figure A.6: Principle of CAN bus access; ITM - Intermission.

collisions, to each message a corresponding priority is assigned. Then, with simultaneous bus accesses, the CSMA/CA method employs bitwise bus arbitration to ensures that the bus node with the highest priority frame obtains bus access. Roughly speaking, the higher the priority of a frame, the sooner the frame can be transmitted on the bus. The node losing arbitration makes another attempt to access the bus as soon as the bus becomes available again, see Fig. A.6. Note that if the bus load is not too high, this type of random, nondestructive and priority-driven bus access offers fair and quick bus access. However, low-priority frames are delayed with increasing bus load. In the worst case scenario, this can lead to lack of real-time capability. Moreover, in the case of poor system design, there is even a risk that low-priority frames might are be transmitted.

We finish with a note on data protection. One of the most effective physical data protection measures is symmetrical signal transmission over a twisted pair line. On the other hand, logical data protection is based on five error detection mechanisms: sending CAN nodes compare each sent bit level with the actual bus level (bus monitoring);

they also evaluate the acknowledgments of the CAN nodes (ACK check); receiving CAN nodes check each data frame for accuracy based on the arriving CRC sequence (Cyclic Redundancy Check); they then inform the sender of the results in the form of a positive or negative acknowledgment; finally, receivers check the specified format (Form check) and conformance to the bit stuffing rule (Stuff check). As soon as a CAN node discovers a transmission error, it stops transmitting data and immediately transmits an error signal (error flag). This causes all CAN nodes to detect a bit stuffing error. Data consistency is assured, because all CAN nodes then abort the data transmission with an error flag. As soon as the CAN bus is available again, the sender repeats the aborted data frame.

We remind the reader that we have omitted a considerable amount of information about serial bus and CAN since we believe its inclusion would considerably distract the reader; for those who wish to learn more about the missing topics please see, for instance, see [177, 178] and references therein and many on-line resources provided by manufactures of related hardware and software, e.g., Vector Informatik®, dSPACE®, etc.

A.3 Simulation setup

As mentioned in Section 7.3, we include only *some* parts related to implementation (HIL simulation) setup. Recall that we take this approach since inclusion of all the details would require to include extensive information from several help and guide manuals form different companies; i.e., MathWorks® (MATLAB®, Simulink®, MPC toolbox, etc.), dSPACE® (hardware, Simulink® RTICANMM toolbox, ControlDesk Next Generation®) and Vector Informatik® (CANdb++ editor). As outlined in Section 7.3, the setup procedure includes:

1. Setting up dSPACE® simulator and establish working connection with the corresponding workstation (e.g., a PC),

2. Choosing the plant you wish to control,

3. Designing CAN structure using Vector Informatik® CANdb++ editor,

4. Making the corresponding physical connections that resemble the designed CAN structure,
5. Designing MPC controller using MATLAB® MPC toolbox,
6. Designing the corresponding NCS model using Simulink®, generating the corresponding C-code and downloading it onto dSPACE® simulator hardware,
7. Designing the layouts in dSPACE® ControlDesk Next Generation which will capture the corresponding test results and running the tests.

Except for the first item we will expand upon the rest further in the sequel. The reason we omit providing more detail on the first item is because hardware setting up (e.g., installing and interconnecting the corresponding boards within dSPACE® Simulator) was done by employees from dSPACE®. Furthermore, installing the corresponding software and interconnecting dSPACE® Simulator with a working station (i.e., a PC) is described in the corresponding dSPACE® manuals.

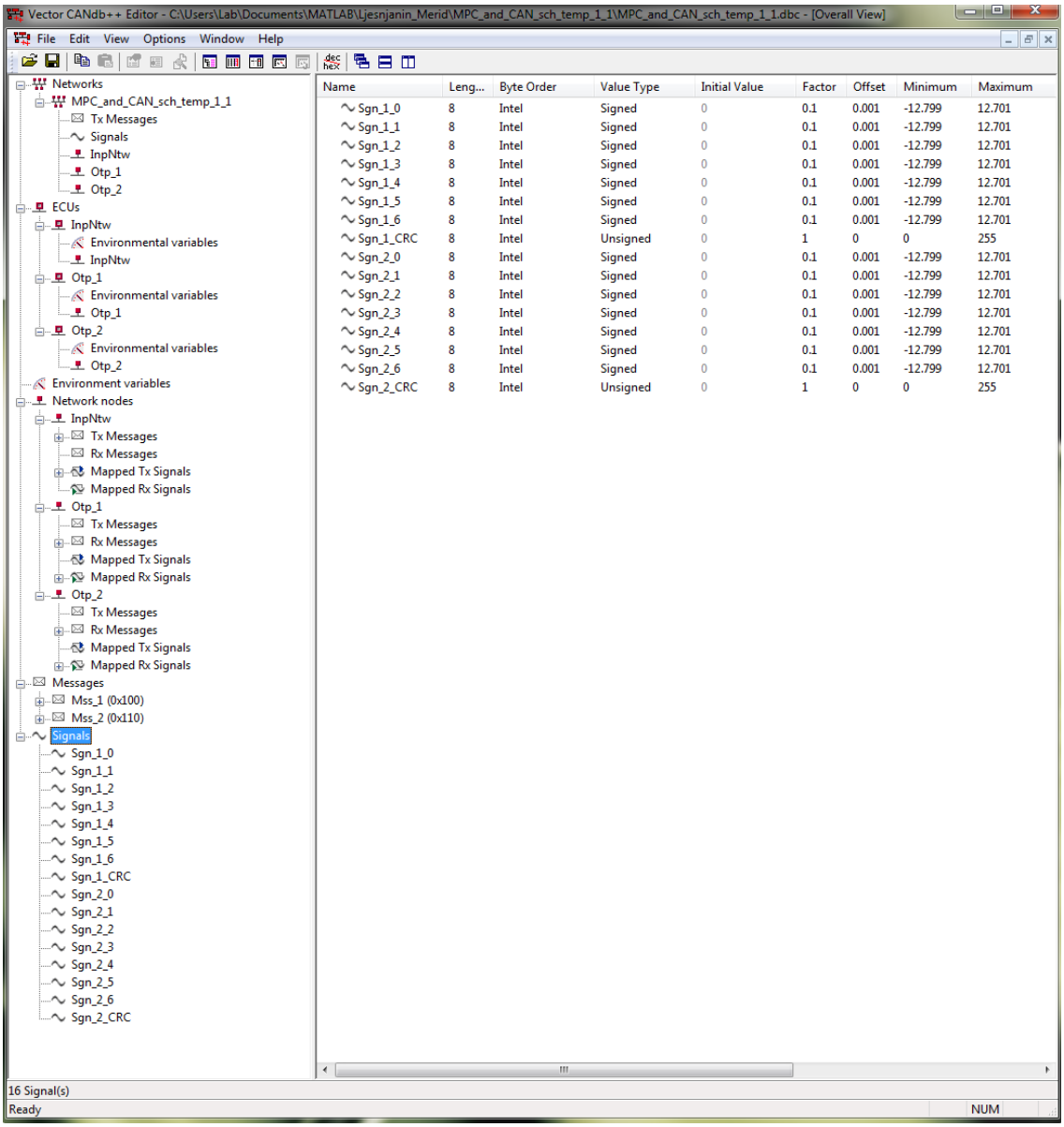
A.3.1 Considered plant

As mentioned in Section 7.1.1, we restrict our attention to linear plants (7.1) because MATLAB® provides an MPC toolbox which ca be used to design the corresponding MPC controller.

A.3.2 Designing CAN structure

We consider a plant with two inputs which means we will need two CAN nodes which will receive the corresponding messages. Furthermore, due to scheduling only one plant input can be accessed at each time instant which means we only need one CAN node which will send the corresponding control values. So, we have two receiver CAN nodes corresponding to plant inputs and one sender CAN node corresponding to controller output. For the sake of simplicity, we will not add more nodes to the CAN structure.

One of the software packages that can be used to design CAN structure is Vector Informatik® CANdb++ editor, see Fig. A.7.



Name	Leng...	Byte Order	Value Type	Initial Value	Factor	Offset	Minimum	Maximum
Sgn_1_0	8	Intel	Signed	0	0.1	0.001	-12.799	12.701
Sgn_1_1	8	Intel	Signed	0	0.1	0.001	-12.799	12.701
Sgn_1_2	8	Intel	Signed	0	0.1	0.001	-12.799	12.701
Sgn_1_3	8	Intel	Signed	0	0.1	0.001	-12.799	12.701
Sgn_1_4	8	Intel	Signed	0	0.1	0.001	-12.799	12.701
Sgn_1_5	8	Intel	Signed	0	0.1	0.001	-12.799	12.701
Sgn_1_6	8	Intel	Signed	0	0.1	0.001	-12.799	12.701
Sgn_1_CRC	8	Intel	Unsigned	0	1	0	0	255
Sgn_2_0	8	Intel	Signed	0	0.1	0.001	-12.799	12.701
Sgn_2_1	8	Intel	Signed	0	0.1	0.001	-12.799	12.701
Sgn_2_2	8	Intel	Signed	0	0.1	0.001	-12.799	12.701
Sgn_2_3	8	Intel	Signed	0	0.1	0.001	-12.799	12.701
Sgn_2_4	8	Intel	Signed	0	0.1	0.001	-12.799	12.701
Sgn_2_5	8	Intel	Signed	0	0.1	0.001	-12.799	12.701
Sgn_2_6	8	Intel	Signed	0	0.1	0.001	-12.799	12.701
Sgn_2_CRC	8	Intel	Unsigned	0	1	0	0	255

Figure A.7: CAN structure designed in Vector Informatik® CANdb++ editor

As captured in Fig. A.7, we added three nodes. Node "InpNtw" is the node used for transmitting the sequence of control values, while nodes "Otp1" and "Otp2" are nodes used for receiving the corresponding sequence. Further, we created two messages "Mss_1" and "Mss_2" which will (depending on scheduling) be sent by node "InpNtw" to nodes "Otp1" and "Otp2", respectively.

Now, recall that CAN message (frame) has only eight bytes of useful data. In this soft-

ware package one exploits this resource via "Signals". We have decided to dedicate first seven bytes for control values while the last byte is needed for CRC-checksum results, see the corresponding algorithm code in Listing A.1; notice that this is a template provided by dSPACE® which is accordingly modified. We have eight signals in each message, e.g., Sgn_1_0" to "Sgn_1_7", "Sgn_1_CRC" in message "Mss_1" and "Sgn_2_0" to "Sgn_2_7", "Sgn_2_CRC" in message "Mss_2". First seven correspond to control values and the last one is used in CRC .

Listing A.1: CRC-checksum code

```
/*************************************************************************/
/*   Automatic generated File from RTI CAN MultiMessage Blockset:        */
//     CRC.h                                                             */
/* INPUT PARAMETER(S)                                                    */
/*          (1) option : crc / crccheck                                  */
/*                  0: TX  (calculate)                                   */
/*                  1: RX (check)                                        */
/*          (2) Pointer to MsgData                                       */
/*          (3) Index of CRC see array in rticanmm_customer              */
/*          (4) StartBit of ChecksumSignal                               */
/*          (5) Length of ChecksumSignal                                 */
/*                                                                       */
/* OUTPUT PARAMETER(S)                                                   */
/*          (1) crccheck  (RX)                                           */
/*   DESCRIPTION:                                                        */
/*     defines customer-specific CRCs                                    */
/*   AUTHOR(S):                                                          */
/*     generated by dSPACE(R) RTICANMM                                   */
/* *********************************************************************/
static UInt8 rticanmmcrc(int crcoption, RTICANMMMsgStruct* Msg,
                                 int crctype, int CsBitPos, int CsLength)
{

 UInt32 crc = 0;
 UInt8 CRCResult = 0;
```

```
 int i=0;
 int BytePos=0;

 // Msg_Length = Msg->len;
 // Id         = Msg->identifier;

 switch (crctype)
   {
    case 1:     // CRConByte7
    case 2:     // CRConByte7_false
            BytePos=7;
            break;
   }

/* CRC calculation */
 for (i = 0; i<Msg->len; i++)
  {
    if(i != BytePos) /* exclude crc byte */
      crc = 0xFF & (crc + Msg->RAW_DATA[i]);
  }

switch(crctype) {
  case 1:{     //CRConByte7

   // Your CRC-Calculation here: Example:

    crc = 0xFF & crc;

  }; break;

  case 2:{     //CRConByte7_false

   // Your CRC-Calculation here: Example:

    crc = 0xFF & (crc + 1);
    if(crc > 254)
     {
```

```
        crc = 0;
      }

  }; break;

 default: {return(1);}
}

CRCResult = (UInt8)(crc);

 if(crcoption==0){
    // Return of CRC (TX)
    Msg->RAW_DATA[BytePos]=CRCResult;
    return(0);
 }
 else{
    // Return of CRC-Check (RX)
    return (Msg->RAW_DATA[BytePos]!=CRCResult);
 }

}
```

Due to data types (e.g., Signed, Unsigned, Float, Double etc.) and the needed corresponding number of bytes, the seven bytes we were left with gave us few options to choose from. Note that, for instance, the range for Signed (Factor 1, Offset 0) is -128 to 127 and one needs only one byte. On the other hand, for instance, the range for Double is $-1.7 \cdot 10^{308}$ to $1.7 \cdot 10^{308}$ (precision of 15 decimal places) and one needs eight bytes. So, in order to have the longest MPC horizon possible, we have decided to represent with each byte one control value. Note that this upper bounded MPC horizons with seven. Then, we chose the Value Type to be Singed, the Factor of 0.1 and the Offset of 0.001 giving us the range with the minimum value -12.799 and the maximum value 12.701. This induces the issue of quantization which in turn induces the issue of saturation, which is not accounted for theoretically. However, when control values are within the range where saturation does not occur, i.e., between -12.799 and 12.701, the results are still relatively good which is promising and it motivates one to consider the issue of quanti-

zation explicitly within the considered framework; e.g., perhaps adaptive quantization is a promising direction to mitigate the effects of quantization.

Indeed, our results were tailored for networks which have more useful data like FlexRay or Ethernet network. However, testing on CAN revealed the effects of quantization issues which can lead to instability. As stated above, this provides additional incentives for considering this issue directly in future work.

A.3.3 Physical realization of CAN structure

Within dSPACE® Simulator there are eight CAN boards. Each board provides four CAN channels (nodes), see Fig. A.8. Since we need only three nodes, one CAN board is used.

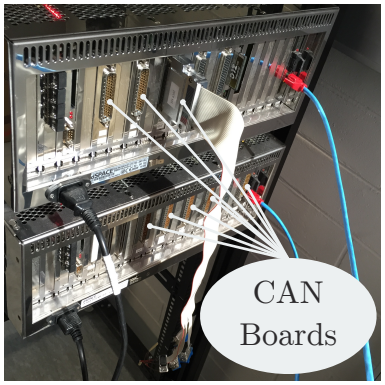


Figure A.8: dSPACE® Simulator - back view.

Each CAN node (channel) has 9 pins, see [179]. Generally, in order to establish serial bus topology one really needs to be careful which pins have to be connected and how. However, since we will not connect a real plant (i.e., the corresponding sensors) nor an actuator, establishing the serial bus topology, see Fig. A.1, simplifies. In particular, it translates to connecting all CAN_High pins together and all CAN_Low pins together, see Fig. A.9.

A.3.4 Designing MPC controller(s)

Mathematical derivations are provided in Section 7.1.4 and here we only provide the corresponding MATLAB® script code. Notice that we use "InputDelay" when defining

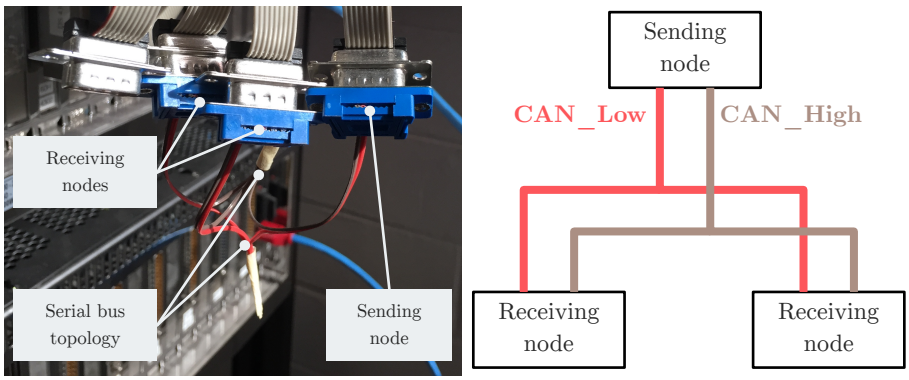


Figure A.9: Physical connections resembling the designed CAN connection structure.

plant models. This is yet another benefit of implementation. Namely, we did not consider delay that network induces but it cannot be avoided. Luckily, it was easy to include the corresponding delay and address it successfully in MPC toolbox. A detailed investigation of the issue of delay within the considered protocol and control co-design is left for future work.

Listing A.2: MATLAB® script code

```
%% MPC controllers

% Clear MATLAB workspace and terminal
    clear all
    clc

% Plant matrices
    A = [1., 1.; .0, 1.];
    B = [.0, .5; 1., .5];
    C = eye(2);
    D = zeros(2);
    W = eye(2);

% Shift matrix (note that buffer length is 7)
S = [diag([1 1 1 1 1 1], 1.), zeros(7); zeros(7), diag([1 1 1 1 1 1], 1.)];

% Matrices used in MPC model for first plant input, i.e., MPC1
```

```
    Z_u1 = [zeros(1, 14); zeros(1, 7) 1. zeros(1, 6)];
    R_u1 = [1. .0; .0 .0];
    A_u1 = [A, B*Z_u1; zeros([14, 2]), S];
    B_u1 = [B*R_u1; zeros(14,2)];
    C_u1 = eye(16);
    D_u1 = zeros(16, 2);

% Plant model for MPC1
    Model_u1 = ss(A_u1, B_u1, C_u1, D_u1, 0.001, 'InputDelay', [1; 1]);

% Matrices used in MPC model for second plant input, i.e., MPC2
    Z_u2 = [1. zeros(1, 13); zeros(1, 14)];
    R_u2 = [.0 .0; .0 1.];
    A_u2 = [A, B*Z_u2; zeros([14, 2]), S];
    B_u2 = [B*R_u2; zeros(14,2)];
    C_u2 = eye(16);
    D_u2 = zeros(16, 2);

% Plant model for MPC2
    Model_u2 = ss(A_u2, B_u2, C_u2, D_u2, 0.001, 'InputDelay', [1; 1]);

% Setting MPC data
% Sampling time
    Ts = 0.001;
% Horizon
    p=7;
% Length of output sequence
    m=7;

% MPC controllers
% MPC1
    MPC_u1 = mpc(Model_u1, Ts, p, m);
% MPC2
    MPC_u2 = mpc(Model_u2, Ts, p, m);
```

A.3.5 NCS model

The conceptual diagram is provided in Fig. 7.6. The corresponding Simulink® model that captures this concept is given in Fig. A.10 while in Fig. A.11 we provide other sub-models custom-made using standard Simulink® blocks. Note that standard Simulink® blocks are thoroughly covered in the corresponding MATLAB® help and user guide manuals (readily available either within MATLAB® software package or as a free on-line resource). Thus, we will omit the details on how sub-models in Fig. A.11 are generated and how they work. Similarly, network related blocks, namely RTICANMM blocks (see Fig. A.12), are exceptionally covered in [180]. Due to the volume of the corresponding material and complexity, in the sequel, we only provide screen-shots capturing settings of used RTI-CANMM blocks, see Section A.4; notice, we provide only setting that differ from default ones. Notice that generating the corresponding C-code is done within Simulink®. Once the code is generated it is automatically downloaded onto dSPACE® simulator hardware because the of item one of our setup procedure.

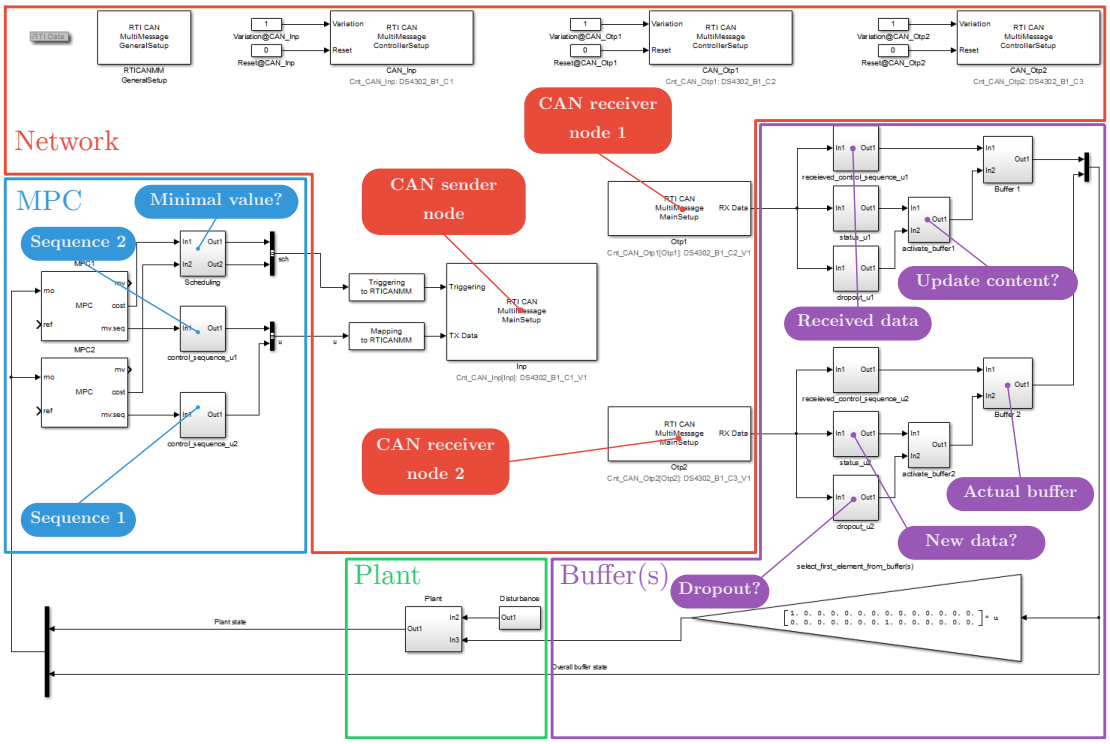


Figure A.10: Simulink® model of the implementation of the corresponding NCS.

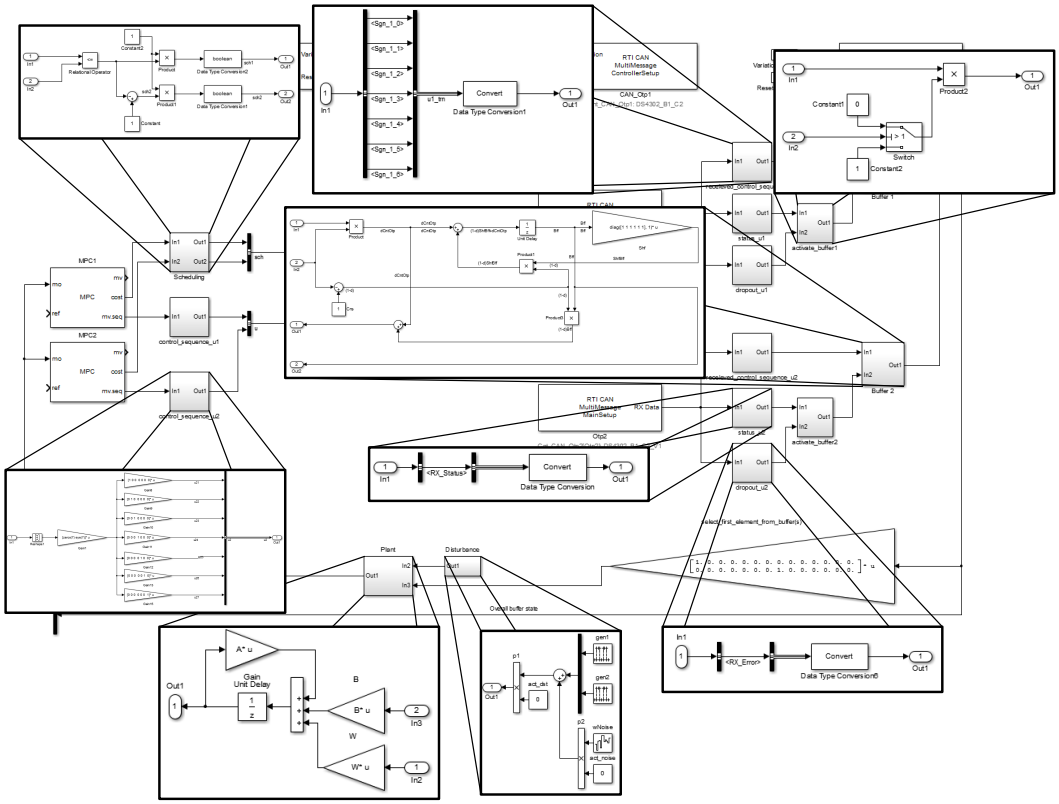


Figure A.11: Other custom made Simulink® sub-models.

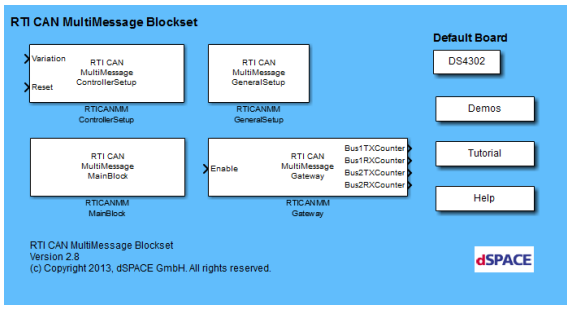


Figure A.12: dSPACE® RTICANMM Simulink® blockset.

A.3.6 ControlDesk Next Generation®

We only provide a screen-shot of a layout in ControlDesk New Generation®, namely, see Fig. A.13. Creating the needed layouts is quite easy and the "how-to" is covered

in the corresponding ControlDesk New Generation® help manuals (available within the software package).

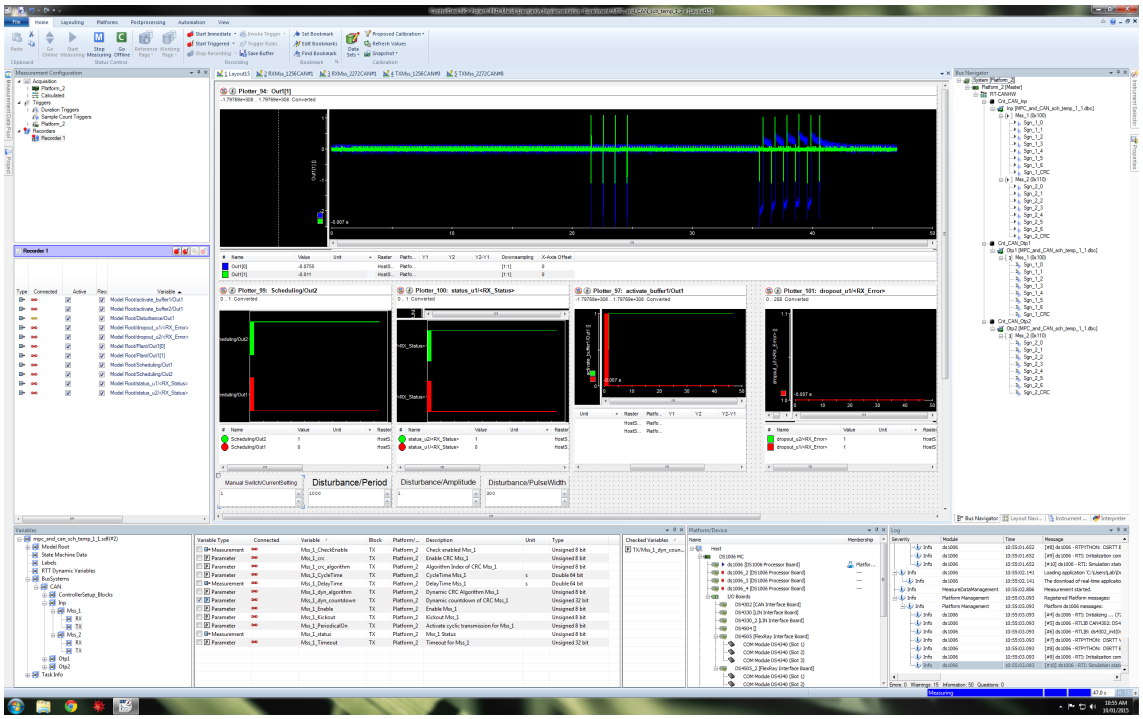


Figure A.13: dSPACE® ControlDesk New Generation®.

As indicated in Section 7.1.2, CAN bus uses a priority-driven CSMA/CA method to avoid packet dropouts (e.g., packet collisions). Since in our case only one node transmits a message, packet dropout is very unlikely to occur. Thus, we need to generate signals which when detected on real CAN bus hardware result in abortion of communication, e.g., packet dropout. Usual practice in automotive industry, see [180], is to write a "faulty" CRC algorithm which creates an incorrect checksum which receiving node recognizes as an error in communication, i.e., communication is aborted which is effectively a packet dropout (e.g., our buffer has to use values from its memory). The corresponding manipulation of the CRC algorithm is provided in Listing A.1. Moreover, we also provide a ControlDesk Next Generation® layout in which we choose this "faulty" CRC to be sent for certain amount of time, see Fig. A.14. Another method is much simpler, namely, during experiment run-time one blocks ("time-out") a sender node for a certain amount of time which results in packets not being received, i.e., effectively a packet dropout(s);

see Fig. A.14.

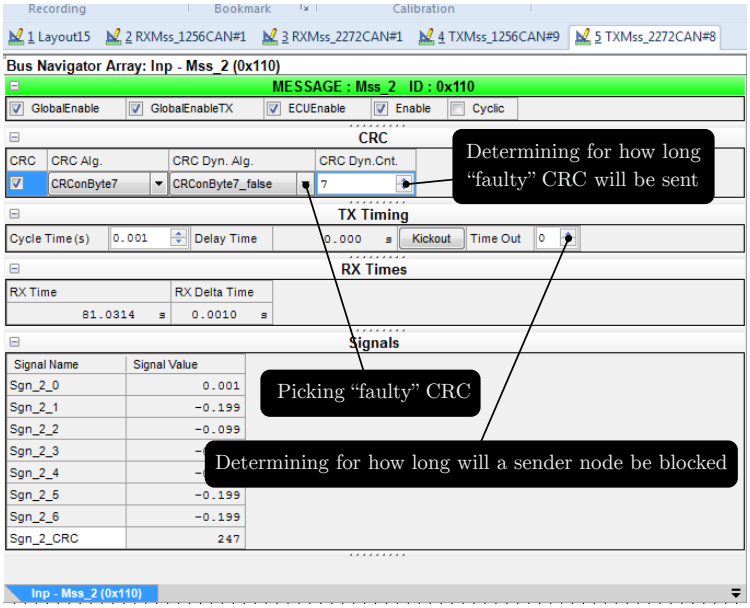
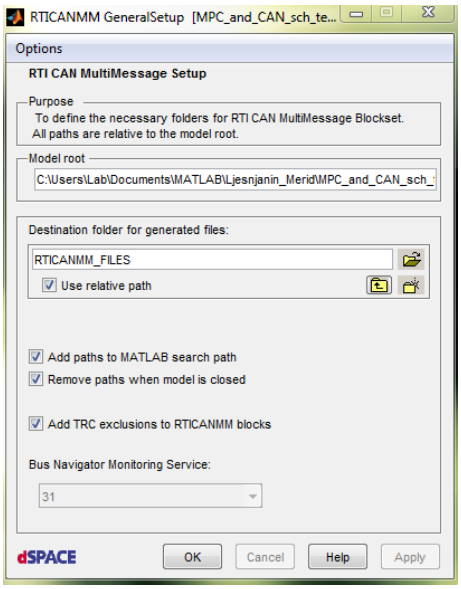
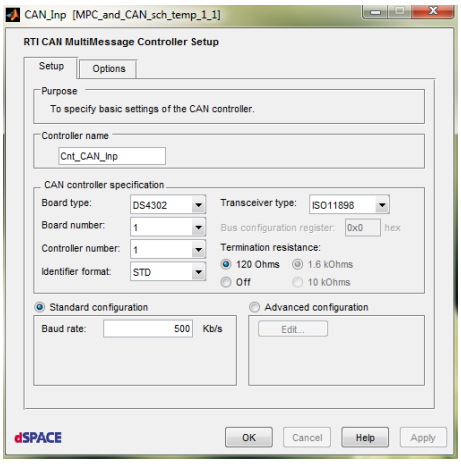


Figure A.14: Generating communication errors and/or misbehaviors which are effectively packet dropouts.

A.4 Specific settings of RTICANMM blocks

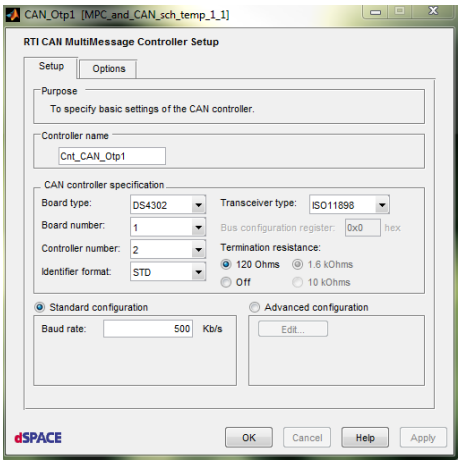


(a) RTI CAN Multi Message general setup.

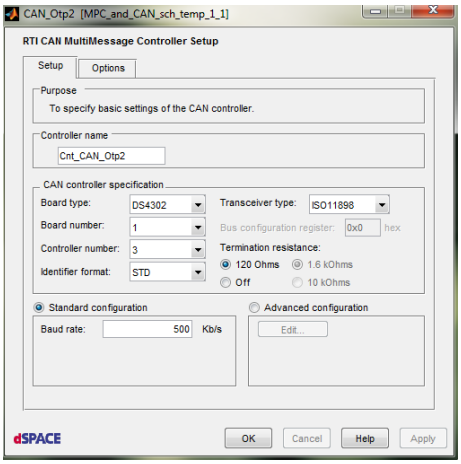


(b) RTI CAN Multi Message controller setup for a sender node.

Figure A.15: Specific settings of RTICANMM blocks

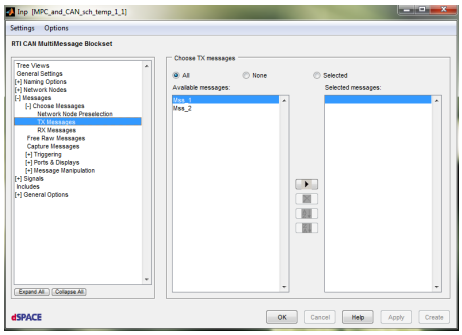


(a) RTI CAN Multi Message controller setup for receiver node 1 (e.g., plant input 1).

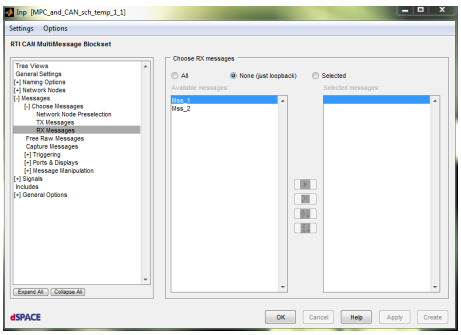


(b) RTI CAN Multi Message controller setup for receiver node 2 (e.g., plant input 2.

Figure A.16: Specific settings of RTICANMM blocks

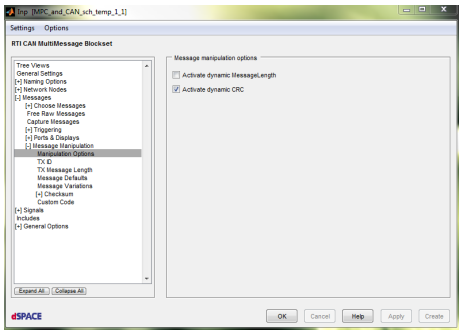


(a) RTI CAN Multi Message TX message(s) for a sender node.

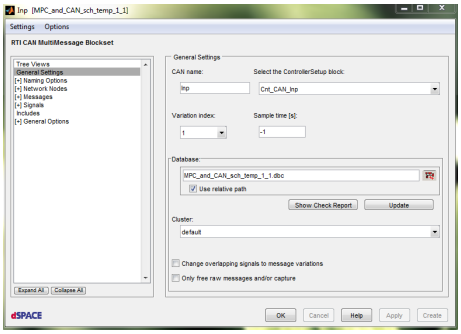


(b) RTI CAN Multi Message RX message(s) for a sender node.

Figure A.17: Specific settings of RTICANMM blocks

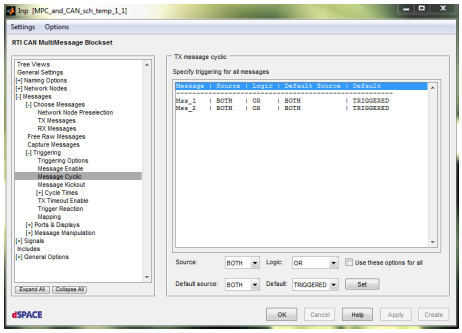


(a) RTI CAN Multi Message message manipulations for a sender node.

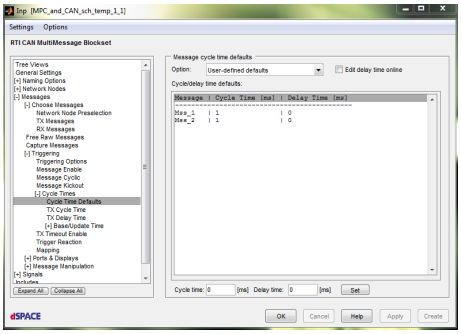


(b) RTI CAN Multi Message general settings for a sender node.

Figure A.18: Specific settings of RTICANMM blocks

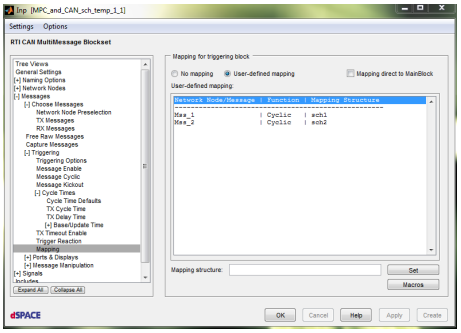


(a) RTI CAN Multi Message message triggering for a sender node.

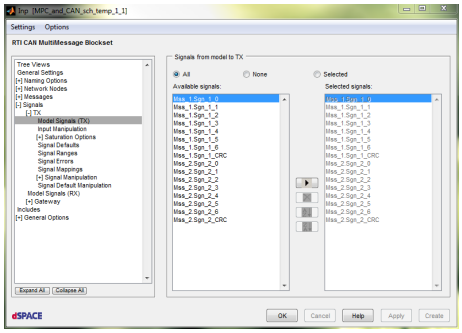


(b) RTI CAN Multi Message cycle time for a sender node.

Figure A.19: Specific settings of RTICANMM blocks

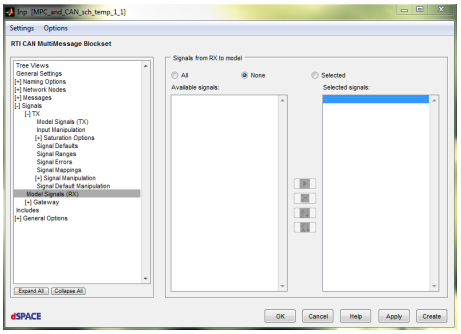


(a) RTI CAN Multi Message triggering mapping for a sender node.

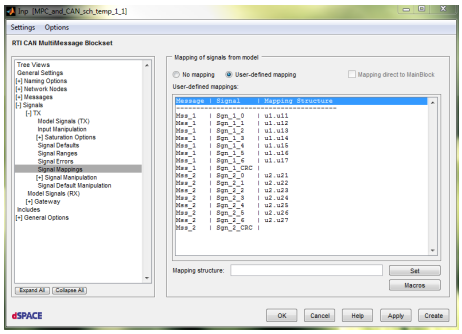


(b) RTI CAN Multi Message TX signals for a sender node.

Figure A.20: Specific settings of RTICANMM blocks

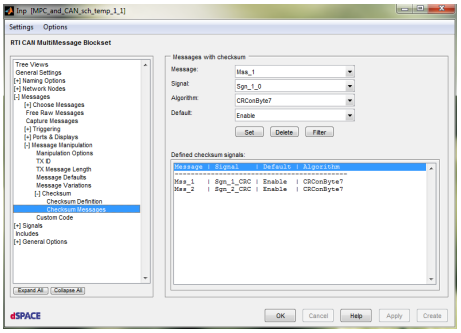


(a) RTI CAN Multi Message RX signals for a sender node.

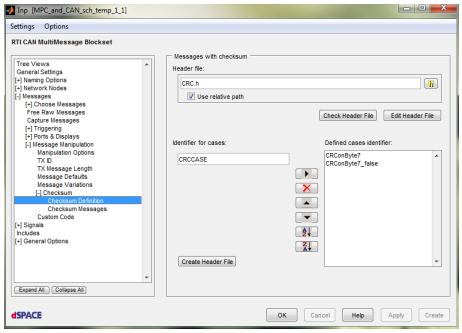


(b) RTI CAN Multi Message TX signal mappings for a a sender node.

Figure A.21: Specific settings of RTICANMM blocks

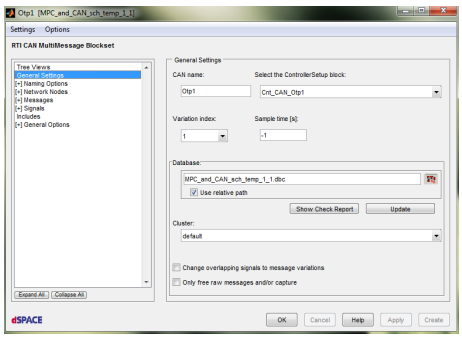


(a) RTI CAN Multi Message checksum messages for a sender node.

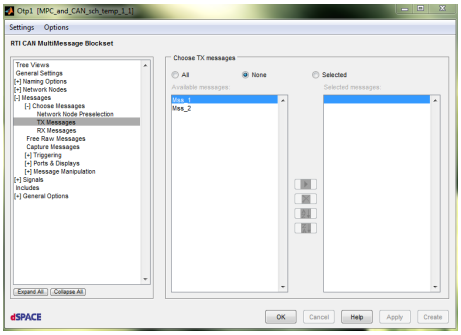


(b) RTI CAN Multi Message checksum definition for a sender node.

Figure A.22: Specific settings of RTICANMM blocks

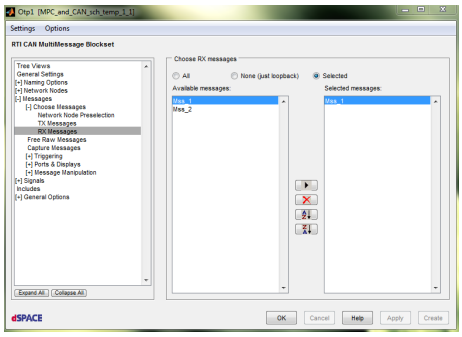


(a) RTI CAN Multi Message general settings for a receiver node (note that only difference between two receiver nodes with respect to settings is in the corresponding index, i.e., 1 or 2.).

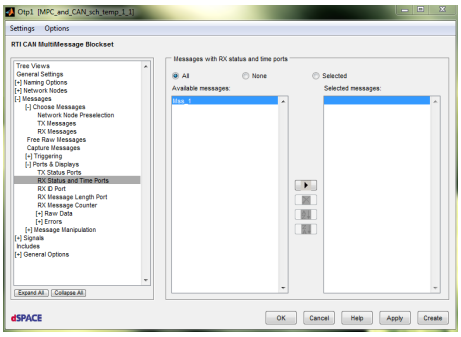


(b) RTI CAN Multi Message TX messages for a receiver node.

Figure A.23: Specific settings of RTICANMM blocks

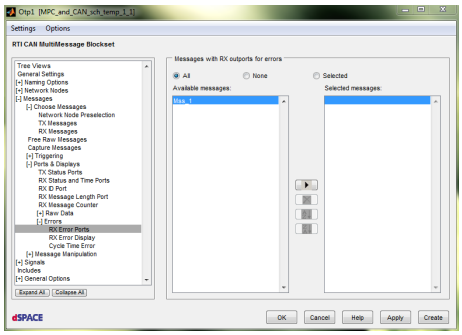


(a) RTI CAN Multi Message RX messages for a receiver node.

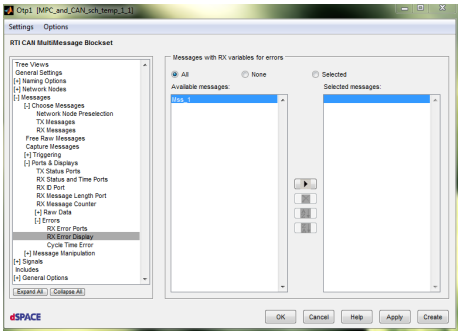


(b) RTI CAN Multi Message RX status time ports for a receiver node.

Figure A.24: Specific settings of RTICANMM blocks

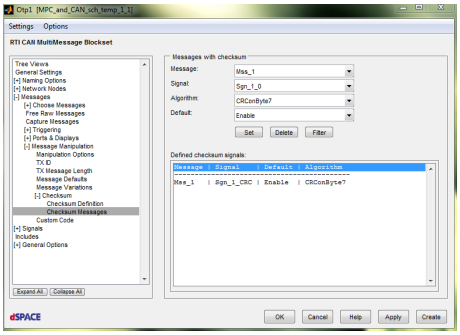


(a) RTI CAN Multi Message RX error ports for a receiver node.

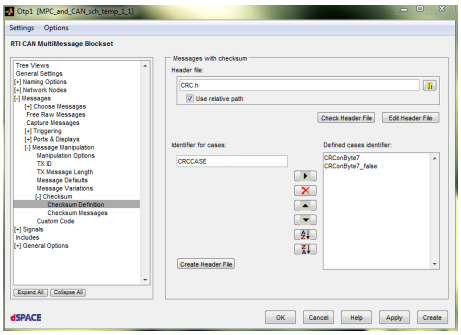


(b) RTI CAN Multi Message RX error display for a receiver node.

Figure A.25: Specific settings of RTICANMM blocks

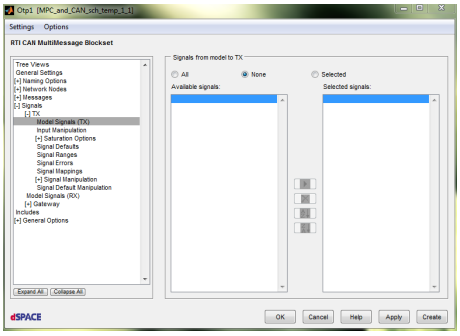


(a) RTI CAN Multi Message checksum messages for a receiver node.

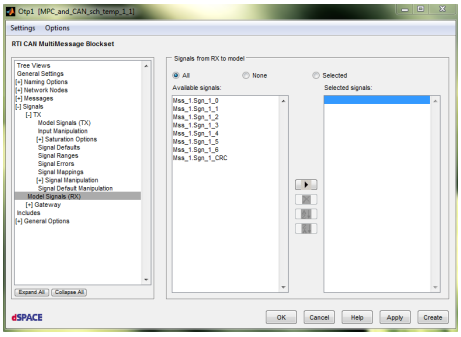


(b) RTI CAN Multi Message checksum definition for a receiver node.

Figure A.26: Specific settings of RTICANMM blocks



(a) RTI CAN Multi Message TX signals for a receiver node.



(b) RTI CAN Multi Message RX signals for a receiver node.

Figure A.27: Specific settings of RTICANMM blocks

Bibliography

[1] W. Rudin, *Principles of mathematical analysis*. McGraw-Hill New York, 1976, vol. 3.

[2] D. Nešić, A. R. Teel, and E. D. Sontag, "Formulas relating KL stability estimates of discrete-time and sampled-data nonlinear systems," *Systems and Control Letters*, vol. 38, no. 1, pp. 49–60, 1999.

[3] E. D. Sontag, *Mathematical Control Theory: Deterministic Finite Dimensional Systems.*, 2nd ed., ser. Textbooks in Applied Mathematics, Number 6. Springer, 1998.

[4] G. C. Goodwin, S. F. Graebe, and M. E. Salgado, *Control system design*. Prentice Hall New Jersey, 2001, vol. 240.

[5] K. J. Aström and R. M. Murray, *Feedback systems: an introduction for scientists and engineers*. Princeton university press, 2010.

[6] G. F. Franklin, J. D. Powell, and A. Emami-Naeini, *Feedback control of dynamic systems*, 6th ed. Pearson Education, 2009.

[7] S. P. Boyd, C. H. Barratt, S. P. Boyd, and S. P. Boyd, *Linear controller design: limits of performance*. Prentice Hall Englewood Cliffs, NJ, 1991.

[8] O. Katsuhiko, *Modern control engineering*, 4th ed. Aeeizh, 2010.

[9] Y.-B. Zhao, G.-P. Liu, and D. Rees, "Actively compensating for data packet disorder in networked control systems," *Circuits and Systems II: Express Briefs, IEEE Transactions on*, vol. 57, no. 11, pp. 913–917, Nov 2010.

[10] Y. Tipsuwan and M.-Y. Chow, "Control methodologies in networked control systems," *Control Engineering Practice*, vol. 11, no. 10, pp. 1099 – 1111, 2003, special Section on Control Methods for Telecommunication.

[11] T. Yang, "Networked control system: a brief survey," *IEE Proceedings - Control Theory and Applications*, vol. 153, pp. 403–412(9), July 2006.

[12] J. Hespanha, P. Naghshtabrizi, and Y. Xu, "A survey of recent results in networked control systems," *Proceedings of the IEEE*, vol. 95, no. 1, pp. 138–162, Jan 2007.

[13] R. Gupta and M.-Y. Chow, "Networked control system: Overview and research trends," *Industrial Electronics, IEEE Transactions on*, vol. 57, no. 7, pp. 2527–2535, July 2010.

[14] Z. Lixian, G. Huijun, and O. Kaynak, "A survey of network-induced constraints in networked control systems," *Industrial Informatics, IEEE Transactions on*, vol. 9, no. 1, pp. 403–416, Feb 2013.

[15] R. M. Murray, Ed., *Control in an Information Rich World*. Society for Industrial and Applied Mathematics, 2003. [Online]. Available: http://epubs.siam.org/doi/abs/10.1137/1.9780898718010

[16] T. Samad and A. M. E. Annaswamy, "The impact of control technology," IEEE Control Systems Society, Tech. Rep., 2011.

[17] D. R. Boggs, J. C. Mogul, and C. A. Kent, *Measured capacity of an Ethernet: Myths and reality*. ACM, 1988, vol. 18, no. 4.

[18] A. Kamerman and G. Aben, "Net throughput with IEEE 802.11 wireless LANs," in *Wireless Communications and Networking Confernce, 2000. WCNC. 2000 IEEE*, vol. 2. IEEE, 2000, pp. 747–752.

[19] F. Lian, J. Moyne, and D. Tilbury, "Performance evaluation of control networks: Ethernet, ControlNet, and DeviceNet," *Control Systems, IEEE*, vol. 21, no. 1, pp. 66–83, 2001.

[20] J. R. Moyne and D. Tilbury, "The emergence of industrial control networks for manufacturing control, diagnostics, and safety data," *Proceedings of the IEEE*, vol. 95, no. 1, pp. 29–47, Jan 2007.

[21] L. Greco, A. Chaillet, and A. Bicchi, "Exploiting packet size in uncertain nonlinear networked control systems," *Automatica*, vol. 48, no. 11, pp. 2801–2811, 2012.

[22] D. Hristu-Varsakelis and W. S. Levine, Eds., *Handbook of Networked and Embedded Control Systems*, ser. Control Engineering. Birkhäuser Boston, 2005.

[23] V. Saligrama, *Networked Sensing Information and Control*. Springer, 2007.

[24] A. Bemporad, M. Heemels, and M. Johansson, "Networked control systems," *Lecture Notes in Control and Information Sciences. Heidelberg: Springer Verlag*, vol. 406, 2010.

[25] L. Bushnell, "Networks and control [guest editorial]," *Control Systems, IEEE*, vol. 21, no. 1, pp. 22–23, Feb 2001.

[26] M.-Y. Chow, "Specail section papers on distributed networed-based control systems and applications [guest editorial]," *Industrial Electronics, IEEE Transactions on*, vol. 51, no. 6, pp. 1126–1126, Dec 2004.

[27] P. Antsaklis and J. Baillieul, "Guest editorial special issue on networked control systems," *Automatic Control, IEEE Transactions on*, vol. 49, no. 9, pp. 1421–1423, Sept 2004.

[28] ——, "Special issue on technology of networked control systems," *Proceedings of the IEEE*, vol. 95, no. 1, pp. 5–8, Jan 2007.

[29] A. Bemporad, "Predictive control of teleoperated constrained systems with unbounded communication delays," *Proceedings of the 37th IEEE Conference on Decision and Control, 1998.*, vol. 2, pp. 2133–2138, 1998.

[30] D. E. Quevedo, E. I. Silva, and D. Nešić, "Design of multiple actuator-link control systems with packet dropouts," in *Proc. of the 17th IFAC World Congress*, 2008.

[31] D. Quevedo and D. Nešić, "Input-to-state stability of packetized predictive control over unreliable networks affected by packet-dropouts," *IEEE TAC*, vol. 88, pp. 792–800, Apr 2015.

[32] M. Lješnjanin, D. E. Quevedo, and D. Nešić, "Packetized MPC with dynamic scheduling constraints and bounded packet dropouts," *Automatica*, vol. 50, no. 3, pp. 784–797, 3 2014.

[33] L. Xiao, A. Hassibi, and J. P. How, "Control with random communication delays via a discrete-time jump system approach," in *American Control Conference, 2000. Proceedings of the 2000*, vol. 3. IEEE, 2000, pp. 2199–2204.

[34] H. Lin and P. J. Antsaklis, "Stability and persistent disturbance attenuation properties for a class of networked control systems: switched system approach," *International Journal of Control*, vol. 78, no. 18, pp. 1447–1458, 2005.

[35] P. Seiler and R. Sengupta, "An H_∞ approach to networked control," *Automatic Control, IEEE Transactions on*, vol. 50, no. 3, pp. 356–364, march 2005.

[36] W. Zhang, M. Branicky, and S. Phillips, "Stability of networked control systems," *Control Systems, IEEE*, vol. 21, no. 1, pp. 84–99, Feb 2001.

[37] D. Yue, Q.-L. Han, and C. Peng, "State feedback controller design of networked control systems," in *Control Applications, 2004. Proceedings of the 2004 IEEE International Conference on*, vol. 1. IEEE, 2004, pp. 242–247.

[38] L. Zhang and D. Hristu-Varsakelis, "Communication and control co-design for networked control systems," *Automatica*, vol. 42, no. 6, pp. 953–958, 2006.

[39] H. Gao and T. Chen, "Network-based output tracking control," *Automatic Control, IEEE Transactions on*, vol. 53, no. 3, pp. 655–667, April 2008.

[40] B. Sinopoli, L. Schenato, M. Franceschetti, K. Poolla, M. Jordan, and S. Sastry, "Kalman filtering with intermittent observations," *Automatic Control, IEEE Transactions on*, vol. 49, no. 9, pp. 1453–1464, Sept 2004.

[41] Z. Wang, F. Yang, D. W. C. Ho, and X. Liu, "Robust control for networked systems with random packet losses," *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, vol. 37, no. 4, pp. 916–924, Aug 2007.

[42] N. Elia and J. Eisenbeis, "Limitations of linear control over packet drop networks," *Automatic Control, IEEE Transactions on*, vol. 56, no. 4, pp. 826–841, April 2011.

[43] T. Gommans, W. Heemels, N. W. Bauer, and N. Wouw, "Compensation-based control for lossy communication networks," *International Journal of Control*, vol. 86, no. 10, pp. 1880–1897, 2013.

[44] G. Walsh, O. Beldiman, and L. Bushnell, "Asymptotic behavior of nonlinear networked control systems," *Automatic Control, IEEE Transactions on*, vol. 46, no. 7, pp. 1093–1097, jul 2001.

[45] G. Walsh and H. Ye, "Scheduling of networked control systems," *Control Systems Magazine, IEEE*, vol. 21, no. 1, pp. 57–65, 2001.

[46] G. Walsh, H. Ye, and L. Bushnell, "Stability analysis of networked control systems," *Control Systems Technology, IEEE Transactions on*, vol. 10, no. 3, pp. 438–446, may 2002.

[47] D. Nešić and A. Teel, "Input-output stability properties of networked control systems," *IEEE TAC*, vol. 49, no. 10, pp. 1650–1667, Oct 2004.

[48] D. Nešić and A. R. Teel, "Input-to-state stability of networked control systems," *Automatica*, vol. 40, no. 12, pp. 2121 – 2128, 2004.

[49] D. Carnevale, A. R. Teel, and D. Nešić, "A Lyapunov Proof of an Improved Maximum Allowable Transfer Interval for Networked Control Systems," *Automatic Control, IEEE Transactions on*, vol. 52, no. 5, pp. 892–897, may 2007.

[50] M. Tabbara, D. Nešić, and A. Teel, "Stability of wireless and wireline networked control systems," *Automatic Control, IEEE Transactions on*, vol. 52, no. 9, pp. 1615–1630, Sept 2007.

[51] M. Tabbara and D. Nešić, "Input-output stability of networked control systems with stochastic protocols and channels," *Automatic Control, IEEE Transactions on*, vol. 53, no. 5, pp. 1160–1175, June 2008.

[52] T. Suzuki, M. Kono, N. Takahashi, and O. Sato, "Controllability and stabilizability of a networked control system with periodic communication constraints," *Systems & Control Letters*, vol. 60, no. 12, pp. 977 – 984, 2011.

[53] F. Smarra, A. D'Innocenzo, and M. D. D. Benedetto, "Fault tolerant stabilizability of mimo multi-hop cotrol networks," in *3rd IFAC Workshop on Distributed Estimiation and Control in Networked Systems*, September 2012.

[54] A. D'Innocenzo, M. Di Benedetto, and E. Serra, "Fault tolerant control of multi-hop control networks," *IEEE TAC*, vol. 58, no. 6, pp. 1377–1389, June 2013.

[55] X. Yu and S. B. Andersson, "Effect of switching delay on a network control system," in *52nd IEEE Conference on Decision and Control*, 2013.

[56] M. Lješnjanin, D. E. Quevedo, and D. Nešić, "Controllability of discrete-time networked control systems with try once discard protocol," in *19th IFAC World Congress*, Cape Town, South Africa, 2014.

[57] D. Yue, Q.-L. Han, and J. Lam, "Network-based robust control of systems with uncertainty," *Automatica*, vol. 41, no. 6, pp. 999–1007, 6 2005.

[58] M. Cloosterman, N. van de Wouw, W. Heemels, and H. Nijmeijer, "Stability of Networked Control Systems With Uncertain Time-Varying Delays," *Automatic Control, IEEE Transactions on*, vol. 54, no. 7, pp. 1575–1580, july 2009.

[59] Y. Zheng, H. Fang, and H. Wang, "Takagi-sugeno fuzzy-model-based fault detection for networked control systems with markov delays," *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, vol. 36, no. 4, pp. 924–929, Aug 2006.

[60] N. B. Almutairi, M.-Y. Chow, and Y. Tipsuwan, "Network-based controlled dc motor with fuzzy compensation," in *Industrial Electronics Society, 2001. IECON'01. The 27th Annual Conference of the IEEE*, vol. 3. IEEE, 2001, pp. 1844–1849.

[61] G. Pin and T. Parisini, "Networked predictive control of uncertain constrained nonlinear systems: Recursive feasibility and input-to-state stability analysis," *IEEE TAC*, vol. 56, no. 1, pp. 72–87, Jan 2011.

[62] L. Zhang, Y. Shi, T. Chen, and B. Huang, "A new method for stabilization of networked control systems with random delays," *Automatic Control, IEEE Transactions on*, vol. 50, no. 8, pp. 1177–1181, Aug 2005.

[63] A. Casavola, E. Mosca, and M. Papini, "Predictive teleoperation of constrained dynamic systems via Internet-like channels," *Control Systems Technology, IEEE Transactions on*, vol. 14, no. 4, pp. 681–694, july 2006.

[64] P. L. Tang and C. W. de Silva, "Compensation for transmission delays in an ethernet-based control network using variable-horizon predictive control," *Control Systems Technology, IEEE Transactions on*, vol. 14, no. 4, pp. 707–718, 2006.

[65] Y.-B. Zhao, G. Liu, and D. Rees, "Improved predictive control approach to networked control systems," *Control Theory Applications, IET*, vol. 2, no. 8, pp. 675–681, Aug 2008.

[66] H. Lin and P. J. Antsaklis, "Robust regulation of polytopic uncertain linear hybrid systems with networked control system applications," in *Stability and control of dynamical systems with applications*. Springer, 2003, pp. 71–96.

[67] D. Delchamps, "Stabilizing a linear system with quantized state feedback," *Automatic Control, IEEE Transactions on*, vol. 35, no. 8, pp. 916–924, 1990.

[68] W. S. Wong and R. Brockett, "Systems with finite communication bandwidth constraints. ii. stabilization with limited information feedback," *Automatic Control, IEEE Transactions on*, vol. 44, no. 5, pp. 1049–1053, May 1999.

[69] N. Elia and S. Mitter, "Stabilization of linear systems with limited information," *Automatic Control, IEEE Transactions on*, vol. 46, no. 9, pp. 1384–1400, Sep 2001.

[70] M. Fu and L. Xie, "The sector bound approach to quantized feedback control," *Automatic Control, IEEE Transactions on*, vol. 50, no. 11, pp. 1698–1711, Nov 2005.

[71] R. Brockett and D. Liberzon, "Quantized feedback stabilization of linear systems," *Automatic Control, IEEE Transactions on*, vol. 45, no. 7, pp. 1279–1289, 2000.

[72] D. Liberzon, "On stabilization of linear systems with limited information," *Automatic Control, IEEE Transactions on*, vol. 48, no. 2, pp. 304–307, feb. 2003.

[73] ——, "Hybrid feedback stabilization of systems with quantized signals," *Automatica*, vol. 39, no. 9, pp. 1543–1554, 9 2003.

[74] D. Liberzon and D. Nesic, "Input-to-State Stabilization of Linear Systems With Quantized State Measurements," *Automatic Control, IEEE Transactions on*, vol. 52, no. 5, pp. 767–781, may 2007.

[75] D. Liberzon and J. P. Hespanha, "Stabilization of nonlinear systems with limited information feedback," *Automatic Control, IEEE Transactions on*, vol. 50, no. 6, pp. 910–915, 2005.

[76] D. Nešić and D. Liberzon, "A Unified Framework for Design and Analysis of Networked and Quantized Control Systems," *Automatic Control, IEEE Transactions on*, vol. 54, no. 4, pp. 732–747, april 2009.

[77] B. Wittenmark, J. Nilsson, and M. Törngren, "Timing problems in real-time control systems," in *In Proceedings of the American Control Conference*. Citeseer, 1995.

[78] B. Hu and A. N. Michel, "Stability analysis of digital feedback control systems with time-varying sampling periods," *Automatica*, vol. 36, no. 6, pp. 897–905, 6 2000.

[79] E. Fridman, A. Seuret, and J.-P. Richard, "Robust sampled-data stabilization of linear systems: an input delay approach," *Automatica*, vol. 40, no. 8, pp. 1441–1446, 2004.

[80] Y. S. Suh, "Stability and stabilization of nonuniform sampling systems," *Automatica*, vol. 44, no. 12, pp. 3222–3226, 2008.

[81] W. Zhang and M. S. Branicky, "Stability of networked control systems with time-varying transmission period," in *Proceedings Of The Annual Allerton Conference On Communication Control And Computing*, vol. 39, no. 2, 2001, pp. 1205–1214.

[82] L. A. Montestruque and P. J. Antsaklis, "On the model-based control of networked systems," *Automatica*, vol. 39, no. 10, pp. 1837–1843, 10 2003.

[83] L. Montestruque and P. Antsaklis, "Stability of model-based networked control systems with time-varying transmission times," *Automatic Control, IEEE Transactions on*, vol. 49, no. 9, pp. 1562–1572, Sept 2004.

[84] M. Garcia-Rivera and A. Barreiro, "Analysis of networked control systems with drops and variable delays," *Automatica*, vol. 43, no. 12, pp. 2054–2059, 12 2007.

[85] H. Li, M.-Y. Chow, and Z. Sun, "Optimal stabilizing gain selection for networked control systems with time delays and packet losses," *Control Systems Technology, IEEE Transactions on*, vol. 17, no. 5, pp. 1154–1162, Sept 2009.

[86] L. Schenato, "Optimal estimation in networked control systems subject to random delay and packet drop," *Automatic Control, IEEE Transactions on*, vol. 53, no. 5, pp. 1311–1317, June 2008.

[87] Y.-C. Tian and D. Levy, "Compensation for control packet dropout in networked control systems," *Information Sciences*, vol. 178, no. 5, pp. 1263–1278, 3 2008.

[88] J. Wang, W. X. Zheng, and T. Chen, "Identification of linear dynamic systems operating in a networked environment," *Automatica*, vol. 45, no. 12, pp. 2763–2772, 12 2009.

[89] J. Xiong and J. Lam, "Stabilization of networked control systems with a logic zoh," *Automatic Control, IEEE Transactions on*, vol. 54, no. 2, pp. 358–363, Feb 2009.

[90] L. Hetel, J. Daafouz, and C. Iung, "Analysis and control of lti and switched systems in digital loops via an event-based modelling," *International Journal of Control*, vol. 81, no. 7, pp. 1125–1138, 2014/06/18 2008.

[91] A. Sala, "Computer control under time-varying sampling period: An lmi gridding approach," *Automatica*, vol. 41, no. 12, pp. 2077–2082, 12 2005.

[92] N. van de Wouw, P. Naghshtabrizi, M. B. G. Cloosterman, and J. P. Hespanha, "Tracking control for sampled-data systems with uncertain time-varying sampling intervals and delays," *International Journal of Robust and Nonlinear Control*, vol. 20, no. 4, pp. 387–411, 2010.

[93] M. B. G. Cloosterman, L. Hetel, N. van de Wouw, W. P. M. H. Heemels, J. Daafouz, and H. Nijmeijer, "Controller synthesis for networked control systems," *Automatica*, vol. 46, no. 10, pp. 1584–1594, 10 2010.

[94] I. G. Polushin, P. X. Liu, and C.-H. Lung, "On the model-based approach to nonlinear networked control systems," *Automatica*, vol. 44, no. 9, pp. 2409–2414, 9 2008.

[95] D. B. Dačić and D. Nešić, "Quadratic stabilization of linear networked control systems via simultaneous protocol and controller design," *Automatica*, vol. 43, no. 7, pp. 1145–1155, 2007.

[96] D.-S. Kim, D.-H. Choi, and P. Mohapatra, "Real-time scheduling method for networked discrete control systems," *Control Engineering Practice*, vol. 17, no. 5, pp. 564–570, 5 2009.

[97] K.-C. Lee, S. Lee, and M. H. Lee, "Qos-based remote control of networked control systems via profibus token passing protocol," *Industrial Informatics, IEEE Transactions on*, vol. 1, no. 3, pp. 183–191, Aug 2005.

[98] W. Heemels, A. Teel, N. van de Wouw, and D. Neš andić and, "Networked Control Systems With Communication Constraints: Tradeoffs Between Transmission Intervals, Delays and Performance," *Automatic Control, IEEE Transactions on*, vol. 55, no. 8, pp. 1781–1796, aug. 2010.

[99] M. Donkers, W. Heemels, N. van de Wouw, and L. Hetel, "Stability Analysis of Networked Control Systems Using a Switched Linear Systems Approach," *Automatic Control, IEEE Transactions on*, vol. 56, no. 9, pp. 2101–2115, sept. 2011.

[100] D. Liberzon, "Quantization, time delays, and nonlinear stabilization," *Automatic Control, IEEE Transactions on*, vol. 51, no. 7, pp. 1190–1195, july 2006.

[101] E. Fridman and M. Dambrine, "Control under quantization, saturation and delay: An lmi approach," *Automatica*, vol. 45, no. 10, pp. 2258–2264, 10 2009.

[102] K. Tsumura, H. Ishii, and H. Hoshina, "Tradeoffs between quantization and packet loss in networked control of linear systems," *Automatica*, vol. 45, no. 12, pp. 2963 – 2970, 2009.

[103] Y. Ishido, K. Takaba, and D. E. Quevedo, "Stability analysis of networked control systems subject to packet-dropouts and finite-level quantization," *Systems & Control Letters*, vol. 60, no. 5, pp. 325–332, 2011.

[104] R. R. Bitmead, M. Gevers, and V. Wertz, *Adaptive Optimal Control: The Thinking Man's GPC*. Prentice Hall, 1990.

[105] E. F. Camacho and C. A. Bordons, *Model predictive control in the process industry*. Springer-Verlag New York, Inc., 1997.

[106] J. M. Maciejowski, *Predictive control: with constraints*. Pearson education, 2002.

[107] J. B. Rawlings and D. Q. Mayne, *Model Predictive Control: Theory and Design*. Madison, WI: Nob Hill Publishing, LCC, 2009.

[108] L. Grüne and J.Pannek, *Nonlinear Model Predictive Control: Theory and Algorithms*, 1st ed. Springer-Verlag, 2011.

[109] E. F. Camacho and C. B. Alba, *Model predictive control*. Springer, 2013.

[110] J. Richalet, A. Rault, J. L. Testud, and J. Papon, "Model predictive heuristic control: Applications to industrial processes," *Automatica*, vol. 14, no. 5, pp. 413–428, 1978.

[111] C. E. García, D. M. Prett, and M. Morari, "Model predictive control: Theory and practice - A survey," *Automatica*, vol. 25, no. 3, pp. 335–348, 1989.

[112] J. B. Rawlings, E. S. Meadows, and K. R. Muske, "Nonlinear model predictive control: A tutorial and survey." Preprints IFAC Symposium ADCHEM, 1994, pp. 185–197.

[113] D. Q. Mayne, "Optimization in model based control." Proceedings of the IFAC Symposium on Dynamics and Control of Chemical Reactors and Batch Processes, 1995, pp. 229–242.

[114] ——, "Nonlinear model predictive control: An assessment." AMERICAN INSTITUTE OF CHEMICAL ENGINEERS, 1997, pp. 217–231.

[115] J. H. Lee and B. Cooley, "Recent advances in model predictive control and other related areas," vol. 93, no. 316. American Institute of Chemical Engineers, 1997, pp. 201–216.

[116] S. J. Qin and T. A. Badgwell, "An overview of industrial model predictive control technology," in *AIChE Symposium Series*, vol. 93, no. 316. New York, NY: American Institute of Chemical Engineers, 1971-c2002., 1997, pp. 232–256.

[117] H. Chen and F. Allgöwer, "Nonlinear model predictive control schemes with guaranteed stability," *Nonlinear Model Based Process Control*, pp. 465–494, 1998.

[118] M. Morari and J. H. Lee, "Model predictive control: past, present and future," *Computers & Chemical Engineering*, vol. 23, pp. 667–682, 5 1999. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0098135498003019

[119] D. Q. Mayne, J. B. Rawlings, C. V. Rao, and P. O. M. Scokaert, "Constrained model predictive control: Stability and optimality," *Automatica*, vol. 36, no. 6, pp. 789–814, 6 2000.

[120] S. J. Qin and T. A. Badgwell, "A survey of industrial model predictive control technology," *Control Engineering Practice*, vol. 11, no. 7, pp. 733–764,

7 2003. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0967066102001867

[121] R. Scattolini, "Architectures for distributed and hierarchical Model Predictive Control – A review," *Journal of Process Control*, vol. 19, no. 5, pp. 723–731, 2009.

[122] G. Grimm, M. J. Messina, S. E. Tuna, and A. R. Teel, "Examples when nonlinear model predictive control is nonrobust," *Automatica*, vol. 40, no. 10, pp. 1729–1738, 2004.

[123] G. Grimm, M. Messina, S. Tuna, and A. Teel, "Model predictive control: for want of a local control Lyapunov function, all is not lost," *Automatic Control, IEEE Transactions on*, vol. 50, no. 5, pp. 546–558, may 2005.

[124] S. S. Keerthi and E. G. Gilbert, "Optimal infinite-horizon feedback laws for a general class of constrained discrete-time systems: Stability and moving-horizon approximations," *Jurnal of Optimization Theory and Applications*, vol. 57, no. 2, pp. 265–293, may 1988.

[125] D. Q. Mayne and H. Michalska, "Receding horizon control of nonlinear systems," *Automatic Control, IEEE Transactions on*, vol. 35, no. 7, pp. 814–824, 1990.

[126] J. B. Rawlings and K. R. Muske, "The stability of constrained receding horizon control," *Automatic Control, IEEE Transactions on*, vol. 38, no. 10, pp. 1512–1516, 1993.

[127] H. Michalska and D. Mayne, "Robust receding horizon control of constrained nonlinear systems," *Automatic Control, IEEE Transactions on*, vol. 38, no. 11, pp. 1623–1633, nov 1993.

[128] D. Chmielewski and V. Manousiouthakis, "On constrained infinite-time linear quadratic optimal control," in *Decision and Control, 1996., Proceedings of the 35th IEEE Conference on*, vol. 2. IEEE, 1996, pp. 1319–1324.

[129] P. O. Scokaert and J. B. Rawlings, "Constrained linear quadratic regulation," *Automatic Control, IEEE Transactions on*, vol. 43, no. 8, pp. 1163–1169, 1998.

[130] C. Chen and L. Shaw, "On receding horizon feedback control," *Automatica*, vol. 18, no. 3, pp. 349–352, 1982.

[131] A. Bemporad, L. Chisci, and E. Mosca, "On the stabilizing property of siorhc," *Automatica*, vol. 30, no. 12, pp. 2013–2015, 1994.

[132] G. de Nicolao, L. Magni, and R. Scattolini, "On the robustness of receding-horizon control with terminal constraints," *Automatic Control, IEEE Transactions on*, vol. 41, no. 3, pp. 451–453, mar 1996.

[133] L. Magni and R. Sepulchre, "Stability margins of nonlinear receding-horizon control via inverse optimality," *Systems & Control Letters*, vol. 32, no. 4, pp. 241–245, 1997.

[134] J. A. Primbs and V. Nevistic, "Constrained finite receding horizon linear quadratic control," in *Decision and Control, 1997., Proceedings of the 36th IEEE Conference on*, vol. 4. IEEE, 1997, pp. 3196–3201.

[135] A. Jadbabaie, J. Primbs, and J. Hauser, "Unconstrained receding horizon control with no terminal cost," in *American Control Conference, 2001. Proceedings of the 2001*, vol. 4, 2001, pp. 3055–3060 vol.4.

[136] F. A. Fontes, "A general framework to design stabilizing nonlinear model predictive controllers," *Systems & Control Letters*, vol. 42, no. 2, pp. 127–143, 2001.

[137] D. Limón Marruedo, T. Alamo, and E. Camacho, "Stability analysis of systems with bounded additive uncertainties based on invariant sets: Stability and feasibility of mpc," in *American Control Conference, 2002. Proceedings of the 2002*, vol. 1. IEEE, 2002, pp. 364–369.

[138] H. Chen, C. Scherer, and F. Allgower, "A game theoretic approach to nonlinear robust receding horizon control of constrained systems," in *American Control Conference, 1997. Proceedings of the 1997*, vol. 5, jun 1997, pp. 3073–3077 vol.5.

[139] L. Magni, H. Nijmeijer, and A. Van Der Schaft, "A receding horizon approach to the nonlinear $\mathcal{H}_\infty$ problem," *Automatica*, vol. 37, no. 3, pp. 429–435, 2001.

[140] L. Magni, G. De Nicolao, R. Scattolini, and F. Allgöwer, "Robust model predictive control for nonlinear discrete-time systems," *International Journal of Robust and Nonlinear Control*, vol. 13, no. 3-4, pp. 229–246, 2003.

[141] M. V. Kothare, V. Balakrishnan, and M. Morari, "Robust constrained model predictive control using linear matrix inequalities," *Automatica*, vol. 32, no. 10, pp. 1361–1379, 10 1996. [Online]. Available: http://www.sciencedirect.com/science/article/pii/0005109896000635

[142] J. a. Lee and Z. Yu, "Worst-case formulations of model predictive control for systems with bounded parameters," *Automatica*, vol. 33, no. 5, pp. 763–781, 1997.

[143] P. Scokaert and D. Mayne, "Min-max feedback model predictive control for constrained linear systems," *Automatic Control, IEEE Transactions on*, vol. 43, no. 8, pp. 1136–1142, aug 1998.

[144] G. De Nicolao, L. Magni, and R. Scattolini, "Robustness of receding horizon control for nonlinear discrete-time systems," in *Robustness in identification and control*. Springer, 1999, pp. 408–421.

[145] W. Langson, I. Chryssochoos, S. Raković, and D. Q. Mayne, "Robust model predictive control using tubes," *Automatica*, vol. 40, no. 1, pp. 125–133, 2004.

[146] D. Q. Mayne, M. M. Seron, and S. V. Rakovic, "Robust model predictive control of constrained linear systems with bounded disturbances." *Automatica*, vol. 41, no. 2, pp. 219–224, 2005.

[147] J. B. Rawlings, D. Bonné, J. B. Jorgensen, A. N. Venkat, and S. B. Jorgensen, "Unreachable setpoints in model predictive control," *Automatic Control, IEEE Transactions on*, vol. 53, no. 9, pp. 2209–2215, 2008.

[148] J. B. Rawlings and R. Amrit, "Optimizing process economic performance using model predictive control," in *Nonlinear Model Predictive Control*, L. Magni, D. M. Raimondo, and F. Allgöwer, Eds. Springer, 2009, pp. 119–138.

[149] D. Angeli, R. Amrit, and J. Rawlings, "On average performance and stability of economic model predictive control," *Automatic Control, IEEE Transactions on*, vol. 57, no. 7, pp. 1615–1626, July 2012.

[150] D. Angeli, R. Amrit, and J. B. Rawlings, "Receding horizon cost optimization for overly constrained nonlinear plants," in *proceeding of CDC/CCC 2009.* IEEE, 2009, pp. 7972–7977.

[151] M. A. Müller, D. Angeli, and F. Allgöwer, "On convergence of averagely constrained economic MPC and necessity of dissipativity for optimal steady-state operation," in *American Control Conference (ACC), 2013.* IEEE, 2013, pp. 3141–3146.

[152] D. Angeli and J. B. Rawlings, "Receding horizon cost optimization and control for nonlinear plants," in *8th IFAC Symposium on Nonlinear Control Systems*, 2010, pp. 1217–1223.

[153] R. Amrit, J. B. Rawlings, and D. Angeli, "Economic optimization using model predictive control with a terminal cost," *Annual Reviews in Control*, vol. 35, no. 2, pp. 178–186, 2011.

[154] D. Angeli, R. Amrit, and J. B. Rawlings, "Enforcing Convergence in Nonlinear Economic MPC," in *proceedings CDC-ECC, 2003.*, Dec. 2011.

[155] M. Diehl, R. Amrit, and J. Rawlings, "A Lyapunov Function for Economic Optimizing Model Predictive Control," *Automatic Control, IEEE Transactions on*, vol. 56, no. 3, pp. 703–707, march 2011.

[156] L. Grüne, "Economic receding horizon control without terminal constraints," *Automatica*, vol. 49, no. 3, pp. 725–734, 2013.

[157] D. E. Quevedo, E. I. Silva, and G. C. Goodwin, "Packetized predictive control over erasure channels," in *American Control Conference, 2007. ACC'07*, 2007, pp. 1003–1008.

[158] T. Kameneva and D. Nešić, "On l_2 Stabilization of Linear Systems With Quantized Control," *Automatic Control, IEEE Transactions on*, vol. 53, no. 1, pp. 399–405, Feb. 2008.

[159] M. Lješnjanin, D. Nešić, and D. E. Quevedo, "Uniform global asynptotic stability of networked contorl systems with bounded packet dropouts and scheduling constraints," *Automatica (submitted)*, 2014.

[160] G. Kreisselmeier and T. Birkholzer, "Numerical nonlinear regulator design," *Automatic Control, IEEE Transactions on*, vol. 39, no. 1, pp. 33–46, jan 1994.

[161] A. R. Teel and L. Praly, "A smooth Lyapunov function from a class-$\mathcal{KL}$ estimate involving two positive semidefinite functions," *ESAIM: Control, Optimisation and Calculus of Variations*, vol. 5, pp. 313–367, 1 2000.

[162] C. Kellett and A. Teel, "On the robustness of $\mathcal{KL}$-stability for difference inclusions: Smooth discrete-time Lyapunov functions," *SIAM Journal on Control and Optimization*, vol. 44, no. 3, pp. 777–800, 2005.

[163] A. Hekler, J. Fischer, and U. D. Hanebeck, "Sequence-based control for networked control systems based on virtual control inputs," in *Decision and Control (CDC), 2012 IEEE 51st Annual Conference on*. IEEE, 2012, pp. 7–13.

[164] D. Quevedo, E. Silva, and G. Goodwin, "Control over unreliable networks affected by packet erasures and variable transmission delays," *Selected Areas in Communications, IEEE Journal on*, vol. 26, no. 4, pp. 672–685, 2008.

[165] L. Grüne, J. Pannek, and K. Worthmann, "A networked unconstrained nonlinear MPC scheme," in *Proceedings of the European Control Conference*, 2009.

[166] M. Huang and S. Dey, "Stability of Kalman filtering with Markovian packet losses," *Automatica*, vol. 43, no. 4, pp. 598–607, 2007.

[167] D. E. Quevedo, A. Ahlén, and J. Østergaard, "Energy efficient state estimation with wireless sensors through the use of predictive power control and coding," *Signal Processing, IEEE Transactions on*, vol. 58, no. 9, pp. 4811–4823, 2010.

[168] D. E. Quevedo, A. Ahlén, A. S. Leong, and S. Dey, "On Kalman filtering over fading wireless channels with controlled transmission powers," *Automatica*, vol. 48, no. 7, pp. 1306–1316, 2012.

[169] H. K. Khalil, *Nonlinear Systems*. Prentice Hall, 2002.

[170] L. Magni, D. M. Raimondo, and R. Scattolini, "Regional input-to-state stability for nonlinear model predictive control," *Automatic Control, IEEE Transactions on*, vol. 51, no. 9, pp. 1548–1553, 2006.

[171] B. Ingalls, E. Sontag, and Y. Wang, "Generalizations of asymptotic gain characterizations of ISS to input-to-output stability," in *American Control Conference, 2001. Proceedings of the 2001*, vol. 3, 2001, pp. 2279–2284 vol.3.

[172] F.-L. Lian, J. Moyne, and D. Tilbury, "Network protocols for networked control systems," in *Handbook of Networked and Embedded Control Systems*, ser. Control Engineering. Birkhäuser Boston, 2005, pp. 651–675.

[173] R. Findeisen and P. Varutti, "Stabilizing nonlinear predictive control over nondeterministic communication networks," in *Nonlinear Model Predictive Control*, ser. Lecture Notes in Control and Information Sciences. Springer Berlin Heidelberg, 2009, vol. 384, pp. 167–179.

[174] D. Munoz de la Pena and P. Christofides, "Lyapunov-based model predictive control of nonlinear systems subject to data losses," *IEEE TAC*, vol. 53, no. 9, pp. 2076–2089, Oct 2008.

[175] H. Riesel, *Prime numbers and computer methods for factorization*. Springer, 1994, vol. 126.

[176] R. Isermann, J. Schaffnit, and S. Sinsel, "Hardware-in-the-loop simulation for the design and testing of engine-control systems," *Control Engineering Practice*, vol. 7, no. 5, pp. 643–653, 1999.

[177] D. Paret, *Multiplexed networks for embedded systems: CAN, LIN, Flexray, Safe-by-Wire...* John Wiley & Sons, 2007.

[178] M. Di Natale, H. Zeng, P. Giusto, and A. Ghosal, *Understanding and using the controller area network communication protocol: theory and practice*. Springer, 2012.

[179] dSPACE GmbH, *dSPACE Hardware Installationand and Configuration Reference*, Release 7.4, vol. Release 7.4.

[180] ——, *RTI CAN MultiMessage Blockset - Tutorial*, 2013, vol. Release 7.4.