Helsinki University of Technology Control Engineering

Espoo 2008 Report 158

PID CONTROLLER DESIGN AND TUNING IN NETWORKED CONTROL SYSTEMS

Lasse Eriksson

TEKNILLINEN KORKEAKOULU
TEKNISKA HÖGSKOLAN
HELSINKI UNIVERSITY OF TECHNOLOGY
TECHNISCHE UNIVERSITÄT HELSINKI
UNIVERSITE DE TECHNOLOGIE D´HELSINKI

Helsinki University of Technology Control Engineering

Espoo October 2008 Report 158

PID CONTROLLER DESIGN AND TUNING IN NETWORKED CONTROL SYSTEMS

Lasse Eriksson

Dissertation for the degree of Doctor of Science in Technology to be presented with due permission of the Faculty of Electronics, Communications and Automation, for public examination and debate in Auditorium AS1 at Helsinki University of Technology (Espoo, Finland) on the 28th of November, 2008, at 12 noon.

Helsinki University of Technology

Faculty of Electronics, Communications and Automation

Department of Automation and Systems Technology

Distribution:
Helsinki University of Technology
Department of Automation and Systems Technology
P.O. Box 5500
FI-02015 TKK, Finland
Tel. +358-9-451 5201
Fax. +358-9-451 5208
E-mail: control.engineering@tkk.fi
http://autsys.tkk.fi/

ISBN 978-951-22-9633-0 (printed)
ISBN 978-951-22-9634-7 (pdf)
ISSN 0356-0872

Yliopistopaino
Helsinki 2008

Available on net at http://lib.tkk.fi/Diss/2008/isbn9789512296347

ABSTRACT OF DOCTORAL DISSERTATION	HELSINKI UNIVERSITY OF TECHNOLOGY P.O. BOX 1000, FI-02015 TKK http://www.tkk.fi

Author Lasse Eriksson

Name of the dissertation

PID Controller Design and Tuning in Networked Control Systems

Manuscript submitted	June 11, 2008	Manuscript revised	September 29, 2008
Date of the defence	November 28, 2008		
☐ Monograph		☒ Article dissertation (summary + original articles)	
Faculty	Faculty of Electronics, Communications and Automation		
Department	Department of Automation and Systems Technology		
Field of research	Control Engineering		
Opponents	Prof. Karl-Erik Årzén, Prof. Hannu Koivisto		
Supervisor	Prof. Heikki Koivo		

Abstract

Networked control systems (NCS) are distributed real-time computing and control systems with sensors, actuators and controllers that communicate over a shared medium. The distributed nature of NCS and issues related to the shared communication medium pose significant challenges for control design, as the control system no longer follows the rules of classical control theory. The main problems that are not well covered by the traditional control theory are varying time-delays due to communication and computation, and packet losses. During recent years, the control design of NCS and varying time-delay systems has been extensively researched. This investment has provided us with new results on stability. Often the proposed methods and solutions are far too complex for industrial use, especially if wireless automation applications are considered. The algorithms are computationally heavy, possibly requiring complete information from say, a network of hundreds or thousands of nodes. In the wireless case this is not feasible.

The above justifies the use and research of simple controller structures and algorithms for NCS. Despite the growing interest towards more advanced control algorithms, the Proportional-Integral-Derivative (PID) controller still has a dominant status in the industry. Nevertheless, using PID for NCS has not been thoroughly investigated, especially with regard to controller tuning. This thesis proposes several PID tuning methods, which provide robustness against the challenges of NCS, namely varying time-delays (jitter) and packet loss.

The doctoral thesis consists of a summary and eight publications that focus on the PID controller design, tuning and experimentation in NCS. The thesis includes a literature review of recent stability and control design results in NCS, a summary of publications and the original publications. The control design methods applied in the publications are also reviewed. In the thesis, several new methods for PID tuning in NCS are proposed. To make the methods usable, a PID tuning tool that implements one of the tuning methods is also developed. In order to verify the results of control design with real processes, the thesis suggests using the MoCoNet platform developed at the Helsinki University of Technology, Finland. The platform provides the tools for remote laboratory experiments in NCS settings. The results of the thesis indicate that the PID controller is well suited for NCS provided that the properties of the integrated communication and control system are taken into account in the tuning phase.

Keywords	Networked control systems, PID controller, tuning, varying time-delay, wireless automation		
ISBN (printed)	978-951-22-9633-0	ISSN (printed)	0356-0872
ISBN (pdf)	978-951-22-9634-7	ISSN (pdf)	
Language	English	Number of pages	98 + 62
Publisher	Helsinki University of Technology, Department of Automation and Systems Technology		
Print distribution	Helsinki University of Technology, Department of Automation and Systems Technology		
☒ The dissertation can be read at http://lib.tkk.fi/Diss/2008/isbn9789512296347			

VÄITÖSKIRJAN TIIVISTELMÄ	TEKNILLINEN KORKEAKOULU PL 1000, 02015 TKK http://www.tkk.fi

Tekijä Lasse Eriksson

Väitöskirjan nimi

PID-säätimen suunnittelu ja viritys verkottuneissa säätöjärjestelmissä

Käsikirjoituksen päivämäärä 11.6.2008	Korjatun käsikirjoituksen päivämäärä 29.9.2008
Väitöstilaisuuden ajankohta 28.11.2008	
☐ Monografia	☒ Yhdistelmäväitöskirja (yhteenveto + erillisartikkelit)

Tiedekunta	Elektroniikan, tietoliikenteen ja automaation tiedekunta
Laitos	Automaatio- ja systeemitekniikan laitos
Tutkimusala	Systeemitekniikka
Vastaväittäjät	Prof. Karl-Erik Årzén, Prof. Hannu Koivisto
Työn valvoja	Prof. Heikki Koivo

Tiivistelmä

Verkottuneet säätöjärjestelmät ovat hajautettuja reaaliaikaisia laskenta- ja säätöjärjestelmiä, joissa antureiden, säätimien ja toimilaitteiden välinen tiedonsiirto tapahtuu digitaalisesti verkon välityksellä. Tästä aiheutuu säätösuunnittelulle huomattavia haasteita, koska säätöjärjestelmä ei enää käyttäydy kuten perinteisessä säätöteoriassa oletetaan. Keskeisimmät verkottuneiden säätöjärjestelmien ongelmat, joita klassinen säätöteoria ei täysin kata, ovat tietoliikenteestä ja hajautetusta laskennasta johtuvat muuttuvat viiveet ja tietoliikennepakettien häviöt. Viime vuosina verkottuneiden ja yleisesti muuttuvaviiveisten järjestelmien säätöä on tutkittu paljon ja näille on esitetty kirjallisuudessa mm. uusia stabiilisuustuloksia. Kuitenkin usein esitetyt menetelmät ovat liian monimutkaisia sovellettaviksi teollisuuden tarpeisiin, etenkin langattoman automaation sovelluksissa. Monet algoritmit ovat laskennallisesti raskaita, ja laskennan edellytyksenä voi olla kaiken verkon keräämän tiedon saaminen käyttöön sadoilta tai tuhansilta antureilta. Etenkään langattomaan ympäristöön tällainen ei sovellu.

Edellä mainituista syistä johtuen yksinkertaisten säätörakenteiden ja –algoritmien käyttö ja tutkimus on perusteltua verkottuneissa säätöjärjestelmissä. Vaikka kehittyneet säätöalgoritmit ovat kasvattaneet suosiotaan, PID-säätimen (Proportional-Integral-Derivative) asema teollisuudessa on edelleen dominoiva. Kuitenkaan PID:n käyttöä verkottuneissa säätöjärjestelmissä ei ole perusteellisesti tutkittu etenkään säätimen virittämisen osalta. Tässä väitöskirjassa esitetään useita uusia PID-säätimen viritysmenetelmiä, joilla saavutetaan robustisuutta verkottuneiden säätöjärjestelmien erityishäiriöitä, muuttuvaa viivettä ja pakettihäviöitä, kohtaan.

Tämä väitöskirja koostuu tiivistelmästä ja kahdeksasta julkaisusta, jotka käsittelevät PID-säätimen suunnittelua, viritystä ja testausta verkottuneissa säätöjärjestelmissä. Väitöskirja sisältää kirjallisuuskatsauksen uusimmista stabiilisuus- ja säätösuunnittelutuloksista verkottuneiden säätöjärjestelmien osalta, tiivistelmän julkaisuista sekä alkuperäiset julkaisut. Myös tutkimuksessa sovelletut säätösuunnittelumenetelmät esitellään. Työssä esitetään useita uusia PID-säätimen viritysmenetelmiä verkottuneisiin säätöjärjestelmiin sekä PID-viritystyökalu, joka helpottaa tulosten hyödyntämistä. Verkottuneiden säätöjärjestelmien algoritmitestaukseen työssä ehdotetaan käytettäväksi MoCoNet-testausalustaa, joka on kehitetty Teknillisessä korkeakoulussa. Työn tulokset osoittavat, että PID-säädintä voidaan hyvin käyttää verkottuneissa säätöjärjestelmissä, mikäli säädin viritetään huomioiden näiden järjestelmien erityispiirteet kuten muuttuvat viiveet ja pakettihäviöt.

Asiasanat	Verkottuneet säätöjärjestelmät, PID-säädin, viritys, muuttuva viive, langaton automaatio		
ISBN (painettu)	978-951-22-9633-0	ISSN (painettu)	0356-0872
ISBN (pdf)	978-951-22-9634-7	ISSN (pdf)	
Kieli	Englanti	Sivumäärä	98 + 62
Julkaisija	Teknillinen korkeakoulu, Automaatio- ja systeemitekniikan laitos		
Painetun väitöskirjan jakelu	Teknillinen korkeakoulu, Automaatio- ja systeemitekniikan laitos		

☒ Luettavissa verkossa osoitteessa http://lib.tkk.fi/Diss/2008/isbn9789512296347

Preface

During the almost eight years I have spent in the Control Engineering Group (formerly known as the Laboratory), at TKK, I have had the opportunity to work with highly motivated and skilled researchers in a relaxed atmosphere. This group of about 50 people is led by my supervisor, Prof. Heikki Koivo. Heikki introduced me to the field of wireless sensor and actuator networks at the end of 2003 after which time I started the research that forms the basis of this thesis. In many ways, Heikki has opened my eyes and showed me the big picture of research, both nationally and internationally. I am very grateful to him for that.

This thesis was pre-examined by two esteemed researchers of control engineering, Prof. Karl-Erik Årzén (Lund University) and Dr. Terho Jussila (Tampere University of Technology). Their detailed comments and statements were extremely valuable in improving the contents, notation and clarity of the thesis, and I would like to thank them for their efforts. Additionally, I wish to thank Mr. William Martin (TKK) for proofreading the thesis.

I want to express my gratitude to the Graduate School in Electronics, Telecommunications, and Automation (GETA), whose student I have been since 2005, for supporting my doctoral studies. In addition, I have been supported by the Finnish Foundation for Technology Promotion, the Walter Ahlström Foundation, Neles Oy:n 30-vuotissäätiö (the 30th Anniversairy Foundation of Neles Inc.), the Finnish Cultural Foundation, the Research Foundation of Helsinki University of Technology, the Jenny and Antti Wihuri Foundation, and the Automation Foundation. Their support is highly acknowledged. I would also like to thank The Finnish Funding Agency for Technology and Innovation (TEKES) for funding the ERHE (2003 – 2005) and WiSA (2006 – 2010) projects, in which I have worked at TKK.

Once the WiSA project started in 2006, I became acquainted with Associate Professor Mikael Johansson (Royal Institute of Technology), with whom I have collaborated in the field of this thesis. His ideas, positive attitude and interest in PID design have given me additional strength and motivation to work on the topic, and I want to thank him for the fruitful discussions and cooperation. At TKK, it has been a pleasure to work in the research team of wireless automation. I want to thank the members of this team, especially Prof. Riku Jäntti and M.Sc. Mikael Pohjola, for their support during the years. In addition, I am greatful to my collegues Lic.Sc. Vesa Hölttä and Dr. Timo Oksanen for the inspiring discussions during the coffee breaks… Besides serving me coffee, they have contributed to some of the publications included in this thesis, so their professional efforts are also recognized!

I am indebted to my parents for their support and the encouragement I have received during the thesis project and previous studies. My father with his three academic degrees and my mother, who started working with her doctoral thesis only a few years before the retirement from work, have by their example instilled in me the attitude that studying is worth doing. Without the inherent passion for learning, investigating and studying, this thesis would have never been written.

Finally, I would like to express my greatest gratitude to my family, my wife Anni-Kaisa and sons Julius and Kasper, whose love, support and care enabled me to complete this thesis.

Lasse Eriksson

List of Publications

[P1] L. Eriksson, "A PID Tuning Tool for Networked Control Systems", *WSEAS Transactions on Systems*, Vol. 4, Issue 1, January 2005, pp. 91-97.

[P2] L. Eriksson, H. N. Koivo, "Tuning of Discrete-Time PID Controllers in Sensor Network based Control Systems", in *Proc. 2005 IEEE International Symposium on Computational Intelligence in Robotics and Automation (CIRA2005)*, Espoo, Finland, June 27-30, 2005, 6 p.

[P3] M. Pohjola, L. Eriksson, V. Hölttä, T. Oksanen, "Platform for Monitoring and Controlling Educational Laboratory Processes over Internet", in *Proc. 2005 Congress of the International Federation of Automatic Control (IFAC)*, Prague, Czech Republic, July 3-8, 2005, 6 p.

[P4] M. Pohjola, L. Eriksson, H. Koivo, "Tuning of PID Controllers for Networked Control Systems", in *Proc. The 32nd Annual Conference of the IEEE Industrial Electronics Society (IECON'06)*, Paris, France, November 6-10, 2006, 6 p.

[P5] L. Eriksson, T. Oksanen, "PID Controller Tuning for Integrating Processes: Analysis and New Design Approach", in *Proc. Fourth International Symposium on Mechatronics and its Applications (ISMA07)*, Sharjah, UAE, March 26-29, 2007. 6 p.

[P6] L. M. Eriksson, M. Johansson, "PID Controller Tuning Rules for Varying Time-Delay Systems", in *Proc. 2007 American Control Conference (ACC 2007)*, July 11-13, 2007, New York, USA, 7 p.

[P7] L. M. Eriksson, M. Johansson, "Simple PID Tuning Rules for Varying Time-Delay Systems", in *Proc. The 46th IEEE Conference on Decision and Control (CDC 2007)*, New Orleans, LA, USA, December 12-14, 2007, 7 p.

[P8] L. M. Eriksson, H. N. Koivo, "Comparison of low-complexity controllers in varying time-delay systems", *SICE Journal of Control, Measurement, and System Integration*, Vol. 1, No. 2, 2008, pp. 111-119.

Contributions of the Author

[P1] The author carried out all the work related to the publication.

[P2],[P8] The author has developed all the methods and simulation models and written the publications under the supervision and guidance of Prof. Koivo.

[P3] The author has been the coordinator and instructor of the developed platform (MoCoNet) and participated strongly in its design, whereas M. Pohjola has mostly implemented the system. The author has written roughly half of the publication.

[P4] The author has instructed M. Pohjola in developing the methods presented in the publication and has participated in writing the publication.

[P5] The author has developed the tuning methods and rules with the help of T. Oksanen. The author has also derived the analysis results regarding the jitter margin. T. Oksanen has tested the different tuning rules in simulations and done the experiments with the agricultural tractor including the modeling work.

[P6],[P7] The author has designed the simulations, implemented the code and derived the results. M. Johansson has contributed at the idea level and in analyzing the results. The author has written the publications with the help of M. Johansson.

Nomenclature

Symbols

a	Controller gain
A, A_0, A_1	State-space model state matrix, continuous-time model
b	Set-point weight in the proportional term of the PID algorithm
B	State-space model input matrix, continuous-time model
c	Set-point weight in the derivative term of the PID algorithm
c_R	Center point of the M-circle
C	Controller transfer function
C_{PI}	PI controller transfer function
C_{PID}	PID controller transfer function
C_λ	Lambda controller transfer function
$d(k)$	PID controller's derivative term in discrete-time
$D(s)$	Laplace transform of the PID controller's derivative term
e	Error signal
$E(s)$	Laplace transform of the error signal
f	Continuous-time function
F	Positive definite matrix
$g(x)$	Equality constraint function
G	Process transfer function
$\tilde{G}$	Process model transfer function
G_c	Closed-loop system transfer function
G_f	Measurement filter transfer function
G_l	Open-loop system transfer function
G_S	Sensitivity function
G_T	Complementary sensitivity function
$\tilde{G}_+$	Non-invertible part of the process model transfer function
$\tilde{G}_-$	Invertible part of the process model transfer function
h	Sample time
$h(x)$	Inequality constraint function
H	Loop transfer function

$i(k)$	PID controller's integral term in discrete-time
$I(s)$	Laplace transform of the PID controller's integral term
J	Cost or objective function
k	Index of time
k_d	Derivative gain of the PID controller
k_i	Integral gain of the PID controller
k_p	Proportional gain of the PID controller
K	Gain of the PID controller
K_p	Static gain of a process
K_{sf}	State feedback gain matrix
K_u	Ultimate gain
K_v	Velocity gain
L	Lagrangian function
M	Value of the maximum sensitivity
n	Order of a filter
N	Derivative filter parameter
O	Filter output in the enhanced PID algorithm for wireless control systems
$p(k)$	PID controller's proportional term in discrete-time
$P(s)$	Laplace transform of the PID controller's proportional term
Q	IMC controller transfer function
r_R	Radius of the M-circle
R	Low-pass filter transfer function in IMC controller
s	Laplace variable
S	(Sub)system
t	Continuous time
T	Time-constant of a process
T_d	PID controller derivative time
T_f	Measurement filter time-constant
T_i	PID controller integration time
T_{Reset}	Integrator resetting time
T_u	Ultimate period of oscillation
u	Control signal
$U(s)$	Laplace transform of the control signal
V	Continuous differentiable function
w	Disturbance
w_1, w_2	Weights in the IERC cost function
x	State or parameter vector
y	Output signal
y_f	Filtered output signal
y_r	Reference signal
$Y(s)$	Laplace transform of the output signal

$Y_f(s)$	Laplace transform of the filtered output signal
$Y_r(s)$	Laplace transform of the reference signal
z	State vector
α	Gain parameter in the Ziegler-Nichols step response method
β	Gain parameter in the Cohen-Coon method
Γ_0, Γ_1	State-space model input matrices, discrete-time model
$\delta(t)$	Time-varying delay in continuous-time
$\delta_{\min}$	Minimum value of a variable delay
$\delta_{\max}$	Maximum value of a variable delay, the jitter margin
δ_0	Target value of the jitter margin
Δ	Uncertainty block, varying time-delay operator
ΔT	Time elapsed since a new value was communicated in the enhanced PID algorithm for wireless control systems
κ	Relative dead time
λ	Lambda tuning parameter, the closed-loop system speed parameter
λ_i, λ_j	Lagrange multipliers
λ_{IMC}	IMC controller tuning parameter
μ	Mean value
σ^2	Variance
ϕ	Initial value of a state variable
Φ	State-space model state matrix, discrete-time model
$\tilde{\Phi}$	Augmented, discrete-time, closed-loop state-space model state matrix
τ	Constant time-delay
τ_k	Delay of a measurement made at discrete time-instant k
τ_k^c	Controller computational delay at k
τ_k^{ca}	Controller-to-actuator delay at k
τ_k^{sc}	Sensor-to-controller delay at k
ω_{gc}	Gain crossover frequency

Operations

$\| \ \|$	Norm
Im(…)	Imaginary part of a complex number
Re(…)	Real part of a complex number

Abbreviations

AMIGO	Approximate M-Constrained Integral Gain Optimization
AODV	Ad Hoc On-Demand Distance Vector
BIBO	Bounded Input, Bounded Output
CAN	Controller Area Network
CARIMA	Controlled Autoregressive Integrated Moving Average
CDMA	Code Division Multiple Access
COTS	Commercial Off-The-Shelf
CSMA	Carrier Sense Multiple Access
CSMA/CA	Carrier Sense Multiple Access with Collision Avoidance
CSMA/CD	Carrier Sense Multiple Access with Collision Detection
CT	Continuous-Time
DT	Discrete-Time
FDMA	Frequency Division Multiple Access
FIFO	First In, First Out
FOLIPD	First Order Lag plus Integrator plus Delay
FOTD	First Order System with Time-Delay
GA	Genetic Algorithm
IAE	Integral of Absolute Error
IERC	Integral of Weighted Sum of Square Error and Required Control Signal Error
IPD	Integrator plus Delay
ISE	Integral of Square Error
ISM	Industrial, Scientific, Medical
ITAE	Integral of Time-Weighted Absolute Error
ITSE	Integral of Time-Weighted Square Error
KLT	See FOTD
LAN	Local Area Network
LMI	Linear Matrix Inequality
LMNR	Localized Multiple Next-hop Routing
LQG	Linear Quadratic Gaussian
LTI	Linear Time-Invariant
MAC	Medium Access Control
MB-NCS	Model-Based Networked Control Systems
MEMS	Micro-Electro-Mechanical Systems

MIMO	Multiple Input, Multiple Output
MoCoNet	Platform for Monitoring and Controlling Educational Laboratory Processes over Internet
MPC	Model Predictive Control
NCS	Networked Control Systems
Ns-2	The Network Simulator
IMC	Internal Model Control
PID	Proportional-Integral-Derivative
PiccSIM	Platform for Integrated Communications and Control Design, Simulation, Implementation and Modeling
QoS	Quality of Service
RCP	Rapid Control Prototyping
RFDE	Retarded Functional Differential Equation
SISO	Single Input, Single Output
SQP	Sequential Quadratic Programming
TCP	Transmission Control Protocol
TDMA	Time Division Multiple Access
UDP	User Datagram Protocol
WLAN	Wireless Local Area Network
WSAN	Wireless Sensor and Actuator Networks
WSN	Wireless Sensor Networks
ZOH	Zero-Order Hold

Contents

1. Introduction

1.1. Background and Motivation

Networked control systems (NCS) are feedback control systems wherein the control loops are closed through a real-time network [60]. The different components of NCS exchange information via a shared medium, a digital data network. The main motivations for using networks for data transmissions in control systems are reduced system wiring, ease of system diagnosis as all information is available everywhere in the system, and increased system agility [95]. After the breakthrough of Internet and the preceding development of computer networks in general, there are several cheap and reliable network technologies available. Because of these reasons networks are and will be even more popular in transferring real-time control data [43].

Networks, especially wireless ones, have already changed the consumer markets. Handheld computers with wireless links through which the computers can communicate, and sensors, such as cameras, are all around us. In the industry, though, the use of wireless technology is at a very early stage, although it would bring obvious benefits, because wireless networking extends the possibilities of NCS even further. Industrial applications having mobile subsystems, or just savings in cabling costs, motivate the use of wireless technologies. The wireless solutions also provide more flexibility. Sensors can be rearranged when and where needed and they can be installed in new positions in processes, where wired sensors would not be suitable, for example, on rollers.

Wireless sensor networks (WSN) have been extensively researched for over a decade, because they provide appealing possibilities for distributed, flexible and ubiquitous sensing applications, where each node in the network performs sensing, data processing and communication functions. The highly distributed nature of WSN makes them fault tolerant and adaptive to dynamically changing environments. Even though one node in the network experiences problems and is shut down, networking protocols and both sensing and data processing algorithms could adapt to the changed situation. Hence packets would not be delivered through the faulty node, routes would be re-established and data processing would adapt to a missing source of measurements. Automatic route establishment and other self-healing properties of networks are required in order to execute control tasks over wireless networks. Often the industrial applications pose stringent timing and reliability requirements, and with wireless solutions these are more difficult to meet than with wired due to the adverse properties of the radio channels [88].

The shared communication medium in NCS with communication faults that are frequent especially in wireless networks poses new challenges for control design and analysis. The traditional control theory is based on the assumption of uniform sampling of variables and it does not completely cover the cases with varying time-delays or lost and intermittent measurements. The shared medium may be reserved at the time of sampling or there might be collisions of packets. Therefore the nodes might have to wait indefinitely before retransmission. The distributed nature of networks makes it sometimes hard to share common timing information, and due to drifting of local clocks the nodes are more or less asynchronic. In addition, there are various computing tasks in NCS, and depending on the extent of the tasks and scheduling policies of the microcontrollers, the computational times may vary significantly [43], [60], [95]. Unmodeled and varying time-delays may yield instability in a closed-loop control system and hence the major challenges for control design in NCS are varying time-delays and packet loss.

During recent years the research of NCS has been rapidly growing and several solutions to cope with varying time-delays and packet loss have been proposed. Currently, it seems that there are a few main streams or methodologies that are under specific investigation. A networked control system could be modeled as a linear discrete-time state-space system with varying time-delays in the control and/or output signals. Once the control law is defined, Lyapunov-like stability criteria are applied and the conditions for the corresponding LMI (Linear Matrix Inequality) problem are presented (e.g. [58], [66]). The LMI is then solved using an efficient numerical solver. Another quite common way is to develop observers that act on the intermittent and delayed measurements and to apply LQG (Linear Quadratic Gaussian) design on top of that (e.g. [44], [74]). In all these approaches the practical applicability is questionable, since the calculations are computationally demanding and an industrial control engineer might feel uncomfortable in applying these mathematically challenging solutions in practice. The computational issues play an even more important role in the case of wireless control systems. The computations should be run on a wireless node that needs to have low power consumption and has a very limited computational power and memory available. This motivates developing computationally light controllers for NCS and analyzing how the deficiencies of the networks could be overcome by the correct choice of the controller structure and tuning parameters.

This thesis discusses the control design aspects of NCS and focuses on the PID (Proportional-Integral-Derivative) controller, its design and tuning. The PID controller is a widely applied control algorithm in wired control systems, and there is no clear reason why it would not be so in NCS or wireless automation applications. The use of PID in NCS has been researched relatively little, although the choice of its structure and its tuning turn out to be crucial for it to be applied in NCS. The PID controller is sometimes discussed in the literature in connection with NCS (e.g. [83], [101]), but quite rarely the characteristics of the networks are taken into account in any way in the controller tuning. Nevertheless, by tuning the PID appropriately, it can be made very robust against varying time-delays and packet loss. To evaluate the stability of a PID controlled system with uncertainties, such as varying time-delays, robust control techniques can be used. For

example, the robustness of different PID tuning methods for a case process with parameter uncertainties has been investigated in [72].

This thesis discusses the PID controller design and tuning in NCS, but also some practical issues are considered, such as software tools for PID tuning and remote NCS environments for verifying the results of the control design.

1.2. Objectives

The main objective of this thesis is to develop tuning methods for the PID controller such that it could maintain the stability of the control system even though the measurements would be affected by the challenges faced in networked control systems such as varying time-delays (jitter) and packet loss. On the other hand, the performance of the control system should be kept adequate despite the robust controller tuning. The primary focus in the thesis is on developing the methods and tools for PID tuning, but also tuning *rules* have an important role. Hence another objective of the thesis is to find PID tuning rules for varying time-delay systems, so that the rules could be applied with ease for a wide range of processes in the NCS setting.

Besides investigating the performance of the PID controller, other relatively simple control algorithms are also considered in the thesis. The objective here is to examine if certain other computationally light algorithms would have remarkable benefits over the PID controller in NCS or in varying time-delay systems. The aim is to justify the choice of the PID controller structure for NCS by comparing the performance from the reference tracking, robustness to delays, and robustness to disturbances and noise perspectives.

1.3. Summary of the Publications

This doctoral thesis is an article dissertation composed of a summary and eight publications. In most of the publications (6/8), new controller tuning methods or PID controller tuning rules are proposed for NCS or more generally, for varying time-delay systems. In addition, the performance of simple controllers in NCS is compared in [P8], a PID controller tuning tool for NCS is proposed in [P1] and a platform for experimenting with NCS and verifying the control design methods is presented in [P3]. An extended summary of the publications is given in Chapter 4, but the main results are also briefly reviewed in this section.

In [P1], a PID controller tuning tool for NCS is proposed. The tool provides a graphical user interface implemented in MATLAB with various options regarding plant and network modeling. With the tool, it is possible to tune the controller manually or automatically using optimization with the aid of simulation. The network dynamics are also taken into account by providing the user the possibility to define the probability distribution of the network delay. The tool also implements certain automatic tuning functions that can be used to optimize the controller performance with

respect to the selected criterion and robustness constraints. The tool implements the tuning method presented in detail in [P2]. The tuning method is based on simulating the plant with varying output delays and on controller parameter optimization, where the step response performance of the closed-loop system is maximized while robustness to delays is provided by guaranteeing the desired gain and phase margins with different values of delays. The desired gain and phase margins are used to constrain the optimization problem to obtain delay-tolerant tuning parameters. It is also possible to simulate load disturbances and measurement noise with the tool, so that the controller performance under the less than ideal circumstances of real-time operability can also be tested.

Simulation and optimization are also used in the tuning method presented in [P4], where the tuning is based on the minimization of a maximum cost over several step responses. This is done, because different realizations of the varying time-delay are used in the simulations, and minimizing the worst-case cost should yield to robust tuning. Both tuning methods presented in [P2] and [P4] are developed for a discrete-time PID controller, which is the obvious choice in NCS due to sampling and the digital networks used for communication. The difference is that in [P2] only varying process output delays are considered, whereas in [P4] the setup is fully-distributed with both process output delays and controller output delays.

The MoCoNet system is presented in [P3]. It is a platform for testing control over networks with simulated or even real processes. Using the rapid control prototyping tools [32], the implementation of any control algorithm on real processes and networks is straightforward in the simulation environment. Many of the methods proposed in the thesis have been tested and verified with the MoCoNet platform or its successor, the PiccSIM platform (see [56], [57]). The MoCoNet platform contains a network simulator that can be used for simulating the performance of NCS realistically. Hence, it is possible to test the controller performance with real processes in a NCS setting by simulating the desired network type. The platform also supports using real networks in the experiments. For example, the platform has been used for control over Internet, when the plant at the Helsinki University of Technology, Espoo, Finland, was controlled with a PID controller at the Royal Institute of Technology, Stockholm, Sweden. The controller was tuned according to the method in [P4], see [69] for details.

Publications [P5], [P6] and [P7] focus not only on developing PID tuning methods, but also take a step further and present tuning rules that are based on the methods discussed in the articles. Since the objective was to derive tuning rules, the analysis was decided to be performed in continuous-time, so that the controllers would be independent of the sampling time. To be applied in NCS, the controllers obtained by the proposed rules need to be approximated with their discrete-time equivalents with small enough sample times to ensure insignificant discretization error. The tuning method proposed in [P6], which was also used in [P5], is based on constrained multi-objective optimization. The optimization problem considers step response performance, robustness to delays and robustness to disturbances and noise, simultaneously. This is achieved by

formulating different objectives and constraints for the problem and using multi-objective optimization methods. The tuning method is essentially the same in these two papers, but in [P6] stable processes are considered and the tuning rules are based on the FOTD (First Order System with Time-Delay) approximation of the plant (a.k.a. KLT model [102]), whereas in [P5] the focus is on marginally stable integrating processes and the FOLIPD model (First Order Lag plus Integrator plus Delay). In [P6], the tuning method is applied for a process batch with different time-constants and nominal delays, and the tuning rules are based on surface identification of the optimized parameters. In [P5], a similar approach is taken and the tuning rules for a very wide range of processes with different time-constants and nominal delays are found. In both cases, the performance of the tuning rules is compared with other recently proposed rules. Furthermore, the tuning rules in [P5] are also verified with an agricultural case process. The tuning method proposed in the publication has been later extended to cover the packet loss case in NCS [16]. The simulations verify the performance of the proposed extensions.

Additionally, in [P5], as a result of the thorough analysis of the delay-robustness of the control system, an alternative way of setting the controller parameters is proposed. The advantage of this method is that the desired *jitter margin* [9] (a measure of how much additional delay the control loop tolerates) can be given into the tuning rules as an input parameter, which makes the use of the rules easy. In [P7], the objective is also to design such tuning rules, where the jitter margin can be given as an input parameter. In this case, stable processes are considered instead of integrating ones. The analysis in [P7] is based on approximating the process with the KLT model and on the AMIGO (Approximate M-constrained Integral Gain Optimization) tuning rules [103]. Using the measurement filter re-design rules and the extended plant approach proposed in [P7] it is possible to tune the PID controller according to the desired jitter margin. The design methodology and its applicability are illustrated with comparisons to the AMIGO tuning in varying time-delay systems.

Finally, in [P8], altogether five different discrete-time controllers are designed and compared in varying time-delay systems. The study also presents a controller tuning method that is applied for all the controllers in the comparison. The objective of this investigation is to find out if the PID controller could be easily outperformed by some other low-complexity controller that would be more suitable for NCS. The paper also discusses other, a bit more complex, but still relatively simple controllers, such as the fuzzy gain scheduling and model based delay-adaptive controllers, and compares their performance with the traditional PID controller. The results indicate that the PID controller is a good choice for NCS or other varying time-delay systems, as far as the delay variance and packet loss are taken into account in the controller tuning phase, using, for example, the tuning methods or rules developed in this thesis.

1.4. Contributions of the Thesis

The contributions of the author presented in this thesis can be summarized as follows:

- Development of four different tuning methods for PID controllers in NCS and varying time-delay systems in general. The tuning methods use single or multi-objective optimization and different objective and constraint functions to provide good performance, robustness to varying time-delays and robustness to disturbances and noise. The tuning methods are presented in [P2] (also applied in [P1]), [P4], [P6] (also applied in [P5]) and [P8].

- Development of two new sets of PID controller tuning rules for stable processes [P6], [P7] in varying time-delay systems. Both tuning rules take the Ziegler-Nichols approach in the sense that they are based on the FOTD model, which may be obtained by a simple step response experiment. In [P6], the closed-loop performance and the delay-tolerance are maximized, while certain robustness to disturbances and noise levels are maintained via constraints on the sensitivity and complementary sensitivity functions. In [P7], the tuning rules take the advantage of the AMIGO tuning proposed by Åström and Hägglund [103]. By re-designing the measurement filter, it is possible to derive tuning rules that depend on the desired jitter margin. This is significant for the applicability of the rules as, for instance, the maximum network delay may be an input to the rules and the plant parameters give the rest of the PID gains.

- Development of two new sets of PID controller tuning rules for marginally stable integrating processes [P5] with varying time-delays. One of the two sets of tuning rules emphasizes robustness to disturbances and noise, and the other focuses on providing the desired jitter margin. The analysis made of the jitter margin and controller performance relationship enables tuning the controller optimally in the performance sense such that a given jitter margin is satisfied. Here too, the desired jitter margin may be given as an input parameter to the tuning rules.

- Comparison of simple controller algorithms in varying time-delay systems motivated by the limited computational resources available, for example, in wireless sensor and actuator networks. Variations of IMC (Internal Model Control) controllers and fuzzy gain scheduling are developed for varying time-delay systems. The contribution is also in showing the applicability of the PID controller for varying time-delay systems as far as its tuning is done properly.

- Development of a MATLAB-based PID tuning tool for NCS that can be used to tune the controllers automatically by optimization or manually. With the tool, it is also possible to generate optimal parameter surfaces for a range of processes, and the surfaces can then be used for generating tuning rules.

- Design and development of the MoCoNet platform for remote laboratories and NCS research experiments. The platform provides tools for verifying the

performance of the developed tuning methods and rules in a realistic NCS setting with real processes.

1.5. Structure and Organization of the Thesis

The thesis is organized as follows. Chapter 2 discusses the NCS and presents the main challenges for control design. The chapter also reviews the related work focusing on the NCS stability and control design research in the literature. Chapter 3 discusses the control design and tuning methods that are applied in the publications of the thesis. Many of the sections are devoted to the PID controller, its properties, tuning methods and design in NCS, because the methods developed in the thesis are mainly for the PID controller. Chapter 4 summarizes the results of publications, and conclusions with a discussion of further research are given in Chapter 5.

2. Networked Control

This chapter gives an overview of the general structure of networked control systems (NCS) and describes their different components. The NCS are discussed from the control design point of view. The related research is also reviewed in this chapter.

2.1. Networked Control Systems

NCS are *distributed real-time control systems* consisting of the plant, sensors, controllers, actuators, and a shared data network that is used for communication between the components of the system [60]. A general NCS layout is depicted in Figure 1. The use of data networks in closed-loop control systems is mainly motivated by reduced system wiring, price, flexibility and modularity of implementation, and the possibility of having plug and play devices [6], [60]. For example, in the car industry, the CAN bus (Controller Area Network) has become very popular, because the measurements from numerous sensors can be transmitted over a single bus instead of wiring each sensor with its own cable. Nowadays, a car can have three CAN buses with different speeds; one for electronic stability control and the anti-lock braking system (high-speed), another for the driver's information management (navigation system, radio; medium speed) and a third one for the central locking system and other low-speed functions. Whereas cars are small in size, industrial processes are large and the savings in cabling can be dramatic when using a fieldbus instead of traditional cables and wires. Besides fieldbuses, Ethernet is being used in industrial environments for automation and production machine control. Ethernet provides a high-speed network and it can be implemented with standard devices such as access points, routers and hubs that are commercial off-the-shelf (COTS) equipment and thus inexpensive.

Generally, the controlled plant in NCS is assumed to be continuous-time, and thus the actuator implements zero-order hold (ZOH) holding the last control until the next one arrives or until the next sample time. Since networks are used for transmitting the measurements from the plant output to the controller, the plant has to be sampled (sample time h), which motivates the use of discrete-time controllers. The controller may be physically placed in a different location from the plant, actuators and sensors, resulting in a distributed control system. The controller can be *time-driven* or *event-driven*, so it can calculate the new control signal at discrete time instants with a constant sample time or it can calculate the control signal immediately once it gets a new measurement from the sensor. In addition, the actuator can be time or event-driven.

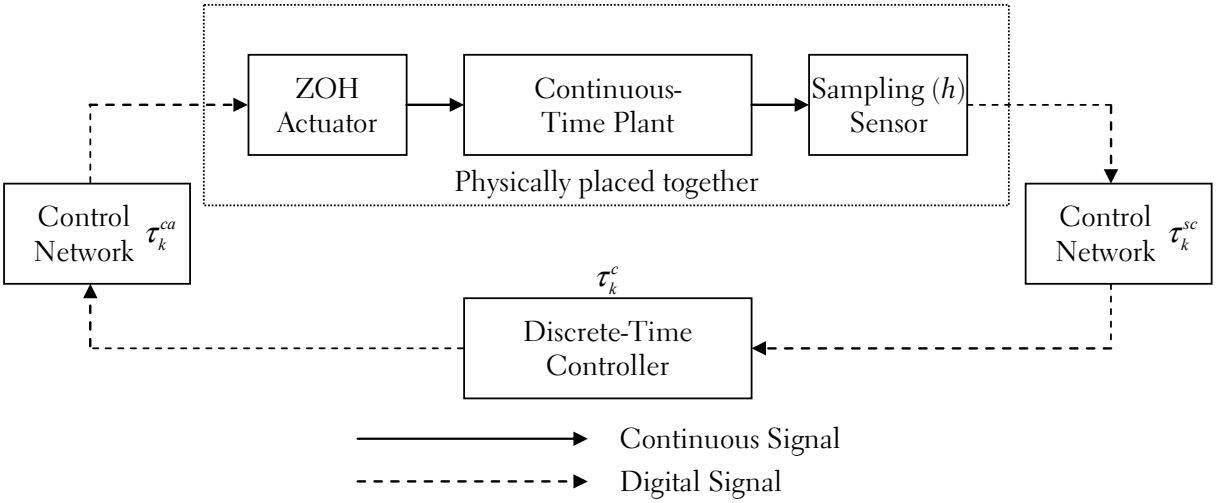

Figure 1. The NCS model (modified from [6] and [60]).

The network induces delays in the signals: τ_k^{sc} in Figure 1 denotes the sensor-to-controller delay and τ_k^{ca} the controller-to-actuator delay at time k. The controller computational delay τ_k^c can be included into τ_k^{sc} or τ_k^{ca} without loss of generality [6]. Depending on the communications protocol used, these delays can be constant, time-varying or random. Delay modeling is further discussed in Section 2.3.1.

The selection of the sample time plays an important role in NCS. Traditionally, a small sample time is chosen to approximate the continuous-time plant as closely as possible and to enable accurate control. Nevertheless, in NCS, a small sample time causes high network load and an increasing risk of network congestion, which results in longer delays and hence lower performance. Thus the network delay and the sample time are coupled, and finding an optimal balance between the two is a core requirement for achieving well performing and stable NCS [6].

The major drawback of NCS is that the network dynamics affect the stability of the control system due to the additional network delays. These delays are caused, for example, by the limited bandwidth and overhead in the communicating nodes and in the network [60]. Due to the shared medium, the nodes are unable to transmit packets simultaneously, and hence medium access control is required. There are several multiple access schemes available that differ in many respects. First of all, the access scheme can be either coordinated by a network coordinator or then distributed among the network nodes. In the former case, different nodes may transmit at a given time (Time Division Multiple Access, TDMA) or frequency (Frequency Division Multiple Access, FDMA), or then coding can be used to differentiate the signals of each transmission pair (Code Division Multiple Access, CDMA). In the distributed case, without the network coordinator, the medium access is based on first listening to the carrier wave and then transmitting if the channel is free (Carrier Sense Multiple Access, CSMA). CSMA can be complemented with collision detection (CSMA/CD), where packet collisions are detected and exponential backoff times are used before retransmissions. If collision detection is not possible, that is, if the transmitters cannot sense while transmitting, which is the case, for example, in IEEE 802.11 based WLAN's, collision

avoidance (CSMA/CA) can be used. In CSMA/CA, if the channel is idle and a node wishes to transmit, it first sends a signal telling other nodes not to send so that collisions would not occur during the transmission of the actual signal. Obviously, in the CSMA cases randomness related to timing is unavoidable and this results in varying time-delays in the network. In wireless multihop networks, the routing of packets also affects the end-to-end delay, which is varying according to the number of hops from source to destination. The multiple access schemes are discussed in more detail, for example, in [15] and [25].

Although the plant would be sampled at discrete time instants at a constant rate, the distributed and asynchronic nature of networks makes the delays time-varying, often in random fashion. Besides the constant delay of the process (though this delay could also be time-varying), the control loop experiences varying time-delays induced by the network. In the literature, these varying delays are often referred to as *jitter*. Many traditional control design methods are based on the assumption of constant time-delays, but this is rarely the case in NCS. Thus new control schemes are needed for NCS, where jitter is present. Therefore the field of networked control systems has lately been researched widely. Recent control design methodologies for NCS are presented, for example, in [82], and the different design approaches will also be reviewed in Section 2.3.2 including more recent results.

Another problem related to NCS is packet loss. Some of the packets get lost in the network due to interference, noise, node failures and packet collisions. Depending on the protocols used, a lost packet is either retransmitted or not. For example, consider the two transport layer protocols used in IP networks. In TCP (Transmission Control Protocol) messages are retransmitted in the case of lost or erroneous packets, but in UDP (User Datagram Protocol) there are no retransmissions. There is a significant difference in these two protocols with respect to how much they congest the network, but also with respect to the reliability of the connection. In the TCP, acknowledgements from the receiver to sender, indicating that the messages are received, need to be sent. This increases the network load. On the other hand, in the TCP the data arrive in correct packet order and all the data finally go through, if the network is up and running. In the UDP the packets may arrive in out-of-sequence, and there are no retransmissions. The UDP is therefore much faster than the TCP, but it is also more unreliable [24].

The computational times of the controllers may also vary depending on the complexity of the control algorithm and the number of tasks given to the microcontroller performing the computations. The computational delay may become a severe problem, if very fast dynamics are controlled, for example, a fast motor. If the sample times are in the millisecond range, low-performance microcontrollers that are often used in commercial products in embedded systems may have insufficient time for calculating the control law, especially, if the same microcontroller has to also perform other tasks such as networking. The performance of the control system may be improved by assigning the control task a high priority by a proper scheduling policy in the microcontroller. Alternatively, the computational times of the controller may significantly vary, if

more sophisticated control algorithms are used. An example of this is the model predictive controller (MPC), in which the control action is based on optimizing the control signal in the control horizon. Depending on the constraints and the operating point of the plant, the optimization problem may become very difficult to solve, which may result in unpredictable computational times [48]. If the computation takes too long, say longer than the sample time, it has to be stopped, and the previous control is applied during the next sample time. This corresponds to having varying controller output delays.

Before discussing the stability analysis, a general NCS model is presented. The model can be used, for example, in the stability analysis of NCS. The plant model is given as

$$\dot{x}(t) = Ax(t) + Bu(t), \tag{1}$$

where $x \in \mathbb{R}^n$ is the state vector, $u \in \mathbb{R}^m$ the input signal, and A and B the state and input matrices with compatible dimensions. If state feedback is used as the control law, the discrete-time controller is

$$u(kh) = -K_{sf}x(kh), \quad k = 0,1,2\ldots, \tag{2}$$

where K_{sf} is the state feedback gain matrix. Since the control law is fixed here, the network delays can be combined for analysis purposes: $\tau_k = \tau_k^{sc} + \tau_k^{ca} + \tau_k^{c}$. This is the delay of each sample and the subscript k refers to sampling instant. Assuming a time-driven sensor, an event-driven controller and that the delay is less than the sample time, i.e. $\tau_k < h$, the system equations can be written as

$$\begin{aligned} &\dot{x}(t) = Ax(t) + Bu(t), \ t \in \left[kh + \tau_k, kh + h + \tau_{k+1}\right) \\ &y(t) = Cx(t), \\ &u(t^+) = -K_{sf}x(t - \tau_k), \ t \in \left\{kh + \tau_k, k = 0,1\ldots\right\}, \end{aligned} \tag{3}$$

where y is the output, C the output matrix and $u(t^+)$ a piecewise continuous signal that only changes its value at $t = kh + \tau_k$. As the system is sampled with sample time h, the discrete-time system model becomes

$$\begin{aligned} &x(kh + h) = \Phi x(kh) + \Gamma_0(\tau_k)u(kh) + \Gamma_1(\tau_k)u(kh - h), \\ &y(kh) = Cx(kh), \end{aligned} \tag{4}$$

where

$$\Phi = e^{Ah}, \quad \Gamma_0(\tau_k) = \int_0^{h-\tau_k} e^{As}dsB, \quad \Gamma_1(\tau_k) = \int_{h-\tau_k}^{h} e^{As}dsB, \tag{5}$$

and Φ, Γ_0 and Γ_1 are the matrices of the discrete-time state-space model. Using an augmented state vector $z(kh) = \left[x^T(kh) \quad u^T(kh - h)\right]^T \in \mathbb{R}^{n+m}$, the closed-loop system can be represented as

$$z(kh+h) = \tilde{\Phi} z(kh), \tag{6}$$

where

$$\tilde{\Phi} = \begin{bmatrix} \Phi - \Gamma_0(\tau_k) K_{sf} & \Gamma_1(\tau_k) \\ -K_{sf} & 0 \end{bmatrix} \tag{7}$$

is the state matrix of the closed-loop system. If the delay is constant, i.e. $\tau_k = \tau$ for all k = 0,1,2…, the system analysis is further simplified as the system becomes time-invariant. This is the case with certain network protocols such as token-ring scheduling (IEEE 802.5). If the delay is longer than the sample time, the derivation of $\tilde{\Phi}$ is more complicated, since it is time-varying and depends on the exact schedule of the receipt of the control inputs [6], [96].

2.2. Stability of NCS

The stability of a closed-loop control system should always be the first concern when designing controllers. As discussed in the previous section, the varying time-delays induced by the network are the major source of instability in NCS. This section reviews some of the stability criteria proposed for NCS and also, in general, for varying time-delay systems.

The stability analysis of varying time-delay systems falls into several subcategories depending on the scenario. The delay can be known or unknown and its maximum value bounded or unbounded. The plant and the controller can be either continuous-time (CT) or discrete-time (DT), and also a mixed case (CT plant and DT controller) is considered in [6] and [36]. If the delay is known, the stability criterion may make use of the lengths of delays. Thus the delay-dependent criteria are presumably less conservative than the delay-independent criteria [91]. Most of the available stability criteria for unknown and varying time-delays are given in the time domain (e.g. [7], [18], [35], [42], [58], [66], [91]), but there are also frequency domain criteria such as [20] and [36].

In NCS, it is reasonable to assume that some upper bound for delay variation can be defined. Nevertheless, if the delay of a single packet exceeds the upper bound, the packet should be abandoned, but this requires considering the effect of packet loss. If most of the packets exceed the upper bound, the control system should probably be halted. For a system with time-varying but bounded delay it might be tempting to use the maximum value of the delay to design a worst-case controller, and hope that the system would be stable. It has been proved, though, that the worst-case delay design does not guarantee stability if the delay varies. Using a first order system with a varying time-delay, it is shown in [29] that in certain cases the stability region of the system is smaller with the varying time-delay than with the constant maximum delay. This indicates that a control algorithm designed for a fixed maximum delay does not necessarily stabilize a system when the delay varies between its minimum and maximum values [61].

2.2.1. Preliminaries

The stability criteria for time-delay systems are often proven using the Lyapunov-Krasovskii or the Razumikhin theorem. Most of the existing stability results for systems with varying time-delays are based on these two theorems [36]. Before these theorems are introduced, some required definitions are given.

Norms

A norm, denoted as $\|\ \|$, is a distance-measuring device that can be applied on vectors, matrices and functions. A norm of x satisfies the following assertions about $\|x\|$ (e.g. [5], [39]):

1) $\|x\| \geq 0$, and $\|x\| = 0$ if and only if $x = 0$
2) for any scalar α and vector x, $\|\alpha x\| = |\alpha|\|x\|$
3) $\|x + y\| \leq \|x\| + \|y\|$

Since any operator that fulfills the above assertions is called a norm, there are several definitions for norms. For example, in the case $x \in \mathbb{R}^n$ is a vector, the p-norm is defined by

$$\|x\|_p = \left(\sum_{i=1}^{n} |x_i|^p \right)^{1/p}. \tag{8}$$

Quite often p has one of the values $p = 1$ (l_1-norm) or $p = 2$ (Euclidean or l_2-norm). In addition, the norm $\|x\|_\infty$ (l_∞-norm) is frequently used in computations. The latter is defined by

$$\|x\|_\infty = \max_i |x_i|. \tag{9}$$

A function norm similar to (8) may be defined as follows. Let E be a set and μ a measure on E. Define the linear space $L_p(E, \mu)$, $1 \leq p < \infty$, to be all equivalence classes of functions that are measurable and p^{th}-power summable on E. Then the p-norm of function f may be defined by

$$\|f\|_p = \left(\int_E |f|^p \, d\mu \right)^{1/p}, \quad f \in L_p(E, \mu). \tag{10}$$

For a more detailed definition of the norm (10), see [5], p. 115.

Stability

There are several definitions of stability, which is a fundamental property of a system. By stability, we usually mean that a *stable* system remains under control and reacts to its inputs with reasonable output. Conversely, it is difficult to observe apparent relation between the input and the output of an *unstable* system. The stability of a linear time-invariant (LTI) system (model) may be analyzed

by bounded input, bounded output (BIBO) stability. For nonlinear systems, the Lyapunov techniques are more appropriate.

The stability definitions below consider an autonomous dynamical system which satisfies

$$\dot{x} = f(x,t), \quad x(t_0) = x_0, \quad x \in \mathbb{R}^n. \tag{11}$$

BIBO stability: A system is bounded input, bounded output stable, if, for every bounded input, the output remains bounded for all time. A LTI system is BIBO stable, if the roots of the system's characteristic equation lie in the left half of the *s*-plane [68]. For nonlinear systems, however, BIBO stability is not an adequate definition for stability. Hence the following generalized stability definitions are briefly discussed here.

Lyapunov stability (local stability): Consider the dynamical system (11) in an initial state x_0 (at $t = t_0$) near to an equilibrium state x_e. In the Lyapunov sense, x_e is stable, if for any $\varepsilon > 0$, there exists a $\delta(t_0, \varepsilon) > 0$ such that $\|x_0 - x_e\| < \delta$ results in $\|x(t) - x_e\| < \varepsilon$ for all $t > t_0$ [68]. Here $\| \ \|$ refers to the Euclidean norm. This definition of stability basically says that if an equilibrium state is Lyapunov stable and the initial state is close to it, the trajectory of the state will remain within a given neighborhood of the equilibrium.

Uniform stability: The Lyapunov stability is defined at a time instant t_0. In order to have a uniformly stable equilibrium state x_e, we insist that δ is *not* a function of t_0, so that the stability definition above applies for all t_0 [53].

Asymptotic stability: The equilibrium state x_e of the dynamical system (11) is asymptotically stable, if it is Lyapunov stable, and in addition, there is a $\delta(t_0)$ such that every motion starting at t_0 in the δ neighborhood of x_e, i.e. $\|x(t_0) - x_e\| < \delta$, converges to x_e as $t \to \infty$ [53], [68].

Uniform asymptotic stability: For x_e to be a uniformly asymptotically stable state, x_e needs to be uniformly stable and there must exist a δ that is independent of t_0 so that the asymptotic stability holds for x_e. Further, it is required that the convergence in the definition of asymptotic stability is uniform. Note that uniform stability is only important for time-varying systems, because for time-invariant systems, stability implies uniform stability and asymptotic stability implies uniform asymptotic stability [53], [68].

Global stability: An equilibrium state x_e is globally stable, if it is stable for all initial conditions $x_0 \in \mathbb{R}^n$. Uniform and asymptotic global stability are defined similarly as above in the local stability definitions, but instead of only stable x_e it is required that the equilibrium state is globally stable [53], [68].

The stability of a system can be investigated using the Lyapunov *methods*. In the direct method of Lyapunov, it is possible to determine the stability without integrating the system differential equations (11). The method is based on defining a Lyapunov function *V*, which in some sense acts

as an energy measure of the system. Observing the rate of change of the energy of the system allows ascertaining the stability of the system. This approach is applied in the Lyapunov-Krasovskii and Razumikhin theorems that are briefly reviewed in Sections 2.2.2 and 2.2.3. The detailed discussion of the Lyapunov methods is omitted here, see for example [20], [53], [68] for details.

Linear matrix inequalities

Many NCS stability criteria based on the Lyapunov-Krasovskii and Razumikhin theorems are given in the form of LMIs (e.g. [35], [66]). This is justified, since there are efficient computational methods available to solve them. LMIs have the general form

$$F(x) = F_0 + \sum_{i=1}^{m} x_i F_i > 0\,, \tag{12}$$

where $x \in \mathbb{R}^m$ and $F_i \in \mathbb{R}^{n \times n}$. The inequality $F(x) > 0$ means that $F(x)$ is a positive definite matrix. The symmetric matrices F_i, $i = 0,1,2\ldots,m$ are fixed and x is the variable. The problem is to find whether or not there exists x such that inequality (12) holds or not [20], [84]. The LMIs can be applied, for example, for finding a positive definite function of the system states, whose derivative with respect to time is negative definite. This problem arises from proving the stability of a linear system via the Lyapunov method.

Retarded functional differential equations

Time-delay systems can be described by retarded functional differential equations (RFDEs) that are generally given as

$$\dot{x}(t) = f(t, x_t)\,, \tag{13}$$

where $x(t) \in \mathbb{R}^n$ is the system state variable and f describes the evolution of the states in time t. The derivative of the state depends on time, but also on $x(\theta)$ for $t - \delta_{\max} \le \theta \le t$, where $\delta_{\max}$ is the maximum delay of the system. The evolution of the states thus depends on the past values of the state x_t defined by

$$x_t = x(t + \theta),\ -\delta_{\max} \le \theta \le 0\,. \tag{14}$$

The initial value of the state variable in a time interval of length $\delta_{\max}$ from $t_0 - \delta_{\max}$ to t_0 is defined

$$x(t_0 + \theta) = \phi(\theta),\ -\delta_{\max} \le \theta \le 0\,. \tag{15}$$

Let $y(t)$ be a solution of (13). For system stability considerations it is important to study the system's behavior when $x(t)$ deviates from the solution $y(t)$. For these investigations it is reasonable to assume that (13) has the trivial solution $x(t) = 0$. If the stability of a nontrivial solution needs to be examined, the variable transformation $z(t) = x(t) - y(t)$ gives the RFDE, which has the trivial solution [20].

2.2.2. Lyapunov-Krasovskii Theorem

The Lyapunov-Krasovskii theorem uses the following definition of norm of the function $\phi \in C([a,b],\mathbb{R}^n)$, where C is a set of $\mathbb{R}^n$-valued continuous functions on the closed interval $[a, b]$. The continuous norm of the function ϕ is defined

$$\|\phi\|_c = \max_{a\le\theta\le b}\|\phi(\theta)\|, \tag{16}$$

where the vector norm $\|\ \|$ represents the 2-norm [20].

Consider the function f in (13), and suppose that u, v and w are continuous nondecreasing functions, where additionally $u(p)$ and $v(p)$ are positive for $p > 0$, and $u(0) = v(0) = 0$. If there exists a continuous differentiable functional V such that

$$u(\|\phi(0)\|) \le V(t,\phi) \le v(\|\phi\|_c), \text{ and} \tag{17}$$

$$\dot{V}(t,\phi) \le -w(\|\phi(0)\|), \tag{18}$$

then the trivial solution of (13) is uniformly stable. If $w(p) > 0$ for $p > 0$, then it is uniformly asymptotically stable. If, in addition, $u(p) \to \infty$ as $p \to \infty$, then it is globally uniformly asymptotically stable. The proof of the theorem is given, for example, in [20]. The functional V here measures in some sense the deviation of the system states from the trivial solution.

2.2.3. Razumikhin Theorem

The basic idea of the Razumikhin theorem is that function $V(x)$ in some sense measures the size of $x(t)$. Let the function

$$\bar{V}(x_t) = \max_{\theta\in[-\delta_{\max},0]} V(x(t+\theta)) \tag{19}$$

serve as such measure of the size of x_t. In order to ensure that $\bar{V}(x_t)$ does not grow, it is only necessary that $\dot{V}(x(t))$ is not positive whenever $V(x(t)) = \bar{V}(x_t)$. More formally, the Razumikhin theorem is given as follows. Let f, u, v and w be defined as in Section 2.2.2, but, in addition, v is strictly increasing. If there exists a continuously differentiable function V such that

$$u(\|x\|) \le V(t,x) \le v(\|x\|),\ t\in\mathbb{R},\ x\in\mathbb{R}^n, \tag{20}$$

and the derivative of V along the solution $x(t)$ of (13) satisfies

$$\dot{V}(t,x(t)) \le -w(\|x(t)\|) \text{ whenever } V(t+\theta,x(t+\theta)) \le V(t,x(t)) \tag{21}$$

for $\theta\in[-\delta_{\max},0]$, then the system (13) is uniformly stable [20]. Asymptotical and global stability can be proven by more strict conditions, but these as well as the proof of the theorem are left out of the presentation here (see [20] for details).

The results based on the Lyapunov-Krasovskii theorem usually prove the stability of a time-delay system, where the delay is bounded both in length and variation [36]. The variation of the delay parameters may be bounded, for example, by $|\dot{\delta}|<1$ [20]. Some new results show that the stability can be proven with the Lyapunov-Krasovskii theorem without the limitation for delay variation [18], [58]. On the other hand, the Razumikhin theorem can be used for proving the stability of a time-delay system, where the delay is bounded, but arbitrarily fast varying (see e.g. [7], [42]). The conservativeness of the stability criteria is considered and compared in [18].

2.2.4. Frequency Domain Approaches

Whereas the stability theorems discussed in the previous sections deal with functions of time, this section reviews some of the recently proposed frequency domain approaches for the stability considerations of varying time-delay systems. In [20] and [36], the stability criteria are based on modeling the varying time-delay as an uncertainty in the feedback loop, and on the use of the small gain theorem [94]. The small gain theorem states that if S and Δ are interconnected systems as in Figure 2a, and they are both causal, linear, and BIBO L_2-stable, then the S-Δ –loop is BIBO L_2-stable if

$$\|S\Delta\|_2 < 1. \tag{22}$$

In [20], the following continuous-time system is considered

$$\begin{aligned} &\dot{z}(t) = A_0(t)z(t) + A_1(t)z\left(t-\delta(t)\right), \\ &0 < \delta_{\min} \le \delta(t) \le \delta_{\max}, \\ &\dot{\delta}(t) \le \rho, \quad 0 \le \rho < 1. \end{aligned} \tag{23}$$

Here A_0 and A_1 are the system state matrices, and the delay function $\delta(t)$ is bounded between the minimum and maximum values, $\delta_{\min}$ and $\delta_{\max}$. The derivative of the delay is bounded so that the delay cannot increase faster than time. Since the delay variance is small, it is reasonable to separate the system such that the forward part (S in Figure 2a) only has a constant delay whereas the feedback part (Δ in Figure 2a) includes the time-varying delay. Using the small gain theorem the BIBO stability of the system can be proven.

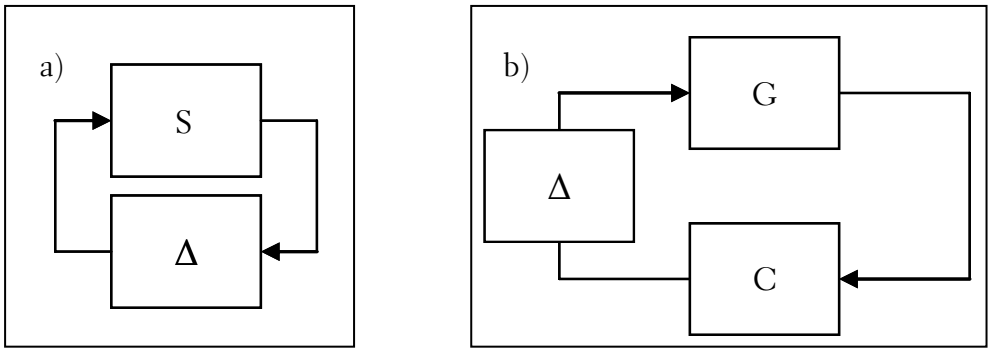

Figure 2. a) Feedback interconnection, b) Control system with a controller output uncertainty.

In [36], quite general stability results of linear control systems with varying time-delays are given. The stability criteria are derived for the similar setting as in Figure 2b, where G is a process, C is a controller and Δ is a varying time-delay. The stability criteria are derived for three different setups: 1) process and controller being CT, 2) process being CT and controller being DT, and 3) process and controller being DT. In all the cases the stability criteria are somewhat similar, but here the result is presented only for the CT-CT case, which is also used in certain publications of this thesis. Assuming that the closed-loop system in Figure 2b is stable for zero delay (delay caused by Δ-block), it is also stable for all varying time-delays defined by

$$\Delta(v) = v\big(t - \delta(t)\big),\ 0 \le \delta(t) \le \delta_{\max}, \tag{24}$$

if

$$\left|\frac{G(j\omega)C(j\omega)}{1+G(j\omega)C(j\omega)}\right| < \frac{1}{\delta_{\max}\omega},\ \forall \omega \in [0,\infty[\,. \tag{25}$$

In (24), Δ is the delay operator, which delays its incoming signal v by the delay $\delta(t)$. The delay is bounded by its minimum value 0 and maximum value $\delta_{\max}$, and in (25) ω is the frequency. The delay variation is not constrained in any way in this or the other criteria proposed in [36], which together with the rather simple form makes them very usable. Nevertheless, the criteria are only sufficient but not necessary conditions for stability, and hence in some conditions they may become conservative. Visualization of (25) is useful for the interpretation of the criterion. Notice that the left side of (25) is the closed-loop frequency response magnitude, which is bounded by a curve (right side of (25)) that is inversely proportional to the frequency. The bounding effect of $\delta_{\max}$ can be clearly seen if these two curves are drawn with respect to the frequency on a logarithmic scale, see [P7], Fig. 2.

Based on the CT-DT version of (25), also derived in [36], Cervin *et al.* [9] have defined the concept of *jitter margin*, which is extensively used in this thesis. Given the system in Figure 2b, the jitter margin is defined as the largest number $\delta_{\max}(\tau)$, for which closed-loop stability is guaranteed for any time-varying delay $\Delta \in \big[\tau, \tau + \delta_{\max}(\tau)\big]$, where τ is a constant delay. As discussed in [9], the possible constant delay part of Δ should be included in $G(j\omega)$, the plant model, to avoid unnecessary conservativeness of the condition (25). In this thesis, it is assumed that all constant delays are part of the process. In the continuous-time case, we may define the jitter margin as the greatest lower bound $\delta_{\max}$, for which the condition (25) still holds.

$$\delta_{\max} = \inf_{\omega \in [0,\infty[} \left|\frac{1+G(j\omega)C(j\omega)}{j\omega G(j\omega)C(j\omega)}\right| \tag{26}$$

As discussed above, (25) and other stability criteria proposed in [36] are conservative and discussions on their conservativeness have arisen in the literature, even though it is shown in [36] that the criteria are not very conservative. Nevertheless, the criteria have been further developed to

improve the performance aspect. The way to do this is to have certain assumptions of the delay dynamics, so that they can be taken into account in the criteria. An example of such modification is Mirkin's lemma [49], which shows that the jitter margin criterion can be made 57 % less conservative, if the delay has the sawtooth shape. This corresponds to a varying sampling interval or the packet loss case in NCS. The sawtooth delay has the following characteristics:

$$\delta(t) = t - t_k \;\; \forall t \in [t_k, t_{k+1}), \quad t_{k+1} - t_k \leq \delta_{\max} . \tag{27}$$

The shape of the sawtooth delay is depicted in Figure 3. On the vertical axis, the delay is shown in sample times, and the horizontal axis represents the time in samples. In the figure, 70 % random packet loss is assumed and the figure shows how the delay changes from sample to sample in such a lossy communication link. Obviously, in the figure, only one possible realization of the delay is represented, but the shape and behavior of the sawtooth delay are clearly seen.

Mirkin's lemma is useful in the control design of wireless networked control systems, where packet loss is a major concern. The case study in [16] (extended version of [P5]) demonstrates the use of the lemma when tuning a PID controller for maximum performance while the stability of the closed-loop system is guaranteed for a maximum of ten consecutive lost packets.

2.3. Control Design in NCS

This section discusses the recent advances in the control design of NCS, and presents some of the proposed approaches to deal with the challenges of NCS. It should be noted that the NCS are generally approached from two directions, either from communications and networking or control points of view. The review presented here considers only the control design approaches.

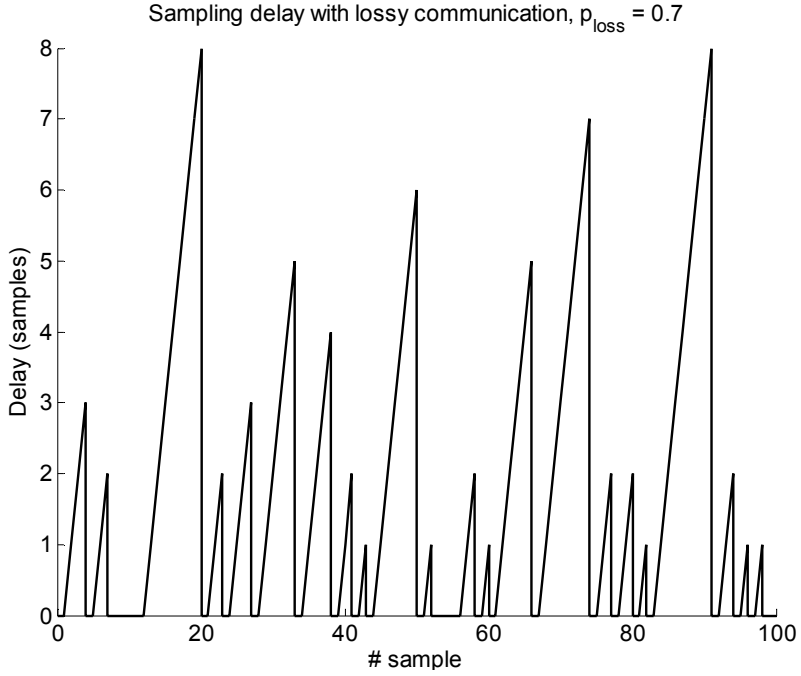

Figure 3. An example of the sawtooth delay.

First, network modeling is discussed, since the models have an important role in many control design approaches, and then the different design methodologies are reviewed. One section is devoted to wireless sensor and actuator networks, because they are currently under extensive research.

The literature on control design in NCS often focuses on the network-induced delay, and numerous methods for dealing with the delay have been proposed. The network delays are the main source of performance degradation and instability in NCS, and thus the research is well justified [82]. There are also a high number of papers discussing control over packet-dropping links (see e.g. [21]), and packet loss is indeed the other major concern in NCS. Some papers deal with both above mentioned problems simultaneously (e.g. [78]).

The controllers proposed in this field differ in many respects and there is no standard solution that could be used. One important issue is the choice of controller architecture. Certain controllers use observers to compensate for the delays and losses, but others act directly on measurements that are in some cases first filtered. The controllers discussed in this section represent different architectures and are based on a variety of control design methods. Quite rarely, though, the design of PID controllers or their robust tuning for NCS is discussed in the literature. These are highly relevant issues because of the simplicity of the PID algorithm, which especially in wireless automation applications is favorable due to the limited resources of computation and communication. The state-of-the-art PID design for NCS will be discussed in Section 3.4 after presenting the PID algorithm in detail.

2.3.1. Network Delay Models

There are several sources of delays in NCS. Not only the network dynamics affect the total delay, but also the signal processing and computational delays that depend on the scheduling policies should be taken into account. The network could also be exposed to failures, which would increase the delay variance. Furthermore, if all the components of the NCS are time-driven, there is an additional synchronization delay, because the components have to wait until the next sample instant until they can act [6], [70]. For example, the controller might receive a measurement from the plant $0.1h$ after the measurement is made, but it would have to wait until the next sample instant until the control signal would be transmitted to the actuator, which again would wait until the next sample instant before actuation of the control command.

Network modeling is in many cases a prerequisite for control design. There are several models available for network delays depending on the network type and protocols used. Generally applicable network delay models are considered in [60], where three different models are discussed. These are: 1) constant delay, 2) random delay that is independent from transmission to transmission, and 3) random delay with probability distributions governed by an underlying Markov chain. It is also suggested that the computational delays could be embedded in these delay models.

The constant delay model is good for cases, where the process dynamics are much slower than those of the networks and network delays are significantly smaller than process time-constants and delays. In this case, the network delays might always be less than the sample time and if the system is time-driven, the variance of the delay has no effect. The network delay experienced by the controller or the actuator would always be one sample time in such a case.

The independent random delay model is justified, because there are several events in the network that can cause asynchronic behavior for communication. Under the shared medium, not all the nodes can transmit simultaneously and thus sometimes they might need to wait for the network to be idle. In the case of packet collisions, there could be a random backoff time after which the nodes would try to transmit again. Altogether these properties make the delays vary in a random fashion, possibly according to some probability distribution.

The independent random delay models do not capture the effect of a high network load that often leads to correlated random delays, meaning that a delay value is actually dependent on the previous value. If the network is experiencing heavy traffic, it is probable that all packets suffer from long transmission delays until the load decreases. If varying network queues or network load need to be modeled, one can use delay distributions that are governed by a Markov chain [60].

Some studies consider the modeling of Internet delays. Quite often in these studies there are two computers involved, ping commands are executed and round trip times are recorded over a period of time. The delay distributions in such cases resemble exponential distributions for relatively short distances and Gaussian distributions for relatively long distances. The latter distribution can be explained by considering each node in the Internet as a FIFO (First In, First Out) queue with random arrivals and exponential service time. As the distances become long, there are several queue processes on the way, and the total delay is a sum of a large number of exponentially distributed independent random variables, resulting in Gaussian distribution. The parameters of the distributions need to be measured frequently, because of the huge growth and development of the Internet [61]. The generalized exponential distribution has also been proposed to model IP network delays [83].

2.3.2. Control Design Methodologies

In [82], the recent control design methodologies for networked control are reviewed. Despite the methodology chosen, the objective is to control the NCS so that stability is guaranteed while providing good performance for the closed-loop control system. The basic concepts of the control algorithms are presented here. The naming of the different methodologies is according to [82].

The *augmented deterministic discrete-time model methodology* [23] is based on discrete-time state-space models. The controller uses j past measurements $z(k) = y(k - i)$, $i = \{1,\ldots,j\}$ to calculate the control signal at k. The network delays are handled by augmenting the delays into the full system

state model, and the stability for periodic delays is proven based on the eigenvalues of the augmented system state transition matrix.

The basic idea of *queuing methodology* is to use observers and predictors to compensate for the delay and hence to make the NCS time-invariant. The methodology is based on queuing mechanisms that are used to reshape random network delays to deterministic delays. The approach presented in [47] uses an observer to estimate the plant states and a predictor to compute the predictive control based on past output measurements. The control and past output measurements are stored in FIFO queues and shift registers, and these are located before and after the controller in the control loop. First, the past measurements are used to estimate the plant state at $k - \theta + 1$, where θ is the size of the shift register between the sensor and the observer. Next, using the previous estimate, the plant state is predicted at $k + \mu$, where μ is the size of the register after the controller. The predictive control signal $u(k + \mu)$ is then calculated and stored in the shift register. Since both the observer and the predictor are model-based, the performance of the system highly depends on model accuracy.

Nilsson [60] developed an *optimal stochastic control methodology* for NCS. In this work the delay was assumed to be random, but less than one sample time. Later Lincoln [43] extended the optimal control methodology for delays that are longer than one sample time. The basic idea of this approach is to formulate the problem as a LQG problem. The system dynamics are given in state-space and the optimal controller gain is solved from the formulated LQG problem using dynamic programming. Solving the problem requires past delay and full state information, and for the latter the Kalman filter may be applied.

The *perturbation methodology* [87] considers the difference between the current plant output values and the most recently transmitted plant output values as a perturbation to the system and searches for limits to this error. The stability is proven using the Lyapunov approach on the dynamics of the error. Several assumptions are made, including error free communications, fast sampling and noiseless observations, but the plant and the controller may be nonlinear and time-variant. In this framework the network resides only between the sensor and the controller, not between the controller and the actuator.

The *sampling time scheduling methodology* can be used for determining the sensor sample times of NCS based on the delay sensitivity of each control loop in the network. The sensitivity to delay is first analyzed using general frequency domain analysis on the worst-case delay bound. Under certain conditions this methodology leads to optimal network utilization [82].

In *robust control methodology* the network-induced delays are treated as perturbations to the nominal system, and the control design is performed in the frequency domain using robust control theory. A major advantage of this approach is that there is no need to know the exact delay distributions in advance. In [22], the network delays are assumed bounded and they are modeled as simultaneous multiplicative perturbations. Using the Padé approximation, the H_∞/μ-synthesis

may be applied and certain multiplicative uncertainty weights are chosen so that the uncertain delays are covered by the controller. Robust performance for randomly time-delayed systems may be achieved using this methodology.

Fuzzy logic modulation methodology takes an advantage of fuzzy logic to update the controller gains based on the error signal between the reference and the actual output of the system, and the output of the controller [3]. This approach is somewhat similar to the fuzzy gain scheduler presented in [P8]. Nevertheless, in [3], updating the gain of a single PI controller based on the error and controller output is considered, whereas the scheduler in [P8] updates the weighting of several PID controllers and computes the weighted sum of controller outputs based on the current delay value. The fuzzy logic modulation methodology includes online and offline membership function design by optimization.

Event-based methodology uses the system motion as the reference of the system instead of time. For example, the distance traveled by an end-effector of a robotic manipulator $y(t)$ could be converted to a motion reference s by a certain mapping. A planner uses the motion reference as an input to calculate the reference $r(s)$, which is used in system control. In a way, the event-based methodology maps the time space into event space and the system stability no longer depends on time. Thus the network-induced delays will not destabilize the system [82].

End-user control adaptation methodology is based on the ability to measure the traffic conditions of the network, and the controller parameters are adapted accordingly. In this methodology, the controller can request and update the Quality-of-Service (QoS) conditions from the network, and if the desired QoS requirements cannot be met, the controller parameters are adjusted aiming for the best possible performance [81].

Besides the above mentioned methodologies, there is a common approach to design controllers for NCS based on modeling the system as a delay-differential equation. The stability of the system is proven using either the Lyapunov-Krasovskii (e.g. [54], [66], [78], [99]) or the Razumikhin (e.g. [7], [42]) theorems. Often the control law is the state feedback requiring an observer. The most common approach is to use Kalman filters for state observers. Kalman filtering for wide, highly distributed, wireless sensor network applications is discussed in [73], where a critical value for the required rate of observation arrivals to the filter under unreliable communication conditions is derived.

There are a number of LQG controllers developed for networked control. Often the studies consider either the problem of varying time-delays or packet loss, but rarely both at the same time. Nilsson [60] considered the problem of controlling a continuous-time linear plant sampled at a constant rate over a network with sensor-to-controller and controller-to-actuator delays. The actuator was assumed to be event-driven and a LQG controller was developed for the case where $\tau_k^{sc} + \tau_k^{ca} < h$ (the optimal stochastic control methodology). Thus the order of the measurements is not corrupted in the network, which simplifies the problem. Lincoln and Bernhardsson [44] solved

the problem for bounded stochastic time-delays that are longer than the sample time assuming the delay distributions to be known. They also showed that the Kalman filter is an optimal observer in such a scenario and that the separation principle holds. Thus the observer and the controller may be designed separately.

Gupta *et al.* [21] developed a LQG controller for a single packet-dropping link between a sensor and a controller. They assumed a discrete-time linear plant and no network-induced delays in the loop. The controller does not require information on the statistical model of the packet-drop event. Both Imer *et al.* [33] and Sinopoli *et al.* [74] developed LQG control over unreliable communication links and compared the performance of the TCP and UDP protocols. Both studies modeled the packet losses between the sensor-to-controller and controller-to-actuator links as independent Bernoulli processes. In [33], it was shown that if the network supports acknowledgements (e.g. when TCP protocol is used), the optimal control law minimizing a quadratic performance criterion is linear. In networks not supporting acknowledgements (e.g. UDP protocol) the optimal control law remains linear if no observation noise is present. Furthermore, [74] shows that under UDP communication, the optimal control law is generally nonlinear, but if the state is fully observable and there is no observation noise, linear control law is obtained. For the TCP case, the bounds for packet arrival probabilities providing bounded infinite horizon cost function values are also derived. In the UDP case the separation principle does not hold, since the controller does not receive acknowledgements on whether the control packet has reached the actuator or not. This makes the controller design impractical, because closed-form solutions are in general unavailable. Since the performance of the controllers is somewhat similar, and UDP is much simpler to implement, designing suboptimal controllers for the general UDP-like scenarios would be desirable.

LQG control has also been developed for the case emphasizing the medium access constraints. Zhang and Hristu-Varsakelis [97] propose abandoning the ZOH and instead choosing to ignore those sensors and actuators that are not actively communicating. The study assumes a discrete-time MIMO LTI system with communication medium that cannot deliver all signals simultaneously. The plant is first extended with a communication sequence that is a time-varying matrix defining which sensors and actuators may communicate at time k. The control system is considered from the controller rather than from the plant point of view, which in a way disregards the ZOH functions in the system. This approach simplifies the analysis of the integrated communication and control systems, because communication sequences and feedback control may be designed separately. The choice of effective communication sequences is also discussed considering the controllability and observability aspects. The control design problem can often be formulated as a standard LQG problem for a periodic time-varying system. The periodicity arises due to the offline-designed communication sequences that are periodic.

An alternative approach is to take the bandwidth of the network into account and to design the controller so that the feedback loop information is maximized. This approach deals with encoding

the measurements into bits, determining the number of bits used, and how these affect the network load and delays, since they are all coupled. Studies on control over limited and shared communication medium can be found, for example, in [30] and [85]. In addition, [93] discusses decreasing the amount of communication required via distributed state estimation, and developing estimators for large-scale distributed control systems. In this case, the estimators are distributed among the nodes of the network, and control is based on local estimates. Only if the estimation error grows above certain limit, the actual state values are broadcast over the network. This framework relaxes the requirement of bandwidth, or alternatively, could lead to shorter communication delays as the network load is decreased.

In [51], the concept of model-based networked control systems (MB-NCS) is considered, where the objective is to decrease the sample rates by effective estimation and prediction of the plant behavior during relatively long sampling intervals. The sampling time is not constant, and this is often the case in real world applications because of network congestion or nondeterministic behavior of the scheduler at the sensor end. The MB-NCS approach is motivated by the fact that upon using popular industrial network protocols, longer sample times reduce the need for bandwidth more effectively than reducing the number of bits per measurement (accuracy), since the number of overhead bits in communication does not decrease in the latter case. The number of transmitted packets is more important. The control design is based on state feedback law.

In some cases of NCS, it might be justified to assume that the delay has slow dynamics. Then control schemes based on (constant) delay identification and compensation could be used. For instance, in multihop wireless networks the number of hops affects the mean delay although the delay variance could be quite large. The routing protocols and thus the routes in multihop networks are affected by network congestion, and end-to-end delay is roughly proportional to the number of hops. If the routes change only seldomly, adaptation to transmission delay might improve the control performance. One approach is to use delay-based gain scheduling and LQR-output feedback control to ensure stability and performance in such conditions [59]. Alternatively, one can use the Kalman filter with a change detection algorithm to track the delay mean changes and to apply predictive control that, in addition, effectively attenuates the high frequency jitter [90].

The problem of time-delay identification has also been discussed in connection with other applications, because slowly varying time-delays are common, for instance, in chemical process control. The delay estimation can be based on time series models, where, in general, the numerator polynomial order and the delay are coupled. In [38], the delay estimation is based on a controlled autoregressive integrated moving average (CARIMA) model, and also a self-tuning controller for varying time-delay processes is presented. In [98], the delay is obtained by maximizing the cross-correlation function of the estimated plant output without the delay and the measured output. On top of this estimation scheme a fuzzy Smith predictor based controller is developed. The Smith predictor compensates for the constant delay, and the fuzzy controller is

used to adapt to other parameter changes in the system. In [77], two approaches for time-delay estimation in nonlinear processes are considered. Either the problem can be formulated as a nonlinear programming problem, or a neural network can be trained to estimate the delay. The first approach is based on the assumption that the delay can be split into integer and fractional parts, and the nonlinear programming problem is based on calculating the gradient of the output of the neural model describing the process with respect to the fractional part of delay. These approaches could, in some extent, be applied in NCS as well.

2.3.3. Wireless Sensor and Actuator Networks

Wireless sensor networks (WSN) have been researched with great interest during recent years. The development in micro-electro-mechanical systems (MEMS) technology and in wireless communications has provided us with cheap, customizable, embedded sensor systems capable of communicating wirelessly among each other. In many applications, the use of wireless technology is well justified because of savings in cabling and installation costs. The wireless nodes do not require extensive cabling to be able to intercommunicate [75]. If the network is fully connected, meaning that each node can reach any other node over one or several hops, the information collected in the network is available for all nodes. It is obvious that in WSN-based applications the amount of information is increased, the configuration of the measurement system is more flexible, the nature of the system is more distributed and the redundancy of sensors provides better tolerance against faults than in traditional wired applications.

In order to perform automatic actions based on measurements, controllers and actuators are also needed. Wireless sensor and actuator networks (WSAN) are capable not only of observing the physical world and processing the data, but also of making decisions and performing appropriate actions based on these observations. The key features of WSAN include distributed sensing, actuation and the coordination of tasks and networking. Typical applications of WSAN are battlefield surveillance, microclimate control in buildings, home automation and environmental monitoring. The requirements of WSAN are far more demanding than those of pure sensor networks. The WSAN must support special characteristics such as real-time response and coordination among actuators [1]. WSAN enable the wireless automation applications that will be further discussed in Section 2.5.

There are two main architectures for WSAN, namely automated- and semi-automated architectures, and these are illustrated in Figure 4. In the case of automated architecture there does not exist any centralized controller or coordinator, whereas in the semi-automated architecture there is a sink node (performing control and coordination tasks) that receives all the information and transmits the control actions to the actuators of the network. Sometimes the use of automated architecture is desired because of low latency and long network lifetime. In the semi-automated architecture the central coordinator could reside several hops away from the active sensors and actuators, which would increase the delay considerably as all packets were sent back and forth in the network. This scenario would waste the energy of the nodes [1].

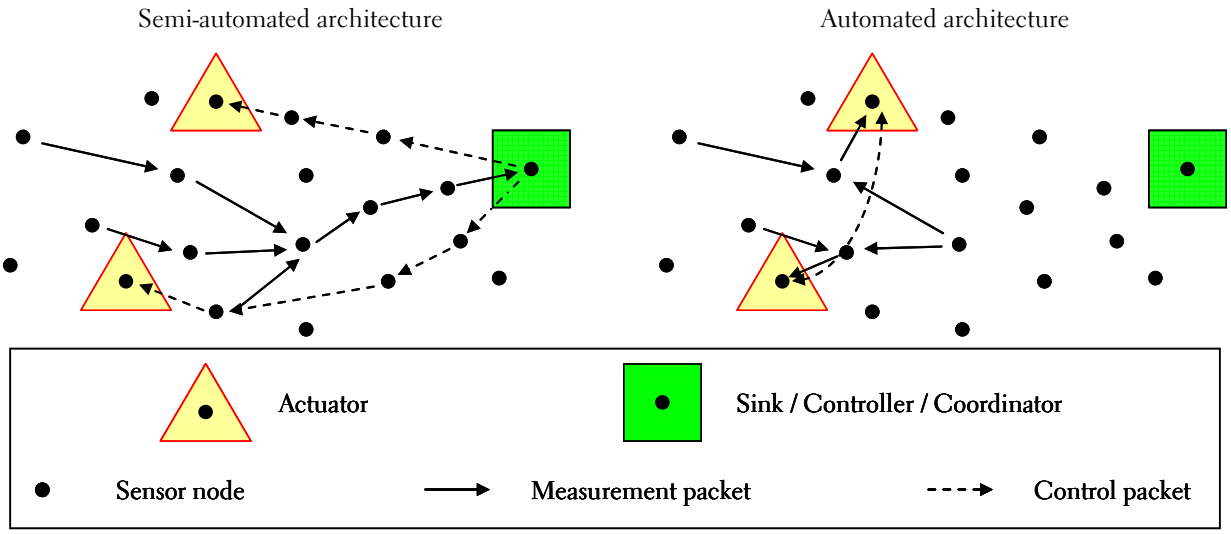

Figure 4. Semi-automated and automated architectures in WSAN [1].

On the other hand, the central coordinator is easier to design and implement. In industrial setups the actuators often require a lot of energy to run, and they need to be wired to power supplies. Thus it is reasonable to integrate the controller nodes (computation unit) with the actuators and, in this case, there is no need to have a wireless link between the controller and actuator.

One of the challenging problems in WSAN is to develop an efficient medium access control (MAC) protocol supporting the real-time requirements. MAC protocols are responsible for controlling the access of nodes to the shared medium, and hence deciding on which node may transmit, when and on which channel. This is definitely not a trivial task in a distributed and dynamically changing environment with events occurring all over the network. Besides the MAC protocol, delays in communications are due to routing. In WSAN, especially in semi-automated architecture, the source and destination nodes may be apart from each other and well out of each other's radio range, and several hops may be required to transmit a packet from the source to the destination. Each hop in the network increases the total end-to-end delay of the transmission, and for several other reasons, including processing delays, random backoff times, and packet retransmissions due to errors and losses, the delays are varying.

The wireless network is prone to packet losses, and there are a number of other problems related to the QoS in WSAN. The QoS requirements of wireless networks from the automation point of view are discussed in [15]. One way to improve the QoS of a wireless automation system is to design such a routing protocol that reduces delay and jitter, but also reliability is a key issue in control applications. It can be improved by choosing an appropriate multipath routing protocol such as the localized multiple next-hop routing protocol (LMNR) [55]. The basic idea of this routing protocol is that it finds multiple paths from source to destination, while only maintaining a single path at a time and thus avoiding the problem of packet synchronization at the destination node. Each of the source and intermediate nodes may choose from the multiple local paths to destination based on a cost function. On the node failure, this protocol has shown relatively good performance in adjusting to the changes in the environment (see e.g. [15]). It also seems to be an appropriate

candidate for WSAN control applications [57]. Compared to the well-known ad hoc on-demand distance vector (AODV) routing protocol [67], the LMNR recovers much faster from network failures and quickly finds the suitable routes.

The control design approaches to overcome the deficiencies of wireless networking are similar to those reviewed in Section 2.3.2, since WSAN are a special subgroup of NCS. The main difference is that in WSAN control design the cross-layer optimization is emphasized more than in the traditional NCS research. This is because of the constraints on energy, communication, computational power and bandwidth. An example of such a design approach is given in [52], where the effects of time-varying fading channels on control performance are evaluated. Then a key performance measure parameter to overall feedback control performance over narrowband channels is derived.

One important aspect of WSAN is time-synchronization. Implementing a highly accurate time-synchronization protocol for WSAN is a great challenge. Due to clock drifts, WS(A)N are most often asynchronic systems and this needs to be taken into account in the control design phase. It should be noted that the design guidelines and the tuning rules proposed in this thesis are also applicable in WSAN (see e.g. [16]), and the proposed methods could solve some of the problems originating from the asynchronic behavior of the networks.

2.4. Simulation and Experimentation of NCS

In order to verify that the designed controllers actually work as they should, testing the performance by simulation before actual deployment is good practice. There are various simulators for control design and, on the other hand, several simulators for networking, but a limited number of co-simulation tools and environments. If NCS are considered, the interoperability of the networks and controllers needs to be verified, since the networks have a great impact on the control performance and vice versa. Having the two components adequately well modeled and implemented on a single simulator is not a straightforward task.

This section discusses the simulation environments that are available to implement, test and verify the control designs for NCS. There are specific simulators developed for communications and control co-design, but there also exists a variety of remote laboratories that could be used for experimenting with the controllers in real NCS settings. First, the MoCoNet remote laboratory and NCS platform developed at the Helsinki University of Technology is presented, then other remote laboratories are discussed, and finally the simulators for NCS are reviewed.

2.4.1. The MoCoNet platform

MoCoNet [P3] is a platform for remote control experiments using real laboratory-scale processes. The platform also provides means for NCS control design and testing using network simulators running in real-time in connection with the processes. The process sensors and actuators can be

virtually distributed over any user defined network that is available in the platform. The user can set the parameters such as delay distributions of networks, number of hops in wireless scenario and probability of packet loss, or choose to use certain predefined networks. The platform provides remote access to any process in the laboratory that is connected to the system. The user can select controllers to be used in the experiments and tune the parameters or upload a self made controller to the system. Once the network is chosen (optional), the user can run experiments on the platform. All measurements are stored in a database, and during process runtime, the measurements are also sent to the user's screen into a scope that updates in real-time. The platform has been used for both educational and research purposes with success, for investigating and prototyping the control schemes for NCS.

The MoCoNet user interface is shown in Figure 5 and the architecture of the system is depicted in Figure 6. The processes in the laboratory are connected to the system through the "xPC Target" computer (see Figure 6), a regular PC with an I/O board running the control algorithms. The measurements are sent to the "NetSim" computer, which simulates the networks by inducing delays and dropping packets. The network simulator returns the packets to the xPC Target for using the measurements in control algorithms. The control signals may also go through the network simulator before they arrive at the actuators, which are controlled by the xPC Target. The "xPC Host", which is the server computer, manages access to the system, takes care of the communications between the server and the clients, stores the measurements and sets up the experiments according to the user preferences.

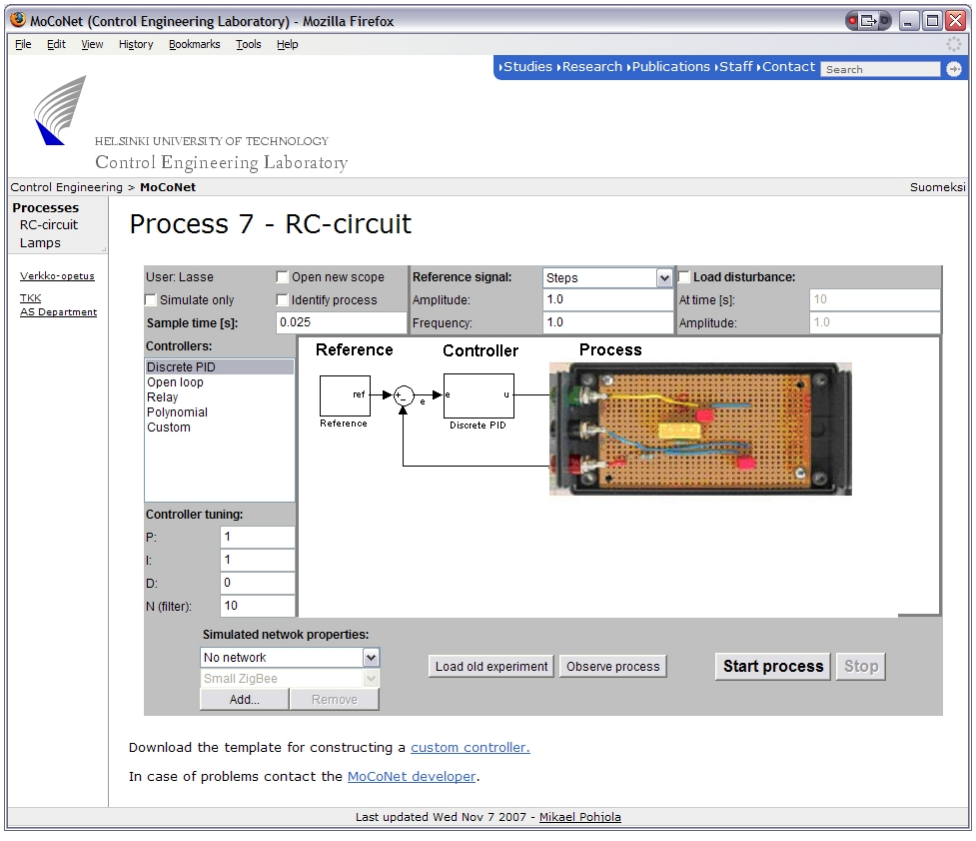

Figure 5. MoCoNet user interface.

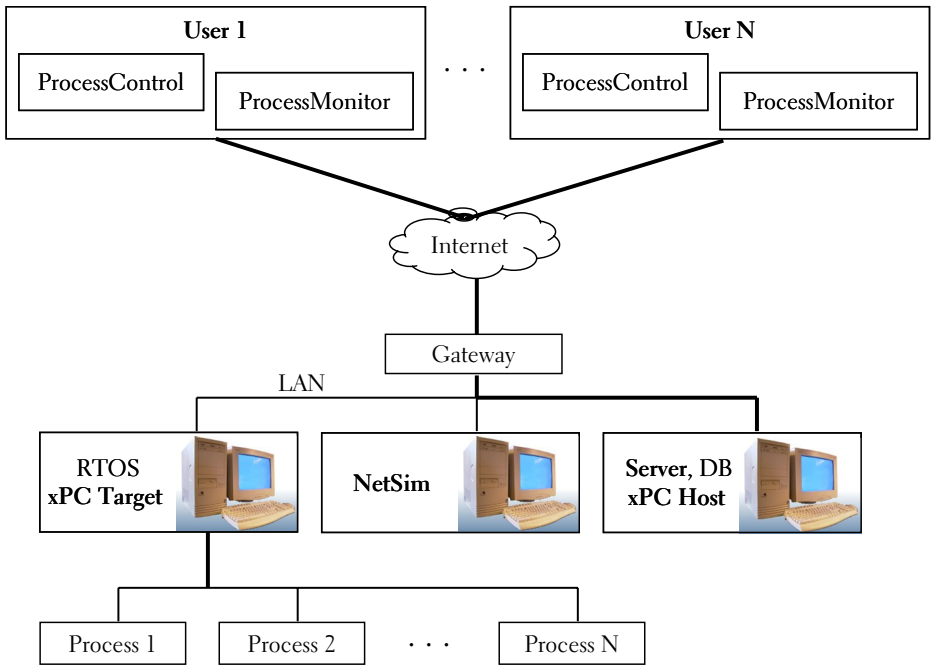

Figure 6. The architecture of the MoCoNet platform [P3].

The methodology applied in the implementation phase of the controllers in the MoCoNet platform is rapid control prototyping (RCP, see e.g. [32]), which enables, among other things, automatic code generation from the control system simulator (running on the xPC Host in Figure 6) to real-time operated controllers (xPC Target). The five steps taken in the RCP process are: 1) modeling the plant in the simulation environment, 2) model validation, 3) designing the control system, 4) simulating the control system, and 5) implementing the control system on the real plant directly from the simulation model using automatic code generation. At any point of the RCP process it might be required to go back to earlier steps and repeat the phases.

The advantages of the RCP include the automated generation of the executable code from high level simulation software, where the controller is tested against the model of the plant in advance. This means decreasing the development time of the executable code, since no low-level language programming is required. Even better, human-made errors in the code can be avoided and the extensive testing phase of the implemented code is fully removed from the prototyping process.

The development environments supporting RCP are many, for example MATLAB and LabView, and specialized hardware for the previous environments is provided by vendors such as dSPACE, Quanser and National Instruments. The MoCoNet takes advantage of MATLAB tools for RCP and hardware (I/O boards) provided by National Instruments and Advantech.

2.4.2. Other Remote Laboratories

The MoCoNet platform falls into the categories such as web-based education, remote laboratories and distance learning, but it also supports NCS experimenting. There are several remote laboratories offered from all around the world, all of them having some common, but also certain

unique features, and some of them are discussed in this section. Many of the remote labs provide a capability for controlling processes remotely and/or simulating the control systems over the Internet. To make the web-based control experiments more attractive, the user is often provided with various scopes showing measurements, live video or virtual reality illustrations either from simulations or actual processes. Remote labs offer good learning environments to students, but they also ease the implementation and experimentation for research case studies. Tools such as rapid control prototyping make the implementation of various controllers flexible and less time-consuming than traditional programming.

One of the remote laboratories relevant to MoCoNet is the Polytechnic University's (Brooklyn, New York) Internet-based real-time control engineering laboratory environment [65], where the controller may reside on the client's computer. This enables running the process over a real network, the Internet, and experiencing problems faced with high traffic. The network reliability issues and delays are also addressed. The philosophy in MoCoNet is different in this respect, as the networks are usually simulated. Nevertheless, experiments have also been conducted over the Internet such that the controller algorithm was run at the Royal Institute of Technology, Sweden, while the process was actuated at the Helsinki University of Technology, Finland [69]. The reason for simulating the networks is that the experiments can be repeated, which is not the case for live traffic. In addition, the parameters of the network are known, and this gives more information about the scenario and the reasons if the control algorithm fails.

An Automatic Control Telelab (ACT) is provided by the University of Siena [8]. The features of ACT include an easy-to-use interface, a Simulink-based interface for controller design, predefined and user-defined controller types and reference signals, controller tuning, live video and online plots, a resource management system, and safety, combined with the simplicity of adding new processes. These features, despite the Simulink-based interface for controller design, are also implemented in the MoCoNet platform, which, in addition, extends the features by having tools for network definitions through the user interface. There are several processes available online in the ACT system, such as a dc motor, a water tank, and a magnetic levitation system. The number of processes available on the MoCoNet platform is currently limited mainly due to a lack of suitable processes for online access.

Laboratory environments considering networked control are also available elsewhere. An undergraduate laboratory for networked digital control systems developed at the University of Maryland [31] combines digital control with networks and information theory. Although this is not a remote lab (over the Internet), in the typical setup there are six stations interconnected over a network. On one station with two computers and a process, the students can design controllers that act on processes over a shared medium. Issues related to NCS such as transmission delays, scheduling, and quantization as well as the choice of communication policies are taken into account in the controller design.

2.4.3. Simulation of NCS

To verify the control designs developed for NCS, simulation results are often provided in the literature and there are several simulators that could be used for this task. For pure communication and networking simulations, the ns-2 (The Network Simulator, [79]) is the de facto standard simulator in the research community. It has been developed in various DARPA-funded projects mainly in different units of the University of California. It is an Open Source platform, and it supports several wired and wireless communication protocols, including IEEE 802.11x and IEEE 802.15.4 that are frequently used in wireless sensor networks. For system level simulations, ns-2 needs to be integrated with some other simulators, such as MATLAB/Simulink or Modelica, to also cover the plant dynamics and control algorithms in the simulations. The integration of ns-2 and the process dynamics simulators is discussed in the following.

Ns-2 has been integrated with the Modelica simulation environment, see [2]. Modelica is an object-oriented language for modeling large, complex and heterogeneous physical systems. Special emphasis has been put on multi-domain modeling, and for example, mechanical, electrical, hydraulic and control systems can easily be integrated in the models [50]. The integration of ns-2 and Modelica is accomplished by implementing communication modules in both simulators so that each sensor and actuator is paired within the two simulators. The challenge in such development is to get the timing of the simulators to work so that they are synchronized. In this case, the synchronization is done such that ns-2 enslaves Modelica. Ns-2 can make the Modelica model to run for a specific time until the next event time, and hence Modelica's simulation time will never exceed that of ns-2. In [2], the integration is demonstrated with simulations of a wireless power transmission system voltage control.

A communications and control co-simulation platform that also takes the integration approach has been recently developed at the Helsinki University of Technology [56], [57]. This PiccSIM platform is based on the MoCoNet platform discussed above. The architecture of the PiccSIM platform is similar to the one shown in Figure 6, but the NetSim computer is replaced by the ns-2 network emulator that can catch live packets and pass them through the ns-2 simulation models to simulate the user specified communication and network protocols. The key features of the PiccSIM platform are:

- Support for powerful control design and implementation tools provided by MATLAB, Simulink and xPC Target Toolbox enabling automatic code generation from Simulink models for real-time execution
- Synchronized simulation of process dynamics in Simulink and network models in ns-2
- Real-time control experimenting of a true or simulated process over a user-specified simulated network
- Capability to emulate any wired/wireless networks readily available in ns-2
- Easy-to-use network configuration tool
- Accessibility over the Internet, i.e., the platform supports remote experimenting

PiccSIM provides the means to run control algorithms and networking protocols in a single environment, either with a real process or by simulating the process model. The control system is designed and the plant defined in MATLAB/Simulink. The plant block in the Simulink model may be an interface definition for the real process or a regular Simulink subsystem (e.g. a transfer function). In the case of a real process, the control system and the process are run in real-time, and the measurements are taken at specified sample times. The packets are sent to the ns-2 simulator, which also runs in real-time, and it captures the live packets from the LAN network that is used to connect the control system and the ns-2 simulator. Pure simulations may be run faster (or slower) than real-time, in which case the synchronization is done by using Simulink as the master of time enslaving ns-2.

TrueTime [27] is a well-known simulator for NCS developed at the Lund University. It is a MATLAB/Simulink-based simulator for real-time embedded control systems. Besides the blockset provided to build the simulation models of different networks including wireless networks [4], it supports simulating the temporal behavior of multi-tasking real-time kernels containing controller tasks and studying the effects of CPU and network scheduling on control performance. The user can define arbitrary scheduling policies for the CPUs. The development of TrueTime is motivated by the increasing interest for networked embedded systems. The TrueTime simulator can be integrated with the Jitterbug analysis tool [45], which calculates the control performance using a continuous-time quadratic cost function. With Jitterbug, it is possible to evaluate, for example, the effects of jitter, delays, lost samples, and aborted computations on control performance.

Besides the above mentioned tools, there is a variety of other tools mainly intended for embedded control systems design and simulation. These tools emphasize the control and scheduling co-design aspects. The scheduling of tasks at the controller affects the time of actuation of each control command, and in fact, also scheduling induces varying time-delays into the control loop. Thus some of the tools for embedded control design could be used for NCS simulations as well. Some of the alternative tools available are compared in a survey [28]. The survey compares the Jitterbug and TrueTime with AIDA and XILO (Royal Institute of Technology), Ptolemy II (University of California, Berkeley), RTSIM (RETIS Laboratory, Pisa), and Syndex and Orccad (INRIA, France). Detailed discussion of the tools is left out of the presentation here.

2.5. Towards Wireless Automation Systems

The use of wireless technology in automation systems could improve the performance through better data availability, increase the amount of information because more sensors would possibly be used in new locations, save cabling costs and increase fault tolerance via sensor redundancy. The current wireless communication protocols are evaluated in [14] emphasizing their applicability for data delivery in control systems. The differences of protocols are pointed out and why they can or cannot be used for control in wireless automation systems. It is also pointed out that currently many applications of wireless automation are related to monitoring rather than

feedback control. The end users are cautious in applying wireless for automation partly due to infeasible communication protocols and a lack of a theory for wireless automation. To make wireless a competing candidate for communications in industrial automation, the special requirements of wireless automation such as maximum delay, power consumption, maximum range and efficient use of frequency band should be better designed as a whole.

The requirements of wireless automation are often very different from the ones faced in other applications of wireless networking. Typically the control systems do not require very high bandwidth for data, but reliable and secure data transmission. Additionally, retransmissions are generally not needed even though some packets were lost. This is due to the fact that the data in the feedback loop would be old at the time of reception of the retransmitted packet. The control algorithms can be tuned such that the control system tolerates packet losses and even varying time-delays, but only up to a certain extent.

The use of COTS equipment, the ISM (Industrial, Scientific, Medical) band and standardized network protocols offers such cost benefits that using resources for developing proprietary solutions for wireless automation does not seem reasonable. This will, though, yield to the use of suboptimal wireless networking solutions for automation. For efficient and reliable wireless automation, specialized frequency bands might have to be allocated to avoid interference, and specialized hardware and networking protocols might have to be developed, which would be both expensive and time-consuming. Hence, there is currently a strong trend in the industry to agree on the standards for wireless automation that are based on the available radio and networking protocols. Two the most important efforts in this direction are ISA100 [34] and WirelessHART [89]. ISA100 is a general standard and it supports several fieldbus technologies, whereas WirelessHART is developed to support wireless communication between HART devices. ISA100 aims at monitoring and process control applications, where latencies of 100 ms or less can be tolerated. The standard defines six classes of applications (classes 0-5), from always critical to non-critical, based on the importance of the message timeliness. The classification is seen in Figure 7.

Both ISA100 and WirelessHART are based on the IEEE 802.15.4 radio and time synchronized frequency hopping communication. To reduce the risk of packet loss, every node in the network is synchronized in time and allocated a time window in which the node can transmit. There is a network coordinator that takes care of the synchronization and the transmission schedules.

The critical issue regarding wireless networking for real-time closed-loop control is how and how well the quality of service is guaranteed. There is still a lot of work to be done at the technology and networking levels, but on the other hand, also control design and, for example, the data fusion algorithms need further development. Applying wireless to non-critical applications is already possible and applications are many, but how long it will take to go a step further, all the way to the closed-loop control applications, is an open question. There are yet a lot of challenges related to the fundamental properties of wireless, before it will be applied on the large scale: the medium is open for all, meaning that adequate security and reliability need to be guaranteed.

Category	Class	Application	Description	
Safety	0	Emergency action	*(always critical)*	Importance of message timeliness increases
Control	1	Closed loop regulatory control	*(often critical)*	
	2	Closed loop supervisory control	*(usually non-critical)*	
	3	Open loop control	*(human in the loop)*	
Monitoring	4	Alerting	*Short-term operational consequence (e.g. event-based maintenance)*	
	5	Logging and downloading / uploading	*No immediate operational consequence (e.g. history collection, sequence of events, preventive maintenance)*	

Figure 7. Application classification according to the criticality of the message timeliness, by ISA [34].

One important point of view when discussing the security of wireless solutions is that if short distance radio technologies are used in the wireless automation applications, the packets cannot be received outside the plant area if the physical dimensions of the plant are large. Of course, the communication may be interfered with from a distance by using high enough transmission power and channels could be jammed. If this is the case, wireless should not be used in critical applications, but still it may prove to be useful for secondary control loops.

Wireless automation is definitely a growing area, which will change the way automation systems are operating. There are numerous opportunities that could be achieved, but also challenging threats exist for the researchers and industry to struggle with. Since a unifying (complete) theory does not exist for this multidisciplinary field of science, the gaps in the theory need to be filled with reasonable design tools such as the simulators discussed in Section 2.4.3.

3. PID Controller Design and Tuning

This chapter discusses PID controller design and tuning and brings out the PID controller's properties that are relevant in NCS. The PID algorithm is discussed in Section 3.1 and the basic variations of it are presented in continuous and discrete time domains. The controller tuning greatly affects the control system properties, such as robustness to disturbances and noise, performance and robustness to delays. Some popular PID tuning techniques are reviewed in Section 3.2, including the tuning schemes relevant to the methods applied in this thesis. In Section 3.3, some more advanced control schemes, such as Internal Model Control (IMC) and fuzzy gain scheduling, are discussed, since these can be useful techniques to complement the PID controller in NCS. By using these techniques, the PID controller tuning can be adapted online based on the delay measurement. Besides controller tuning, also the structure of the PID algorithm affects the performance in NCS. In Section 3.4, the structural perspectives of the PID controller with respect to NCS are discussed and an overview of PID tuning methods for NCS is presented. Section 3.5 presents the controller tuning method widely used in the thesis, the simulation based optimization of the controller parameters.

3.1. PID Control

The PID controller is the most common controller in control systems. For example, in the mid 1990's the PID controller was used in over 95 % of the control loops in process control [102]. The best features of the controller can only be achieved if the controller is well tuned. The tuning of PID controllers has been discussed in numerous papers and books, but seldom for systems with varying time-delays or for NCS. Some results exist, though, and [37] discusses the tuning of a continuous-time PID controller in state-dependent delay systems, whereas the discrete-time PID controller tuning is addressed in [70]. In [P2] and [P4], discrete-time PID controller tuning methods are presented that optimize the closed-loop performance and improve robustness in varying time-delay systems. [P6] and [P7] discuss the tuning of the PID controller for stable systems with varying time-delays and novel tuning rules are proposed. [P5] considers the tuning of a continuous-time PID controller in integrating processes with varying time-delays, whereas among other things, [P8] discusses the performance of time and event-driven PID controllers in the NCS framework. In addition, a PID controller tuning tool for NCS with varying time-delays is presented in [P1]. The literature on PID tuning for NCS is further reviewed in Section 3.4.

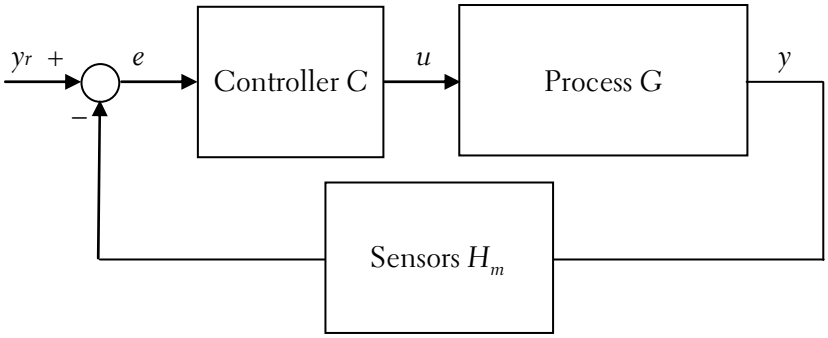

Figure 8. The classical feedback loop.

Figure 8 represents the classical control system block diagram with feedback. The process G is measured with sensors H_m and controlled by controller C, and y_r, e, u and y are reference, error, control and output signals, respectively. From here on, it is assumed that $H_m = 1$, but if needed in the analysis, the dynamics of the measurement system could be included in the process model G. An ideal continuous-time PID controller calculates the control signal u based on the error as

$$u(t) = K\left(e(t) + \frac{1}{T_i}\int_0^t e(\alpha)d\alpha + T_d\frac{de(t)}{dt} \right), \tag{28}$$

where the controller tuning parameters are K (proportional gain), T_i (integration time) and T_d (derivative time) [107]. The controller is often presented in the equivalent form

$$u(t) = k_p e(t) + k_i\int_0^t e(\alpha)d\alpha + k_d\frac{de(t)}{dt}. \tag{29}$$

The tuning parameters k_p, k_i and k_d, that are the proportional, integral and derivative *gains*, respectively, are related to those in (28) by

$$k_p = K,\ \ k_i = \frac{K}{T_i},\ \ k_d = KT_d. \tag{30}$$

The Laplace transform of the control signal $U(s)$ is given in (31) based on the Laplace transforms of the proportional, integral and derivative parts, or, $P(s)$, $I(s)$ and $D(s)$, respectively, given in (32) - (34). Here one-sided Laplace transforms have been used assuming that the initial values of the variables are zeros. The approximation in (34) limits the gain of the derivative term at high frequencies in order to reduce the effects of measurement noise.

$$U(s) = P(s) + I(s) + D(s) \tag{31}$$

$$P(s) = KE(s) = k_p E(s) \tag{32}$$

$$I(s) = \frac{K}{T_i s}E(s) = \frac{k_i}{s}E(s) \tag{33}$$

$$D(s) = KT_d sE(s) \approx \frac{KT_d s}{1 + sT_d/N}E(s) = \frac{k_d s}{1 + k_d s/(k_p N)}E(s) \tag{34}$$

Here $E(s)$ is the Laplace transform of the error signal and T_d/N is a filtering time-constant. At low frequencies the approximation (34) is quite accurate, but at higher frequencies, where measurement noise occurs, the gain is limited. The integer N is typically chosen from the range between 3 and 20 [107].

In NCS, controllers run in discrete-time. The CT controller discussed above may be discretized as follows. The proportional part of the controller is static and requires no approximation, only sampling. The backward difference method can be used in the approximation of the integral and derivative parts. The DT PID controller algorithm is given in (35) - (38).

$$u(kh) = p(kh) + i(kh) + d(kh) \tag{35}$$

$$p(kh) = k_p e(kh) \tag{36}$$

$$i(kh) = i(kh-h) + k_i h e(kh) \tag{37}$$

$$d(kh) = \frac{k_d}{k_d + k_p N h} d(kh-h) + \frac{k_p k_d N}{k_d + k_p N h}\left[e(kh) - e(kh-h)\right] \tag{38}$$

3.2. PID Tuning

PID controller tuning for NCS or varying time-delay systems is considered in [P1], [P2], [P4], [P5], [P6], [P7] and [P8]. This section discusses the PID controller tuning methods that are often applied in traditional control design in the industry. These PID tuning methods are based on some process parameters that are either obtained via process experiments or known a priori. There also exists adaptive tuning methods that de-tune the controller during the process runtime, but these are left out of the discussion. Most of the discussed tuning rules can be found among many others in [63], where PI and PID tuning rules are given for various process models, but only the most well-known and relevant PID tuning methods are reviewed here. For a more complete overview, see [62] and [63], where merely for the FOTD model, there are 27 different tuning methods listed.

3.2.1. Ziegler-Nichols Methods

The Ziegler-Nichols step response and frequency response methods are the classical tuning methods for PID controllers. They were presented already in 1942 [100], but they are still widely used in the process industry as the basis for controller tuning [102]. The step response method is based on an open-loop step response test of the process, hence requiring the process to be stable. The unit step response of the process is characterized by two parameters, α and τ. These are determined by drawing a tangent line at the inflexion point, where the slope of the step response has its maximum value. The intersections of the tangent and the coordinate axes give the process parameters as shown in Figure 9, and these are used in calculating the controller parameters. The parameters for P, PI and PID controllers obtained from the Ziegler-Nichols step response method are shown in Table 1.

Table 1. PID controller parameters in the Ziegler-Nichols step response method.

Controller	K	T_i	T_d
P	$1/\alpha$		
PI	$0.9/\alpha$	3τ	
PID	$1.2/\alpha$	2τ	$\tau/2$

Table 2. PID controller parameters in the Ziegler-Nichols frequency response method.

Controller	K	T_i	T_d
P	$0.5K_u$		
PI	$0.4K_u$	$0.8T_u$	
PID	$0.6K_u$	$0.5T_u$	$0.125T_u$

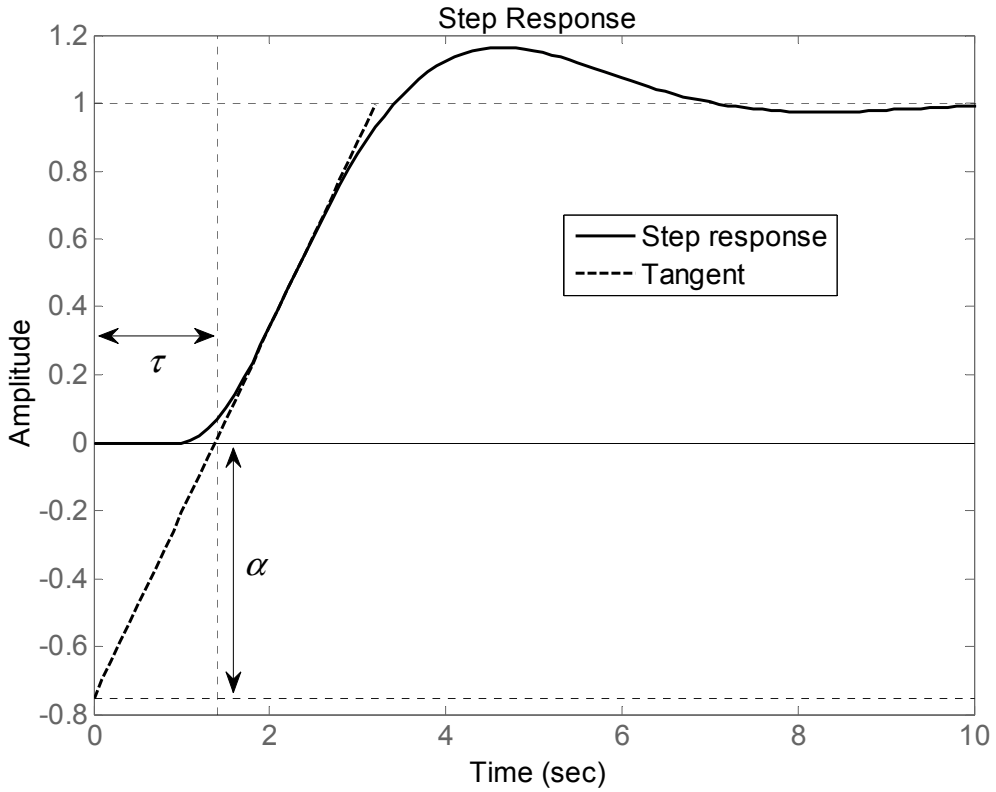

Figure 9. Determining the process parameters in the Ziegler-Nichols step response method.

The frequency response method is also based on describing the process with two parameters that are the ultimate gain, K_u, and the ultimate period, T_u. For determining these parameters, the plant is controlled with a P-controller, and its gain is increased until the system oscillates critically. The gain yielding marginal stability is the ultimate gain, and the ultimate period is the period of oscillation at this gain. The parameters for the P, PI and PID controllers obtained from the Ziegler-Nichols frequency response method are shown in Table 2 [102].

3.2.2. Cohen-Coon Method

The Cohen-Coon method's [10] main objective is load disturbance rejection. The method is based on the FOTD model

$$G(s) = \frac{K_p}{1+sT} e^{-s\tau}, \tag{39}$$

where K_p is the static gain of the process, T the time-constant and τ the delay. The model (39) needs to characterize the process dynamics adequately, so the process should be stable for the method to be applicable. The method attempts to position the dominant poles so that a quarter amplitude decay ratio is achieved. In the case of a PI or PID controller the integral gain $k_i = K / T_i$ is maximized, which corresponds to the minimization of the integral of error due to a unit step load disturbance [104]. Cohen and Coon derived the controller parameters based on the FOTD process model and analytical and numerical computations. The controller parameters are given in Table 3, where $\beta = K_p\tau / T$ is gain and $\kappa = \tau / (\tau + T)$ is the relative dead time.

3.2.3. IMC Tuning of PID Controllers

Internal Model Control (IMC) is a model-based control method. The IMC method can also be used as a tuning method for the PID controller. Generally, the method is applicable for systems with constant delays, but in [P8], the IMC method is also applied for varying time-delay systems.

Figure 10 represents the IMC principle. In the diagram, the process G is controlled with an IMC controller Q, $\tilde{G}$ is the process model, and disturbances are denoted with w_y. The model output error $y - \tilde{y}$ is subtracted from the reference signal and fed into the IMC controller which calculates the control signal.

Table 3. PID controller parameters according to Cohen-Coon method [104].

Controller	K	T_i	T_d
P	$\frac{1}{\beta}\left(1+\frac{0.35\kappa}{1-\kappa}\right)$		
PI	$\frac{0.9}{\beta}\left(1+\frac{0.092\kappa}{1-\kappa}\right)$	$\frac{3.3-3.0\kappa}{1+1.2\kappa}\tau$	
PD	$\frac{1.24}{\beta}\left(1+\frac{0.13\kappa}{1-\kappa}\right)$		$\frac{0.27-0.36\kappa}{1-0.87\kappa}\tau$
PID	$\frac{1.35}{\beta}\left(1+\frac{0.18\kappa}{1-\kappa}\right)$	$\frac{2.5-2.0\kappa}{1-0.39\kappa}\tau$	$\frac{0.37-0.37\kappa}{1-0.81\kappa}\tau$

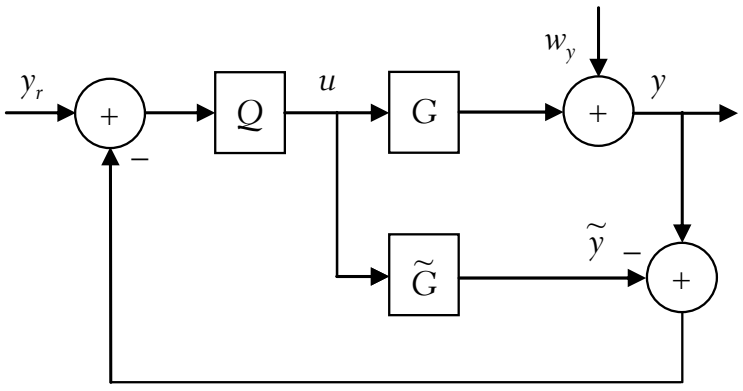

Figure 10. The IMC structure (modified from [19]).

A CT IMC controller $Q(s)$ is calculated so that the process model is first divided into two parts

$$\tilde{G}(s) = \tilde{G}_{+}(s)\tilde{G}_{-}(s), \tag{40}$$

where $\tilde{G}_{+}(s)$ is the non-invertible part of the model including all unstable zeros and delays. The rest of the model is included in $\tilde{G}_{-}(s)$. The controller is given as

$$Q(s) = \tilde{G}_{-}^{-1}(s)R(s), \tag{41}$$

where R is a low-pass filter transfer function of order n given as

$$R(s) = \frac{1}{(1+s\lambda_{IMC})^{n}}. \tag{42}$$

The low-pass filter is required in order to have a causal controller and λ_{IMC} is the tuning parameter of the IMC method. The value of λ_{IMC} has a significant effect on the performance and robustness of the controlled system. There is a trade-off; a very fast and simultaneously very robust tuning is generally difficult to achieve. Especially in varying time-delay systems, where robustness with respect to delay variance plays an important role, the tuning of λ_{IMC} turns out to be crucial. The dependency between the jitter margin (26), the control system performance and the λ_{IMC} parameter is further discussed in [P8] in the case where IMC control is used in the NCS setup.

When implementing the IMC controller, it is useful to recognize the dependency between the IMC controller (Q in Figure 10) and the controller in the classical feedback loop (C in Figure 8). The IMC control law in the classical control loop becomes

$$C(s) = \frac{Q(s)}{1-\tilde{G}(s)Q(s)}. \tag{43}$$

If the controller (43) is used, the process delays must be approximated with linear transfer functions in order to be able to calculate the controller. A constant delay of τ seconds corresponds to an exponential function $e^{-\tau s}$ in the Laplace domain, and the delay can be approximated with the Taylor series expansion or the first order Padé approximation

$$e^{-\tau s} \approx \left(1-s\frac{\tau}{2}\right) \Big/ \left(1+s\frac{\tau}{2}\right). \tag{44}$$

Although the IMC design often yields high order controllers, under certain assumptions it is possible to obtain the PID structure from the IMC design and thus get the tuning parameters for a regular PID controller. Consider the FOTD process model (39). Using the IMC design and the first order Taylor series expansion $e^{-s\tau} \approx 1-s\tau$ with n = 1 (order of the low-pass filter), the controller C becomes [102]

$$C_{PI}(s)=\frac{1+sT}{K_p s(\lambda_{IMC}+\tau)}=\frac{T}{K_p(\lambda_{IMC}+\tau)}\left(1+\frac{1}{sT}\right). \tag{45}$$

This has the PI controller structure with parameters

$$K=\frac{T}{K_p(\lambda_{IMC}+\tau)},\ T_i=T\ \text{ or }\ k_p=\frac{T}{K_p(\lambda_{IMC}+\tau)},\ k_i=\frac{1}{K_p(\lambda_{IMC}+\tau)}. \tag{46}$$

If the Padé approximation (44) of the delay is used, the controller C becomes

$$C_{PID}(s)=\frac{(1+sT)\left(1+s\frac{\tau}{2}\right)}{K_p s\left(s\frac{\lambda_{IMC}\tau}{2}+\lambda_{IMC}+\tau\right)}\approx\frac{(1+sT)\left(1+s\frac{\tau}{2}\right)}{K_p s(\lambda_{IMC}+\tau)}, \tag{47}$$

which is a PID controller. This form of PID controller is actually the 'interacting' controller or the 'analog algorithm' [63]. Comparing (45) and (47) reveals that

$$\begin{aligned} C_{PID}(s)&=\underbrace{\frac{T}{K_p(\lambda_{IMC}+\tau)}\left(1+\frac{1}{sT}\right)}_{C_{PI}(s)}\left(1+s\frac{\tau}{2}\right)\\ &=\underbrace{\frac{T}{K_p(\lambda_{IMC}+\tau)}\left(1+\frac{\tau}{2T}\right)}_{k_p}+\underbrace{\frac{1}{K_p(\lambda_{IMC}+\tau)}}_{k_i}\frac{1}{s}+\underbrace{\frac{T\tau}{2K_p(\lambda_{IMC}+\tau)}}_{k_d}s. \end{aligned} \tag{48}$$

This is a PID controller of the same structure as (29). In the classical form of the PID controller (28) the parameters are given as

$$K=\frac{T}{K_p(\lambda_{IMC}+\tau)}\left(1+\frac{\tau}{2T}\right),\ T_i=T+\frac{\tau}{2},\ T_d=\frac{T\tau}{2T+\tau}. \tag{49}$$

The IMC tuning method for PID can be used with even higher order processes, but the first step is then to characterize the process with the FOTD model (39), after which the PID controller parameters are obtained similarly as above [102].

3.2.4. Lambda Tuning

Lambda tuning has been a popular tuning method in the process industry, especially in the pulp and paper industries [64]. The Lambda tuning method is a special case of pole placement design and it was originally proposed by Dahlin in 1968 [11]. The tuning procedure starts with modeling the plant with the FOTD model and approximating the delay with Taylor series expansion or Padé approximation. The desired closed-loop system is defined as

$$G_c(s) = \frac{1}{1+s\lambda' T} e^{-s\tau} = \frac{1}{1+s\lambda} e^{-s\tau}, \tag{50}$$

where λ is a tuning parameter representing the speed of the closed-loop system. On the basis of an open-loop model and the desired closed-loop model, the controller can be solved

$$C_\lambda(s) = \frac{1+sT}{K_p\left(1+s\lambda T - e^{-s\tau}\right)}. \tag{51}$$

It can be seen that the controller has integral action, since $C_\lambda(0) = \infty$. In order to derive the tuning rules, first consider the PI controller

$$C_{PI}(s) = K\frac{1+sT_i}{sT_i}. \tag{52}$$

If the integration time T_i is chosen equal to the process time-constant T, the controller will cancel the process pole. Using the Taylor series expansion for the delay the loop transfer function becomes

$$G_l(s) = C_{PI}(s)G(s) = K\frac{1+sT}{sT}\frac{K_p}{1+sT}e^{-s\tau} \approx \frac{K_pK(1-s\tau)}{sT}. \tag{53}$$

The characteristic equation of the closed-loop system

$$s\left(T - K_pK\tau\right) + K_pK = 0 \tag{54}$$

gives a pole $s = -K_pK/\left(T - K_pK\tau\right)$, which should be equal to the pole of the desired closed-loop system in (50), $s = -1/\lambda$. This gives the following simple tuning rules for PI control [102], [104]

$$K = \frac{1}{K_p}\frac{T}{\tau+\lambda}, \quad T_i = T. \tag{55}$$

Similarly, the PID controller tuning rules can be derived using the Padé approximation for the delay. The tuning rules become [104]

$$K = \frac{1}{K_p}\frac{T+\frac{\tau}{2}}{\lambda+\frac{\tau}{2}},\ T_i = T+\frac{\tau}{2},\ T_d = \frac{T\tau}{2T+\tau}. \tag{56}$$

3.2.5. AMIGO Tuning Rules

The AMIGO (Approximate M-constrained Integral Gain Optimization) tuning rules [103], [104] proposed by Åström and Hägglund represent the recently developed tuning rules for the classical PID controller. These state-of-the-art tuning rules are derived using the step response test approach

for a batch of different process models, stable and marginally stable, in the spirit of the work done by Ziegler and Nichols. All the stable processes in the test batch are first characterized with the simple FOTD model. The use of techniques like robust loop shaping and thorough analysis of robustness, performance and closed-loop system properties described in [103] and [104] finally give the following tuning rules for stable processes.

$$K = \frac{1}{K_p}\left(0.2 + 0.45\frac{T}{\tau}\right),\ T_i = \frac{0.8T + 0.4\tau}{0.1T + \tau}\tau,\ T_d = \frac{0.5T\tau}{T + 0.3\tau} \tag{57}$$

Åström and Hägglund point out that the AMIGO tuning might be conservative especially for lag-dominated ($T >> \tau$) processes. The reason for this is that the largest time-constant of the plant usually maps into the time-constant of the FOTD model used to characterize the plant, but the smaller time-constants may be included in the delay parameter of the FOTD model. Thus the delay of the FOTD model may become unnecessarily large. Presumably, better modeling of the process and tuning rules based on a more complicated process model could improve the performance, but this would make the modeling and control design procedure less simple. The use of the step experiment as the basis of modeling restricts the complexity of the process model to that of FOTD. Higher order models would hence require more advanced identification procedures.

Besides the controller gains, AMIGO tuning defines values for the set-point weights and the measurement filter time-constant. The following control algorithm, differing slightly from (28) and (29), is used.

$$u(t) = k_p\left(by_r(t) - y_f(t)\right) + k_i\int_0^t \left(y_r(\alpha) - y_f(\alpha)\right)d\alpha + k_d\left(c\frac{dy_r(t)}{dt} - \frac{dy_f(t)}{dt}\right) \tag{58}$$

Parameters b and c are the set-point weights and y_f the measured and filtered process variable. The filtering in the previous algorithm affects all terms in the controller, on the contrary to use of the filter only in the derivative term as in (34). The Laplace transform of $y_f(t)$ is $Y_f(s) = G_f(s)Y(s)$, where

$$G_f(s) = \frac{1}{\left(1 + sT_f\right)^n} \tag{59}$$

is the measurement filter with a time-constant T_f. The order of the filter n is typically one or two. The tuning rules for set-point weights and the filter time-constant are

$$\kappa := \frac{\tau}{\tau + T},\ b = \begin{cases} 0, & \kappa \le 0.5 \\ 1, & \kappa > 0.5 \end{cases},\ c = 0,\ T_f = \begin{cases} \dfrac{0.05}{\omega_{gc}}, & \kappa \le 0.2 \\ 0.1\tau, & \kappa > 0.2 \end{cases}, \tag{60}$$

where ω_{gc} is the gain crossover frequency. It is mentioned in [103] that the above choice of T_f reduces the phase margin of the system by 0.1 rad. Since the phase margin and the jitter margin (see (26)) are intuitively coupled, there is a good motivation for investigating the relationship of T_f and the jitter margin, when considering the use of AMIGO rules for NCS. It is shown in [P7] that T_f can be tuned based on the jitter margin requirement, which leads to tuning rules, where the desired jitter margin can be given as an input parameter to the rules.

In AMIGO tuning as well as in the publications of this thesis, where controller (58) is used, the set-point weight $c = 0$. This is due to the fact that in the CT case the derivative of a step-like reference signal is infinite at the time the step is applied. Unit steps are frequently used in the simulations as the reference and thus the set-point weight c needs to be zero. In the case $\kappa \leq 0.2$ the filter time-constant is calculated based on the gain crossover frequency, which is the lowest frequency where the magnitude of the open-loop system becomes less than one.

The AMIGO tuning also considers integrating processes, and the tuning rules for them are presented separately. The process model used is the integrator plus delay (IPD)

$$G(s) = \frac{K_v}{s} e^{-s\tau}, \tag{61}$$

where K_v is the velocity gain. The tuning rules for the integrating processes are

$$K = \frac{0.45}{K_v},\ T_i = 8\tau,\ T_d = 0.5\tau. \tag{62}$$

3.2.6. Robustness Perspective on Controller Tuning

Robustness is an important quality of control systems and it characterizes how modeling errors, disturbances and noise or other variations affect the performance. The control system is robust, if it provides similar performance when exposed to variations than it would under nominal circumstances. Robustness analysis of linear systems is traditionally performed by using the sensitivity functions. In the following, the sensitivity and complementary sensitivity functions are defined and their characteristics discussed. A more detailed description of robustness of control systems can be found in textbooks, such as [13].

Consider Figure 11, where the control system is exposed to input (w_u) and output (w_y) disturbances and measurement noise (w_z). The output of the system may be written as

$$y = \frac{GC}{1+GC}(y_r - w_z) + \frac{G}{1+GC} w_u + \frac{1}{1+GC} w_y. \tag{63}$$

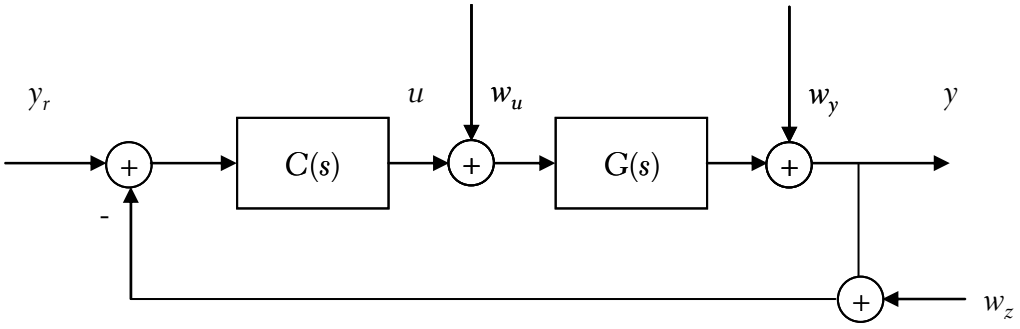

Figure 11. Control system with input (w_u) and output (w_y) disturbances and measurement noise (w_z).

Sensitivity Functions

Let the loop transfer function be $H(s) = G(s)C(s)$. Then the sensitivity function $G_S(s)$ and the complementary sensitivity function $G_T(s)$ are defined as

$$G_S(s) = \frac{1}{1+H(s)}, \quad G_T(s) = \frac{H(s)}{1+H(s)}. \tag{64}$$

Note that $G_S(s) + G_T(s) = 1$ for all frequencies. Using the sensitivity functions (64), the output of the closed-loop system in Figure 11 may be written in the form

$$y = G_T\left(y_r - w_z\right) + G_S\left(Gw_u + w_y\right). \tag{65}$$

It can be seen from (65) that in order to have good reference tracking performance, G_S should be zero for all frequencies to prevent the disturbances affecting the output y. Since $G_S(s) + G_T(s) = 1$, the complementary sensitivity function would be one in this case. If measurement noise was not present ($w_z = 0$), this combination would yield perfect control ($y = y_r$). In reality, though, measurement noise is always present in some form. Typically, measurement noise occurs at relatively high frequencies, which means that G_T can be made large at low frequencies, but it has to decrease at higher frequencies to dampen the effect of noise. This has the effect that G_S has to grow at higher frequencies and hence the disturbances w_u and w_y will have a stronger effect there. Nevertheless, these disturbances typically occur at low frequencies, where G_S may still be small.

Now consider the control signal in the system:

$$u = -G_T w_u + \frac{G_T}{G}\left(y_r - w_z - w_y\right). \tag{66}$$

In general, the control signal should be kept as small as possible to avoid excessive use of energy. This objective may be attained according to (66) by designing such a controller that makes G_T small for all frequencies. To conclude, the control design and tuning procedure of any linear controller should yield the sensitivity function G_S (based on (65)) and complementary sensitivity function G_T (based on (66)) that have small values for all frequencies to decrease the effect of disturbances and noise on the output and control signals. Nevertheless, both of the sensitivity

functions cannot be kept very small simultaneously, since the sum of their values at any frequency is, by the definition, equal to one.

Sensitivity to Parameter Variations

Above, the sensitivity functions were considered from the disturbance and noise compensation perspective. Equally important is to study the effects of process parameter variations on the performance. Referring to Figure 11, the closed-loop system transfer function G_{cl} may be written as

$$G_{cl} = \frac{y}{y_r} = \frac{H}{1+H} = \frac{GC}{1+GC}. \tag{67}$$

The sensitivity of the closed-loop system to changes in G is obtained by derivation:

$$\frac{dG_{cl}}{dG} = \frac{C}{(1+GC)^2} = \frac{G_{cl}}{G} \cdot \frac{1}{1+GC}. \tag{68}$$

Hence the relative sensitivity of the closed-loop system with respect to G may be written in the form

$$\frac{dG_{cl}}{G_{cl}} = \frac{dG}{G} \cdot \frac{1}{1+GC} = G_S \frac{dG}{G}. \tag{69}$$

It is seen that if the sensitivity function G_S is small, the changes in G have a relatively minor effect on the closed-loop system.

Robust Stability

Control design is often based on a simplified model of the true process. This is also a relevant assumption in this thesis as many of the developed methods are based on a simple model. Obviously, this assumption may result in relatively large modeling errors. Hence it is important to evaluate how much error in the model is allowed while still guaranteeing stability. This can be done by using the robust stability condition discussed below.

Consider a model $\widetilde{G}$ of the true process G and assume that the unmodeled dynamics may be described by additive perturbations $\Delta\widetilde{G}$ so that

$$G = \widetilde{G} + \Delta\widetilde{G}. \tag{70}$$

Note that the control design is based on the model $\widetilde{G}$ rather than the process G, and hence the loop transfer function is not $\widetilde{H} = \widetilde{G}C$, but $H = \widetilde{G}C + \Delta\widetilde{G}C$. According to the Nyquist stability criterion, to have a stable closed-loop system, the loop transfer function should encircle (counter-clockwise) the critical point (-1,0) as many times as the loop transfer function has unstable poles. Here the process perturbations are assumed to be stable, so that there are no additional right-half-

plane poles that would require additional encirclements of the critical point. The distance between the loop transfer function $\widetilde{H}$ and the critical point is $\left|1+\widetilde{H}\right|$. Even with the perturbations, stability is maintained, if

$$\left|C\Delta\widetilde{G}\right| < \left|1+\widetilde{H}\right|, \tag{71}$$

which implies

$$\left|\Delta\widetilde{G}\right| < \left|\frac{1+\widetilde{G}C}{C}\right| \Leftrightarrow \left|\frac{\Delta\widetilde{G}}{\widetilde{G}}\right| < \frac{1}{\left|\tilde{G}_T\right|}. \tag{72}$$

Since (72) must hold for all frequencies on the Nyquist curve, the condition for robust stability may be written in the form

$$\left|\frac{\Delta\widetilde{G}(j\omega)}{\widetilde{G}(j\omega)}\right| = \left|\frac{G(j\omega)-\widetilde{G}(j\omega)}{\widetilde{G}(j\omega)}\right| < \frac{1}{\left|\tilde{G}_T(j\omega)\right|}. \tag{73}$$

The condition of robust stability determines how much the process model may deviate from the true process. Large variations are allowed in the range where the complementary sensitivity function is small and only small variations are allowed in the range where $\tilde{G}_T$ is large. Note that a conservative estimate of permissible process variations that will not lead to instability is obtained by

$$\left|\frac{\Delta\widetilde{G}(j\omega)}{\widetilde{G}(j\omega)}\right| < \frac{1}{M_t}, \qquad M_t = \sup_{\omega}\left|\tilde{G}_T(j\omega)\right| = \left\|\frac{\widetilde{G}C}{1+\widetilde{G}C}\right\|_{\infty}. \tag{74}$$

M-circle Criterion for Controller Tuning

As seen above, both the sensitivity and complementary sensitivity functions should be small to achieve robustly stable controller tuning so that disturbances, noise and modeling errors would not affect the output or control signal. Since both functions cannot simultaneously be zero, it is still possible to aim at bounding the values for all frequencies. The AMIGO tuning (see Section 3.2.5) takes this approach and uses the so called M-circle criterion in evaluating the robustness against modeling errors, disturbances and noise. M-circle is a conservative criterion that bounds the maximum values of the sensitivity and complementary sensitivity functions. The following result of robustness defines the M-circle. If the Nyquist curve of the loop transfer function does not intersect a circle with center c_R and radius r_R defined as

$$c_R = -\frac{2M^2-2M+1}{2M(M-1)},\ r_R = \frac{2M-1}{2M(M-1)}, \tag{75}$$

then the sensitivity function G_S and the complementary sensitivity function G_T are less than M for all frequencies [105]. Hence, robustness is captured by one parameter only, M. The value M = 1.4

was used in the AMIGO rule development, although finally the rules did not quite satisfy the constraint. For the process test batch a 15 % increase of M was reported ($M \approx 1.6$).

The M-circle robustness criterion was also employed in [P5], [P6] and [P7], where PID controller tuning rules were developed for stable and marginally stable integrating processes. The robustness criterion is illustrated in Figure 12, where an M-circle with $M = 1.5$ is drawn (the value used in [P5] and [P6]) together with various open-loop Nyquist curves.

The open-loop Nyquist curves in Figure 12 are obtained by appending the process model (IPD + first order lag) with optimized PID controllers for a range of process parameters (see [P5] for details). It can be seen that the Nyquist curves at most touch the robustness circle, but they do not intersect it. Thus the sensitivity and complementary sensitivity functions of the systems are less than 1.5 for all frequencies, and controller tuning is robust with respect to the M-circle criterion.

3.3. Complementary Control Techniques for PID in NCS

The power of the PID controller can be strengthened by using other control techniques that support systematic adaption of the controller parameters based on a certain scheduling variable, which often in the NCS case is the delay. The techniques used in [P8], namely delay-adaptive IMC control and fuzzy gain scheduling, are briefly presented here. Although in [P8], the delay-adaptive IMC controller is used as such, the same technique could be used as a PID tuning technique and tuning rules, such as (49), could be obtained.

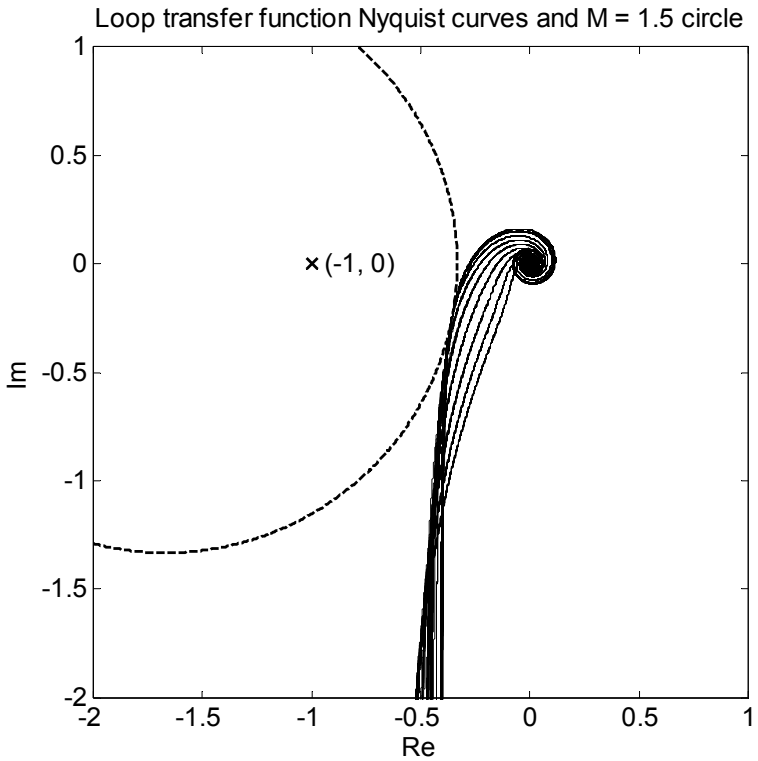

Figure 12. Open-loop Nyquist curves and $M = 1.5$-circle [P5].

3.3.1. The Delay-Adaptive IMC Controller

If the delay is known, as can often be assumed if it is constant, the Padé approximation (44) of the delay can be used to design an IMC controller. If the delay varies, there are two possibilities: 1) if the exact delay of each measurement is known, one can recompute the controller at each sample time (DT controller) and update the delay parameter in the equations accordingly, or 2) if the delay distribution is known rather than the exact delay times, the expectation value of the delay can be used in the Padé approximation. In this case, the tuning of the λ_{IMC} parameter in the IMC controller has to be done so that the varying time-delays are tolerated (e.g. by guaranteeing the closed-loop system an adequate jitter margin, as is done in [P8]).

In order to illustrate the two alternatives discussed above, an IMC controller is derived here for an example process. Consider the FOTD process model (39) with $K_p = 1$ and a delay of τ seconds. Using (40) - (43), the IMC controller can be calculated and changed into the form of the controller in the classical feedback loop (see Figure 8). If the delay approximation in (44) is used and the filter order is fixed to $n = 1$, the controller becomes

$$C(s) = \frac{\frac{T\tau}{2}s^2 + \left(\frac{\tau}{2} + T\right)s + 1}{\frac{\tau\lambda_{IMC}}{2}s^2 + (\tau + \lambda_{IMC})s}. \tag{76}$$

This is a continuous-time IMC controller. Its discrete-time equivalent can be derived using, for example, the Tustin approximation. In the time domain, the control signal of the DT IMC controller is calculated as

$$\begin{aligned} u(kh) = {} & \frac{2\lambda_{IMC}\tau}{(h+\lambda_{IMC})\tau + h\lambda_{IMC}}u(kh-h) + \frac{(h-\lambda_{IMC})\tau + h\lambda_{IMC}}{(h+\lambda_{IMC})\tau + h\lambda_{IMC}}u(kh-2h) \\ & + \frac{\left(T+\frac{h}{2}\right)\tau + hT + \frac{h^2}{2}}{(h+\lambda_{IMC})\tau + h\lambda_{IMC}}e(kh) + \frac{h^2 - 2T\tau}{(h+\lambda_{IMC})\tau + h\lambda_{IMC}}e(kh-h) + \frac{\left(T-\frac{h}{2}\right)\tau - hT + \frac{h^2}{2}}{(h+\lambda_{IMC})\tau + h\lambda_{IMC}}e(kh-2h). \end{aligned} \tag{77}$$

The controller (77) depends on τ that can either be updated as new measurements arrive, or an expectation value of the delay can be used giving a fixed control law. The performance of these approaches is discussed in [P8].

3.3.2. Fuzzy Gain Scheduling

Fuzzy logic can be used to update the PID controller parameters online. In addition, fuzzy controller may be used as the primary controller in NCS applications so that the delays and packet losses are compensated by the fuzzy logic. A self-tuning fuzzy controller for NCS has been proposed in [80]. This controller has a two-level architecture, where the gains of the first-level fuzzy controller, which has the structure of a PID controller, are updated by the second-level fuzzy

controller. In [P8], a fuzzy gain scheduler is used for controlling a varying time-delay system and its performance is compared with different PID and IMC controllers, including those discussed in Section 3.3.1. The fuzzy gain scheduler determines the weighting of five PID controllers based on the delay measurement. The basics of gain scheduling and fuzzy logic are briefly reviewed below.

Gain Scheduling

The process dynamics may change depending on the operation point and conditions. If the effects of different conditions on process behavior are known, it is possible to adjust the controller parameters by monitoring the process operating conditions. This strategy is called gain scheduling, since it was originally used only for selecting the controller gains. In gain scheduling the controller parameters are changed during control in a predefined way. The technique can enlarge the operation area of linear controllers in nonlinear systems [86]. The strategy is especially useful in compensating the variations in process parameters or known nonlinearities of the process.

In gain scheduling design the scheduling variables are first selected. These have to be measurable or estimated, since they are used for adjusting the controller parameters continuously. Typical scheduling parameters are the delays and time-constants of the process. Next, the controller parameters need to be calculated for different operating conditions using a suitable control design method. The stability of the control system is often investigated with simulations. Especially the transient behavior between different operating conditions must be examined in detail [106].

Gain scheduling for networked PI control over an IP network has been investigated in [83], where a gain scheduling method that uses real-time IP traffic conditions as the scheduling variable is proposed and discussed. Fuzzy logic can be used for estimating the state of the controlled process and for deciding on the gain(s) as seen in [P8].

Fuzzy Logic

In a broad sense, fuzzy logic refers to all theories and technologies that employ fuzzy sets, which are classes with noncrisp boundaries. The core of fuzzy logic systems constitutes of four basic elements:

- Fuzzy sets
- Linguistic variables
- Possibility distributions
- Fuzzy if-then rules

Fuzzy sets are sets with smooth boundaries and they allow partial membership. The values of the linguistic variables are both quantitatively and qualitatively described by a fuzzy set. If a variable belongs to a certain fuzzy set, the name of the fuzzy set gives the qualitative value of the variable, whereas the corresponding membership function describes the value quantitatively. Possibility distributions are constraints on the value of a linguistic variable imposed by assigning it a fuzzy set.

Thus the possibility distribution describes the possibility that a certain value of the linguistic variable belongs to a fuzzy set. Fuzzy if-then rules form a knowledge representation scheme for describing a functional mapping or a logic formula that generalizes an implication in two-valued logic. Mathematically, the fuzzy if-then rules can be viewed as an interpolation scheme, since they enable fusion of several fuzzy rules when their conditions are all satisfied to a degree [92].

During the last decades, fuzzy logic control has been successfully applied to industrial systems, which has also promoted the acceptance of fuzzy logic theory. Generally speaking, fuzzy logic control tries to combine the specifications of the system performance with its measurements and observations into a rule-based linguistic control strategy. In [92], the architecture of a generic fuzzy control system is discussed. The inputs of fuzzy logic control are the error signal and its derivative. The fuzzy logic controller calculates the increment on the previous control signal and thus implements integral action. This architecture resembles the PID control strategy, but the nonlinearities of the process can be taken into account in fuzzy logic and the operability range of PID control may be extended.

3.4. PID Control in NCS

This section briefly discusses the properties of the PID algorithm in control of NCS and points out some modifications that have been proposed to better tackle the network-induced delays and packet losses. When applying the ideal PID algorithm (28) in a control loop, where varying time-delays are present, one can show by using the concept of jitter margin (26) that with a FOTD process the stability of the system cannot be guaranteed, at least in the sense of the jitter margin condition, for any additional delays. In such case the jitter margin is zero. It should be noted, though, that the jitter margin is a conservative criterion and only sufficient, so the zero jitter margin does not necessarily imply instability for additional delays. The reason for having zero jitter margin is that the complementary sensitivity function of the system does not have roll-off at high frequencies and in the frequency response plot, the closed-loop magnitude crosses the bounding curve that the jitter margin defines, see Figure 13. This can also be shown analytically, see [P7]. The problem can be easily solved by using a measurement filter, such as (34) or (59), which will saturate the gain of the controller derivative term at high frequencies.

Figure 13 shows the advantage of using measurement filters in the PID controller from the jitter margin point of view. It is seen in the figure that without the filter the complementary sensitivity function of a PID controlled FOTD process hits the jitter margin bound and obviously the jitter margin is zero. When filtering is applied the jitter margin is positive, because the complementary sensitivity function remains below the jitter margin bound for all frequencies. Nevertheless, if the controller is not well tuned, the lines may intersect and the required jitter margin is not achieved.

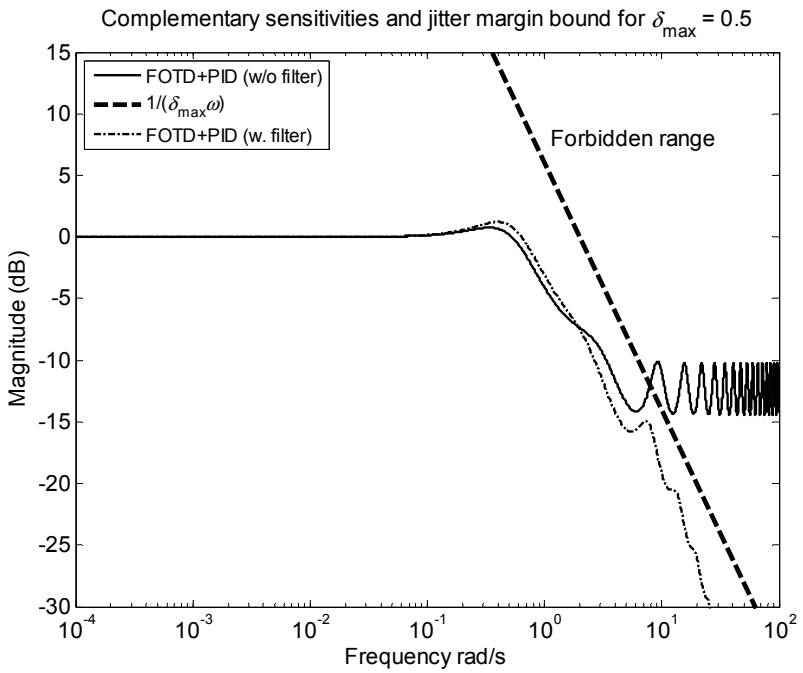

Figure 13. Complementary sensitivity functions of PID controlled FOTD process, with and without the measurement filter, and the jitter margin bound for δ_{max} = 0.5 [P7].

Packet losses cause undesired behavior in the signals of a PID-controlled system, if the losses are not considered in the controller design procedure. Under packet losses in communications, the D-term of the PID controller is likely to cause spikes in the control signal [76]. This would happen when the communications are reestablished after a period of disconnectivity. The amplitude of the spike is determined based on how much the measured value of the process variable has evolved between the times of successful reception of packets (before and after the packet losses). It is pointed out in [76] that during the time of packet losses the control signal is a linearly changing function of error, since there is no new information available of the process variable. Thus the error is constant and the I-term integrates the error over time. The set-point and measurement signals are static functions of error, and as a result, a linearly increasing or decreasing control signal is obtained. This can endanger the stability of the process or cause undesired actions. It should also be noted that during packet losses there will be no information regarding actuator saturation and the possible anti-windup functions may fail. In addition, on controller output losses the actuator may experience bumps in the control signal.

In [76], an enhanced PID algorithm for wireless control systems is proposed to overcome the problems of discontinuous communications. The integral and derivative parts of the controller are remodeled such that they are updated only on the arrival of new packets. The integrator is replaced by a filter that receives information about the actuator position at the same time as new measurement packets arrive. The filter output is calculated as

$$O(k) = O(k-1) + \left[u(k-1) - O(k-1)\right] \cdot \left(1 - e^{-\frac{\Delta T}{T_{Reset}}}\right), \tag{78}$$

where $O(k)$ is the new filter output, $O(k-1)$ its previous value, $u(k-1)$ the controller output for the last execution (based on the actuator position feedback), ΔT the elapsed time since a new value was communicated, and T_{Reset} a tuning parameter (integrator resetting time). The last communicated actuator position is used in the filter output calculation and hence this structure should be able to automatically compensate for any loss in the output of the controller.

The derivative part of the proposed algorithm is basically an event-based version of the regular D-term, and it is given as

$$d(k) = k_d \frac{e(k) - e(k-1)}{\Delta T}, \tag{79}$$

where $d(k)$ is the controller derivative term, k_d the gain, and $e(k) - e(k-1)$ the difference of current and previous error. The difference with respect to the normal PID algorithm is that the sample time is updated based on reception of measurement packets and thus the derivative action is smaller the longer the time interval between the two last packets. Note that the control law is updated only upon receiving new information.

Event-based PID control has also been considered in [101], but from a different perspective. The basic idea of event-based control is to transmit sensor and command data only when needed. Especially in bandwidth limited systems, such as networked control systems, event-based control is desirable, because the communication medium is only used if something has happened since the last event [26]. Although the main motivation for developing the event-based PID controller in [101] arises from the field of embedded control systems and the problems faced with scheduling and control co-design, a similar controller structure is very usable in networked control systems (or even networked embedded control systems). In event-based control the execution of the control algorithm is not time-based, but the control law is updated upon certain variables exceeding predefined thresholds, that is, when events occur. Different criteria for event detection may be defined based on the process and the scenario. An event could occur, for example, if a measured variable would exceed a certain value, or relative changes of variables could be tracked. Sampling should also be performed at reference changes. The tracking of error signal rather than the measurement signal might be more effective, because on the sudden change of the reference signal the error changes immediately, whereas the measured variable reacts much slower. An event could be triggered if the absolute value of current error would deviate from the error at the previous sample more than a predefined threshold value. It might also be useful to set a maximum sample time after which sampling would occur even though none of the triggering events were activated. Event-based control may be very efficient from the CPU utilization and networking points of view, since presumably the control law is updated more rarely than in the case of time-based control. The drawback of event-based control, however, is that the control system becomes more difficult to analyze [101].

In the literature, the structure of the PID controller is sometimes discussed in the light of NCS and modifications such as presented above have been proposed to the algorithm. The tuning of the PID controller for NCS has not been comprehensively dealt with, even though it has a significant impact on robustness properties especially with respect to varying time-delays and packet loss. The discrete-time PID controller tuning problem is discussed in [70] in systems with random delays. The NCS is assumed to be fully-distributed and the tuning is based on simulation based optimization of the controller parameters for a specific process model. In [41], a dual PID structure is proposed to overcome the effects of load disturbances and to improve the dynamic performance and disturbance rejection in NCS. In addition, a relay auto-tuning method is applied for obtaining the process parameters, which are then used for determining the controller gains. The framework is based on certain simplifications of the NCS model, for example, the network delays (τ_k^{sc} and τ_k^{ca}) are lumped together as a single delay.

In [17], fuzzy logic is used to set the integral and derivative terms of the PID controller. For the proportional term, the paper suggests using a fuzzy immune algorithm that is based on the ideas of the biological immune system. The proportional term is modified so that the adaptive gain k_{p1} is calculated by

$$k_{p1} = K\left[1-\eta f\left(u(k),\Delta u(k)\right)\right], \tag{80}$$

where K is gain, η is an adaption speed parameter, f is a selected nonlinear function, which is approximated by a simple fuzzy logic scheme, and $\Delta u(k)$ is the change of control signal u at k. The PID controller algorithm is implemented in the incremental form.

Genetic algorithms (GA, see e.g. [12]) have been proposed to tune the PID controller, also in the NCS setup. The GA in [40] updates the given initial PID gains based on evaluating a fitness function that is a weighted sum of several performance criteria, including settling time, overshoot and normalized integral of square error (ISE) cost criterion (see Section 3.5.1). The experiments reported show that the delay in the Profibus-DP fieldbus setup that is used in the study varies between one and four sample times. The performance of a modified Ziegler-Nichols tuning and the proposed GA based PID tuning method is compared in the Profibus-DP fieldbus testbed. This GA based tuning procedure resembles those proposed in [P2] and [P4], but the optimization method is different. The idea is the same in the sense that the performance of the control system is evaluated in the presence of varying time-delays and the performance is optimized. If the optimization criteria are carefully chosen, these methods provide good performance and robustness against delays.

3.5. Simulation based Controller Parameter Optimization

In the publications of this thesis, the simulation and optimization based controller tuning technique was applied in several phases. The idea is to model the plant and the network, and to

build a simulation model of the NCS. With the simulation model it is possible to evaluate the control performance and adjust the controller parameters according to an optimization procedure such that the performance is improved. By adjusting the parameters, re-evaluating the performance, and iterating, the optimal controller parameters can be found. The technique is not dependent on the controller type or structure, but it requires that the controller structure is fixed and has adjustable parameters that affect the closed-loop system properties such as performance.

3.5.1. Controller Performance Evaluation

In order to evaluate the performance of a closed-loop control system, a cost criterion must be set. The integral of error functions are often used as such measures. The most common ones are ITAE (Integral of Time-Weighted Absolute Error), IAE (Integral of Absolute Error), ISE (Integral of Square Error) and ITSE (Integral of Time-Weighted Square Error). They are given, respectively, in (81) - (84), where the error signal is $e(t) = y_r(t) - y(t)$ [102].

$$J_{ITAE} = \int_0^\infty t\,|e(t)|\,dt \tag{81}$$

$$J_{IAE} = \int_0^\infty |e(t)|\,dt \tag{82}$$

$$J_{ISE} = \int_0^\infty (e(t))^2\,dt \tag{83}$$

$$J_{ITSE} = \int_0^\infty t\,(e(t))^2\,dt \tag{84}$$

These cost criteria are well-known and it is easy to modify them to derive more suitable criteria for the control system concerned. To evaluate the performance of NCS different measures that also consider, for example, sensitivity to delays and packet losses should be defined to complement the ones above. One candidate measure for NCS is the jitter margin (26). In addition, it should be noted that the cost criteria above do not depend on the controller output, the control signal, at all. If the control signal use needs to be considered, the IERC criterion in (85) could be used (Integral of Weighted Sum of Square Error and Required Control Signal Error).

$$J_{IERC} = \int_0^\infty \left[w_1 (e(t))^2 + w_2 \left(y_r(t) - K_p \cdot u(t) \right)^2 \right] dt \tag{85}$$

The weights w_1 and w_2, for which $w_1 = \gamma$, $w_2 = 1 - \gamma$ and $0 < \gamma \leq 1$, define how much the error and the control signal use are weighted in the criterion. The larger γ, the less the control signal is weighted and vice versa. The process static gain K_p scales the control and the reference signals onto same level. Thus the difference $y_r(t) - K_p \cdot u(t)$ equals zero, when the control signal is on the level that is required to have the process output on the reference signal level in stationary conditions. This criterion resembles the optimal control cost, but here also non-zero references are allowed

[32]. Note that the cost function (85) values might, for some values of γ, be very sensitive to small changes in the value of γ. In the thesis, the value of γ parameter is always fixed in advance and hence this effect of sensitivity is not considered as a major problem.

3.5.2. Constrained Optimization Methods

There are many methods for unconstrained optimization (e.g. steepest descent or quasi-Newton's methods) that can be used for controller tuning, if the decision variables (controller parameters) are not constrained and there are no other limiting factors affecting the optimization. The different criteria that need to be fulfilled by the optimized controller can be expressed in the optimization problem either as objectives or constraints, which can greatly affect the optimization procedure and the choice of method used for solving the problem. For example, the jitter margin (26) could be considered in the controller parameter optimization as an objective function (as is done in [P5] and [P6]) or it could be formulated into a constraint function (as is done in [P8]). In the former approach, there is no need to set a specific target value for the objective, and the value of the objective function is simply optimized. In the latter approach, the constraint function must have a fixed target value that is known in advance. If the jitter margin has a specific target value, it depends on the application (e.g. the maximum delay of a certain network). Often in practice, there are limitations or requirements in the control system that can also be expressed as constraints to the optimization problem.

A general constrained optimization problem can be expressed as

$$\begin{aligned} &\text{Min } J(x) \\ &\text{s.t. } g(x) = 0 \\ &\quad h(x) \leq 0 \\ &\quad x \in \mathbb{R}^n, \end{aligned} \tag{86}$$

where J is the objective function to be minimized and x is the vector of decision variables. If single-objective optimization is considered, J is a scalar-valued function. In the multi-objective case J is a vector. The feasible set of variables x is determined in (86) by the *equality* and *inequality constraint functions* or $g(x)$ and $h(x)$, respectively [46].

There are different methods for solving unconstrained and constrained optimization problems. One of the advanced methods for solving constrained nonlinear optimization problems is sequential quadratic programming (SQP). SQP has been used in solving the optimization problems in [P1], [P2], [P4] and [P8]. The SQP algorithm consists of three main phases. At each iteration the Hessian matrix of the Lagrangian function is first updated. The Lagrangian is given by

$$L(x, \lambda) = J(x) + \sum_{i=1}^{m_i} \lambda_i g_i(x) + \sum_{j=1}^{m_j} \lambda_j h_j(x), \tag{87}$$

where λ_i and λ_j are the Lagrange multipliers and m_i and m_j are the number of equality and inequality constraints, respectively. Secondly, a quadratic programming subproblem is solved and the solution is used to calculate a new search direction. Finally, the step length and the next iterate are calculated using a proper line search method [48].

3.5.3. The Tuning Procedure

As discussed above, the optimization problem can be given constraints regarding the closed-loop system performance, values of the controller parameters and the worst case delay of a network. Constraints can be formulated such that the robustness against delay variation and disturbances is taken into account. For example, in [P5] and [P6], in the proposed multi-objective optimization based controller tuning method the M-circle (75) is formulated into a robustness constraint as will be explained in more detail in Section 4.4.1. The constraints need to be satisfied in order to successfully terminate the optimization procedure. Figure 14 shows the typical steps in controller parameter optimization when simulation is used for evaluating the controller performance. This approach has been used in [P1], [P2], [P4], [P5], [P6] and [P8] for tuning the controllers.

At the first step, the initial parameters of the controller are chosen. This turns out to be an important step for successful optimization. Here some engineering expertise is required, since it is important to be able to start the optimization with reasonable values. Well chosen initial values might avoid running the optimization into local, non-global minima, which is a risk for non-global optimization methods.

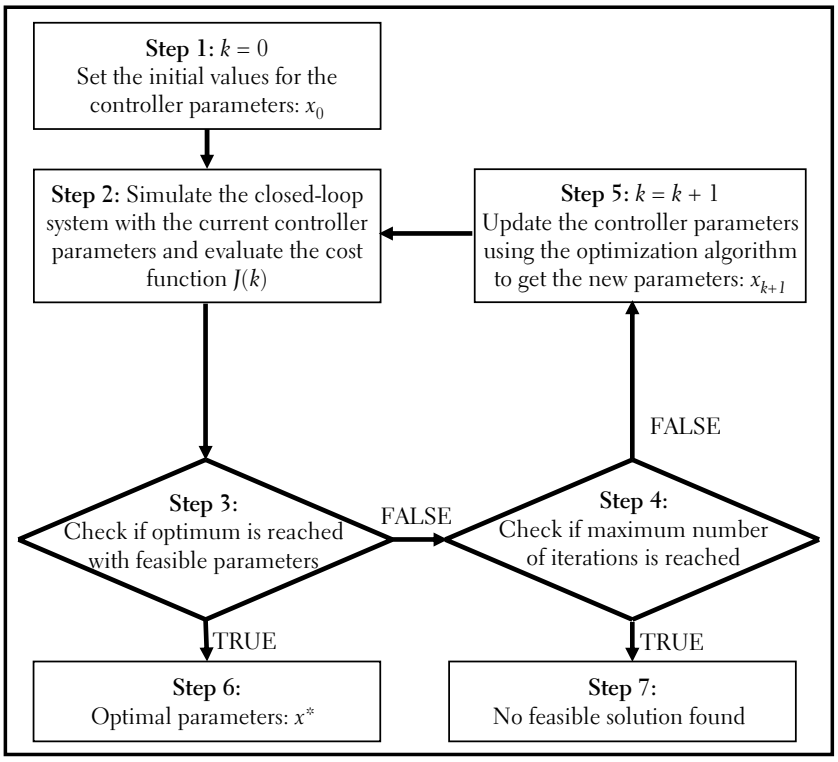

Figure 14. The steps in controller parameter optimization (modified from [P1]).

The PID tuning tool proposed in [P1] is very useful in defining the initial values for the optimization as the response of the control system is illustrated in a scope on the user interface and updated as the tuning parameters are manually changed. Suitable initial parameters that qualitatively provide reasonable performance for the closed-loop system may usually be easily found. The optimization procedure may be started with the values obtained. Alternatively, the tuning may be initiated with values obtained by using some well-known tuning rules, such as the AMIGO rules (57).

After the initialization step, the iterative part of the optimization procedure begins. At each iteration, the closed-loop system is simulated using the available controller parameters and the cost function is evaluated. Typically in the simulations, a step is applied to the reference input and the simulation is run until the time after which the output error remains at zero. At this point the integral cost criteria, for example those in (81) - (84), do not increase further and the cost function value can be evaluated. After that a termination test is performed in order to see if it is necessary to continue the iteration. If the test is not passed (Step 3 in Figure 14), the number of performed iterations is evaluated against the maximum number of iterations (Step 4). If the limit is not exceeded, the optimization algorithm updates the controller parameters based on the cost function value and optimization constraints. The iteration continues by simulating the system with the new controller parameters. If the test in Step 3 is passed, the iteration stops and the optimal parameters are available. The solution must satisfy all the constraints given for the optimization. If the number of iterations increases too high, the optimization procedure is terminated (Step 4) without feasible solution and the user must restart the procedure from different initial values. Depending on the constraints and objectives, it is possible that no feasible solution exists.

4. Summary of Publications

In the publications of the thesis, several new tuning methods and rules for PID controllers in NCS and, more generally, in varying time-delay systems are proposed. This chapter summarizes the scenarios, methods and results of the publications. First, the control design approaches and tools used in the thesis are summarized in Section 4.1. In Section 4.2, the main ideas of each publication are briefly presented. Then the main results of the thesis are summarized by sorting them into three categories that are discrete-time PID controller tuning (Section 4.3), tuning rule development (Section 4.4), and design and tuning tools and platforms for NCS (Section 4.5).

4.1. Control Design Approach and Tools Used in the Thesis

The motivation for researching the PID controller in NCS lies in the simplicity of the implementation, the low computational resource and memory requirements, as well as the intuitive and familiar structure and low mathematical complexity of the controller. These properties are valuable especially in the resource constrained and new application area of wireless automation systems. There is a need to understand the effect of the PID algorithm structure on the robustness against network-induced delays and losses, and how robustness can be improved by tuning the controller parameters. The publications in the thesis approach the PID controller design and tuning problem from various points of view, and the objective of this work is to improve the performance and robustness of the closed-loop control system despite the varying time-delays and packet losses. Besides developing the tuning methods and rules for the PID controller, the thesis also presents a tuning tool that implements one of the tuning methods and an environment for NCS experiments that has been applied to test the performance of the proposed tuning rules and methods. This thesis focuses on the single input single output (SISO) PID controller design and tuning in NCS.

Throughout the thesis it is assumed that the controlled plant is a continuous-time system. This assumption is justified since most of the plants in the process industry are in fact continuous-time systems. On the other hand, the tuning methods proposed in [P2] and [P4] and the comparison in [P8] consider discrete-time controllers, which again is justified since the controllers are most often implemented on computers. This is definitively the case when networked control is considered as the measurements are transmitted over a digital network. Nevertheless, in the tuning rule development ([P5], [P6], [P7]) the controllers are in continuous-time to make the developed rules

more general and independent of the sample time. Furthermore, the analysis in [P5] and [P7] is simplified when assuming that both the plant and the controller are in the same time domain (CT in this case). Based on the discussion in [36], it can be claimed that the error made when assuming continuous-time controllers in the analysis is insignificant, if a relatively small sample time is used in the controller.

The tuning tool in [P1] supports both continuous-time and discrete-time controllers, but the tuning method proposed in [P2] that the tool also implements is only for discrete-time controllers due to the formulation of the constraints in the optimization problem. In [P8], the controllers are all in discrete-time, because one of the compared controllers is an event-based PID controller, which acts only upon receiving a new measurement packet. This has to be implemented in discrete-time and in order to have comparable results all other controllers are also implemented in discrete-time. In the case of discrete-time controllers the plant is sampled at a constant rate, but due to the varying time-delays of the network, the packets are received at the controller irregularly. In the thesis, the controller is usually time-driven except for the comparison in [P8], where also an event-driven version of the PID controller is considered. The actuators are assumed to be time-driven and they implement ZOH.

In most of the cases the methods proposed are based on the assumption of semi-distributed NCS, where the measurements are passed over the network to the controller, but the controller and the actuator are assumed to be located physically together ($\tau_k^{ca} = 0 \ \forall k \in \mathbb{N}$). This is justified especially for considerations regarding wireless NCS, since typically the actuators require a lot of energy and they need to be wired by the power cables. Hence, it might be useful to have the controller (that also needs some energy) located in the same place as the actuator unless there is a specific need to fully distribute the controller, actuator and sensors. The fully-distributed case is considered in [P4], but the rest of the papers consider varying plant output delays or losses only in the sensor-to-controller link. Such a scenario could also be relevant in the wireless automation applications, where the wireless sensor network would be used for collecting the data and passing them to the controller node that would be located in the actuator. The results obtained in this setup are also applicable, if the sensor-to-controller and the controller-to-actuator delays can be lumped together as $\tau_k = \tau\left(\tau_k^{sc}, \tau_k^{ca}\right)$. This assumption is sometimes made in the literature to simplify the analysis (e.g. in [41] and [80]).

The methods developed in the thesis do not generally require that the measurements would have time-stamps or that the network should be synchronized. The control is based on the received measurements from the output of the plant and the data are used by the controller as such. In [P8], some of the controllers do use the timing information to adapt the controller parameters based on the delay, but otherwise the proposed solutions do not require any knowledge of the measurement time or length of delays. This basic assumption has several advantages, since it allows very simple implementation of the methods into existing infrastructures. There is no need to develop any time synchronization algorithms in the control systems, which saves the burden of implementing such

algorithms and even spares some additional hardware. Additionally, the solutions may be applied for a wide range of varying time-delay processes and networked control systems. On the other hand, there has to be some logic to prevent very old measurements arriving at the controller, since most of the methods assume a certain maximum value of the delay. By some network protocols, the maximum delay can be guaranteed, but especially in wireless applications this is extremely challenging. A single measurement that is older than the maximum delay would be seen as noise by the controller and this would hardly destabilize the system, but compromising the delay limit continuously may yield instability.

The architecture of the controller has a significant impact on the control system robustness against delays, disturbances and noise, sensitivity to modeling error and performance. The architecture of the controller refers here to its structure and components, that is, whether there is an estimator before the controller to compensate for the delays and losses, a filter that smoothens the measurement or simply a controller that acts on the raw data. In the thesis, the measurement filtering approach is taken. Besides removing the measurement noise, which is required in order to use the derivative term of the PID controller, the filter has a significant effect on the jitter margin of a control system (see [P7]), and by tuning the filter time-constant it is possible to manipulate the jitter margin. Use of an estimator, such as the Kalman filter, has been left out of the investigations in the thesis due to the desire to have as simple and general methods and tuning rules as possible. The Kalman filter might require a detailed model of the plant in order to be able to sufficiently well estimate the plant states. Recently, it has also been shown in [71] that the Kalman filter would not necessarily improve the performance of a control system suffering from jitter compared to the filtering approach. Further, if WSAN applications are considered, the computational aspects of the Kalman filter should be taken into account. The Kalman filter algorithm requiring (inverse) matrix computations might be relatively heavy to be calculated on a wireless sensor node. The node can usually only manage the basic math operations and it has a limited memory available. Nevertheless, in some cases, it could be possible to perform some pre-calculations so that the computation of the Kalman filter would be arranged in a manner that better supports running the algorithm with the scarce resources of a sensor node.

Regardless of the analysis presented in [P5], [P7] and [P8] most of the results are based on numerous simulations and computations that have been run on MATLAB and Simulink. Some of the experiments have also made use of the MoCoNet platform [P3], which executes the algorithms designed in Simulink on the MATLAB-based xPC Target real-time operating (RTOS) system. The optimization routines provided by MATLAB have been extensively used throughout the thesis as in many cases the development of the tuning methods and rules is based on optimizing certain performance criteria.

The classification of the key properties of the scenarios, tools and main results that were discussed in this section is shown in Table 4. In the table, the controller time domain rows indicate if a continuous-time or a discrete-time controller is considered in the publication.

Table 4. Classification of the scenarios, tools and results of the publications.

Property	Publication	[P1]	[P2]	[P3]	[P4]	[P5]	[P6]	[P7]	[P8]
Controller time domain	Continuous-time	x				x	x	x	
	Discrete-time	x	x	x	x				x
Architecture	Semi-distributed	x	x	x		x	x	x	x
	Fully-distributed			x	x				
PID filter	D-term	x	x		x				
	Pre-filter					x	x	x	x
Main results	Tools	x		x					
	Tuning method		x		x	x	x		x
	Tuning rules		x			x	x	x	x
	Analysis					x		x	x
	Simulations	x	x	x	x	x	x	x	x
	Experiments			x	x	x			

The architecture rows refer to the structure of the control system. In the semi-distributed case, the varying time-delay or the network is assumed to reside in the process output, whereas in the fully-distributed architecture there is a varying time-delay or a network also between the controller and the actuator. The PID filter refers to the structure of the PID controller, whether it has the filter only in the D-term (see (34)) or if the whole output signal is filtered before assigning it to the controller (see (58) - (59)). The main results row characterizes the contributions of the publications, whether they involve developing tools, tuning methods and rules or if analysis, simulations and experiments are performed and reported.

4.2. Abstracts of the Publications

This section briefly presents the main assumptions and results of each publication.

[**P1**] The publication presents a PID controller tuning tool that can be used for controller design and tuning in NCS. The tuning tool implements the tuning method for discrete-time PID controllers presented in [P2]. The tool can also be used for tuning continuous-time PID controllers and it supports tuning rule development as it can handle a batch of processes at once. Based on the optimized controller parameter surfaces produced by the tool, the dependencies of the process parameters and the optimal controller parameters can be found and tuning rules derived. The performance of the controller can be evaluated from different points of view as the delay type, noise and disturbance properties, and reference signals may be easily varied.

[**P2**] The publication discusses the tuning of discrete-time PID controllers for NCS. The network is modeled by Gaussian distributed random delays. The tuning method is formulated as a constrained optimization problem with constraints on the gain and phase margins. The phase margin constraints are formulated based on the delay distribution. The system stability for a worst-case delay is guaranteed also by a constraint. In the tuning, a unit step is used as the reference to

the simulated closed-loop system and the performance measure is the ITAE cost criterion (81). This is justified, since using the ITAE criterion in the optimization of controller parameters on stable or marginally stable plants typically results in well dampened responses. Qualitatively, this property of ITAE optimization is due to time-weighting in the criterion, which makes the cost function value significantly increase if oscillation or, in general, errors occur even a "long" time after the step. Additionally, the criterion allows for a slow response right after the step, since due to the time-weighting of the error, its values immediately after the step are relatively insignificant. Thus the response may initially change slowly, which also supports approaching the reference smoothly. This has a positive effect on the stability and robustness of the system. In the paper, the method is applied for an example FOTD system and the resulting optimal controller parameter surfaces are shown. They are also compared with the results obtained from solving the optimization problem for a constant delay case. The differences of the parameter surfaces are clearly seen in the results.

[**P3**] The publication presents the MoCoNet platform that can be used, among other things, for testing NCS remotely. The platform has been designed to be used for both education and research of (networked) control systems. There is a network simulator in the platform through which the measurements and control signals can be passed during the process runtime. Thus networks with different properties (delay distributions, packet loss probabilities) can be simulated and, in addition, the simulated network can be attached to any real process available in the system. The performance of the designed controllers can be easily tested with real processes with the aid of simulated networks. The networks are modeled with different delay distributions, such as Gaussian and Gamma distributions, and the probability of packet loss. There are certain default settings for a few networks, but the users can also define their own networks, for example, by changing the parameters of the delay distributions. The platform provides a graphical user interface in the Internet, through which the experiments are configured and observed. It supports a webcam and live data from the process, and it stores all the experiments in a database for later investigations. The platform can be easily extended with new processes, and thanks to its flexible and component based architecture, different components can be replaced. As an example of the flexible architecture, the network simulator was recently replaced by the ns-2 simulator, which provides more realistic network models and an ability to implement, for example, routing protocols in wireless sensor networks [56], [57].

[**P4**] The publication discusses the tuning of a discrete-time PID controller for NCS. The tuning method is based on simulation assisted optimization as is the case in [P2]. The difference in these methods is that in [P4] different realizations of the varying time-delay are considered in the simulations and the optimization is applied on the worst-case delay realization, that is, the realization that causes the highest cost function value over several step responses. In the paper, the concept of relative cost variance is proposed as the measure of robustness of tuning. The relative cost variance basically describes the variance of the performance measure over several simulations with different realizations of the delay. Hence, if the tuning is very robust, the different simulations

differ only a little and the variance becomes small. Additionally, this allows comparing the results obtained with different sample times.

Another difference to the work presented in [P2] is that in this publication the results are provided for the fully-distributed scenario of the NCS. The proposed tuning method is also compared with other PID tuning methods and the IMC method. The comparisons are made by simulations and with the MoCoNet platform using a case process and typical Internet parameters for the simulated network. The choice of a suitable sample time is also discussed, and it is concluded that the sample time should be chosen based on the delay distribution so that the delay experienced by the controller would most often vary as little as possible (one or two samples in this case).

[**P5**] This publication discusses the PID controller tuning for marginally stable, integrating processes with varying time-delays. For the process model, the FOLIPD (First Order Lag plus Integrator plus Delay) structure is assumed. The paper shows that if a certain PID tuning rule structure is applied for the FOLIPD process model, the analytical jitter margin condition of the control system has a relatively simple form. Although the jitter margin cannot be solved analytically, it is solved numerically for a wide range of processes. By doing so and then by identifying the resulting jitter margin-controller gain surface, simple tuning rules are derived. The rules then depend on the required jitter margin, which is often desirable. In addition, the same tuning method that is proposed in [P6] is applied here to obtain an alternative set of tuning rules for integrating processes. The difference of these two sets of proposed rules is that the first set gives a maximum gain for the controller without considerations of the robustness to noise and disturbances, but the desired jitter margin can be given as an input parameter to the rules. In the second case, the obtained rules maximize the performance and the jitter margin simultaneously, and satisfy the M-circle robustness with $M = 1.5$ in the case of nominal plants.

The tuning rules are tested in a case process involving an agricultural tractor (control over the CAN bus). The proposed tuning rules are also compared with other tuning rules found in the literature, and the simulation results indicate that the proposed rules have many appealing properties in varying time-delay systems. The methodology has later been applied in the control design of a wirelessly controlled DC motor (simulation case study, [16]).

[**P6**] This publication focuses on developing new tuning rules for the PID controller in stable systems with varying time-delays. The tuning method presented is based on simultaneous optimization of the closed-loop system performance and the jitter margin. The goal attainment method is applied in solving the multi-objective optimization problem, and robustness of the controller is considered by bounding the maximum values of sensitivity functions by the M-circle (75) criterion. The simulation based optimization problem is solved for a range of processes, and based on the optimal values of the parameters, the tuning rules are identified. The proposed tuning rules are compared with the AMIGO tuning rules [103]. It is shown that compared to the AMIGO tuning, the proposed tuning may provide significant improvements in performance and robustness to delays in terms of ITAE-performance and jitter margin, respectively.

[**P7**] This publication first analyzes the performance of the AMIGO tuning rules in varying time-delay systems. The AMIGO rules are considered as the state-of-the-art PID tuning rules for process control and here their performance in systems with varying time-delays is evaluated. It is concluded that in order to have positive jitter margin, filtering is needed in PID control. The ideal PID controller does not have roll-off at high frequencies, which may result in zero jitter margin. Unless the process makes the magnitude of the complementary sensitivity function decrease at higher frequencies, filtering is required. A control system without the filter may still be stable, but there are no guarantees of stability in the sense of jitter margin, if additional delays are introduced in the control loop. The main result in the paper is the proposition of the extended plant approach for PID controller tuning in varying time-delay systems, where the process itself is assumed to be stable. The idea is to first design the measurement filter using the tuning rule provided that depends on the desired jitter margin, and then to filter the step response of the plant with the designed filter. Based on the filtered response the FOTD model parameters are determined and the AMIGO tuning is applied in order to have the controller gains. Using this approach, it is shown that the desired jitter margin is achieved relatively well for the process test batch including over 100 processes. The tuning rules are also compared in simulations with the AMIGO rules and the superiority of the proposed tuning in varying time-delay systems is clearly shown.

[**P8**] In this publication, discrete-time controllers are compared in a networked control framework, where the network is modeled either as constant delay, random delay, or correlated random delay. There are altogether five different controllers compared in the study and there is also some novelty in the structure of the discussed controllers. Two of the controllers are PID (time-driven and event-driven), two IMC (regular and delay-adaptive), and one is a fuzzy gain scheduler of five PID controllers. The event-driven PID controller acts only upon receiving new packets, the delay-adaptive IMC controller updates the control law based on the measured delay, and the fuzzy gain scheduler calculates a weighted sum of the outputs of five PID controllers based on the delay value. For the controller tuning, a simulation based optimization technique is proposed, where the optimization constraints are formulated based on the desired jitter margin. Here again, only stable processes are considered. The comparison of the controllers is done by extensive simulations. The additional complexity of the delay-adaptive controllers does not necessarily improve the performance of the control system if varying time-delays are present. With the regular PID controller comparable performance may be achieved if the controller tuning is done as proposed in this paper. Furthermore, the differences in performance among the controllers vanish as the ratio of the delay amplitude and the process time-constant becomes small, and the delay range is clearly smaller than the time-constant of the process.

4.3. Discrete-Time PID Controller Tuning

One of the three main contribution areas in the thesis is the design and tuning methods for the discrete-time PID controller. The work presented in [P2], [P4] and [P8] is summarized in this section. In the publications, stable processes are considered, but some of the methods could be

extended to unstable processes as well. In [P2] and [P4], the tuning of a discrete-time PID controller is considered and the controller has the form seen in (35) - (38), where only the derivative term of the controller has a filter. On the contrary, in [P8], the discrete-time PID controller considered filters the process output and the filtered signal is used in all terms of the controller as in (58) - (59). In [P2] and [P8], we consider a varying time-delay in the process output due to, for example, wireless sensor network dynamics, but in [P4] the system is fully-distributed having two varying time-delays in the control loop. The tuning is based on optimizing the closed-loop system performance using simulation and constrained optimization methods.

Generally, the optimization problem is defined as in (86), where $J(x)$ is the cost function, x is a vector of decision variables (controller gains), and $g(x)$ and $h(x)$ are the equality and inequality constraint functions. In [P2] and [P4], $J(x)$ is the ITAE cost function (81), but in [P8] the IERC cost function (85) is used. Since the studies consider only stable processes, the use of ITAE is motivated, because in the case of stable processes the ITAE-optimized responses usually have little oscillation. In addition, the IERC cost function also considers the use of the control signal. In [P4], the cost function is the maximum ITAE cost of K simulations with different realizations of the delay. In all cases, the controller gains may be kept positive by a constraint $x > 0$. Equality constraints are not used here, but with inequality constraints different requirements for the parameters and for the control system are formulated. In [P2], the constraints are formulated such that the tuning guarantees adequate gain and phase margins for the closed-loop system with all possible constant values of the delay. In addition, the stability is guaranteed for a worst-case delay scenario, for which the discrete-time closed-loop system is stable. In [P8], the stability of the closed-loop system is guaranteed even if the delay varies within the allowed range by formulating the jitter margin condition (25) into a constraint function form for the problem. This is done by introducing a constraint function $\delta_{\max}(x) > \delta_0$, where δ_0 is the desired jitter margin defined by the application requirements and $\delta_{\max}(x)$ is the jitter margin provided by the PID controller with controller gains x. In [P4], constraints are not used in optimization.

The optimization problem (86) is solved by simulating the closed-loop system, evaluating the performance criterion $J(x)$ for a unit step reference, and iterating the controller parameters according to the scheme discussed in Section 3.5.3. In [P2] and [P8], the process is modeled as a simple first order system. In [P4] the tuning problem is solved for a case process, which is of second order. In [P2] and [P8], the optimization problem is solved for a range of processes with different time-constants. In [P2], also the sample time is varied. As an example, the optimized PID parameters for the case with Gaussian distributed random delay with mean value μ = 1 s and variance σ^2 = 0.1 s^2 are shown in Figure 15 [P2]. Clearly, both the time-constant and the sample time affect the optimal gains, but the cost function values are almost constant for a fixed sample time regardless of the time-constant, although one would expect the cost function to increase at higher values of the time-constant.

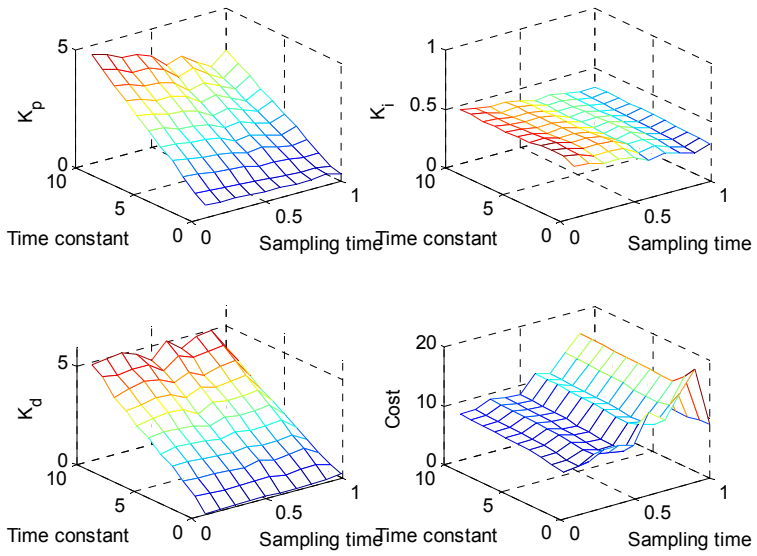

Figure 15. Optimal PID controller parameters and ITAE cost function values for random Gaussian distributed (μ = 1 s, σ^2 = 0.1 s^2) delay [P2].

This behavior shows that the sample time has a significant effect in varying time-delay systems as is also discussed in [P4], where the optimal sample time vs. the varying time-delay aspect is further elaborated. It is concluded that the optimal sample time has the characteristic that it makes the delay in sample times vary as little as possible (usually only one or two sample delays). In addition, in Figure 15, it is seen that the cost function dramatically changes at certain sample times. For example, at h = 1 s, the performance is clearly better than for h = 0.9 s. This could be explained by the fact that the mean delay and the sample time are equal at h = 1 s. Hence half of the measurements experience only one sample delay, whereas in the case h = 0.9 s more than half of the measurements experience a delay of more than one sample time (1.8 s or more). Especially the ITAE criterion considered here, increases rapidly due to delays.

In addition to the PID controller tuning method, [P8] presents a comparison of discrete-time controllers in varying time-delay systems, where the main idea is to analyze if by using a slightly more complicated controller structure, high gains in performance or robustness would be achieved. In order to do this, a delay-adaptive IMC controller (Section 3.3.1) and a fuzzy gain scheduler (Section 3.3.2) are formulated for the control of varying time-delay systems. The tuning results and the analysis performed show that the optimal tuning of the IMC controller depends on the process dynamics and the required jitter margin. This is shown by simple calculations.

The performance of the controllers is compared by extensive simulations in three scenarios with constant, random or correlated random delay in the loop. Correlated random delays are often encountered in NCS, since upon network congestion, the delay times typically increase and remain high until there is less traffic in the network. The investigations show that, in general, the increased complexity of the controllers provides some benefits over the regular PID controller, especially if the delay values are correlated (see Figure 16), but the gains in performance decrease as the delay amplitude becomes insignificantly small compared to the time-constant of the process.

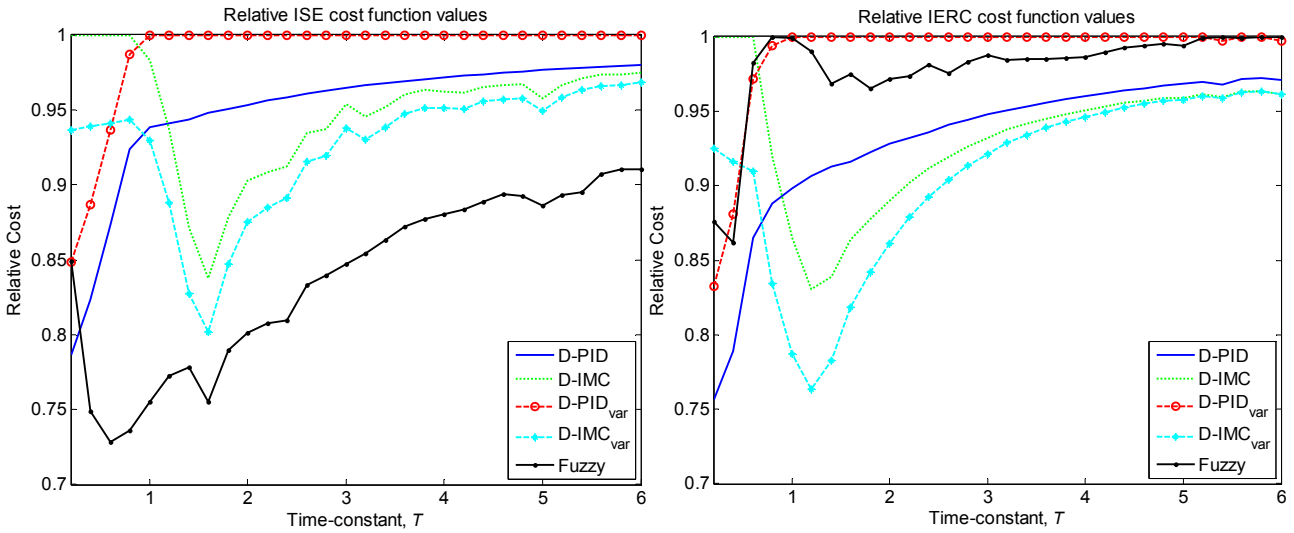

Figure 16. Relative ISE and IERC cost function values for the correlated random delay case [P8].

In Figure 16, the relative performance of the compared controllers is evaluated for a first order system with correlated random delays. The performance criteria (ISE and IERC separately in the two diagrams) are scaled such that the controller with the highest criterion value equals one in the plots (lowest performance). The performance is plotted against the time-constant of the process. On the left, where the ISE criterion is shown, it is seen that the fuzzy gain scheduler performs the best for almost all values of the process time-constant. Also the delay-adaptive IMC controller is significantly better than the PID controllers for $T > 1$. The evaluation of the IERC criterion on the right reveals that the enhanced performance of the fuzzy gain scheduler in the ISE sense results from a very tightly tuned controller, which significantly uses the control signal. If the control signal is considered (as in the IERC criterion), the performance of the fuzzy controller is not very good. Nevertheless, the IMC controllers seem to perform well with respect to both criteria, and by adapting to the delay some additional benefits can be achieved. In any case, the regular PID controller that is tuned to be robust against varying time-delays performs adequately well in the scenario.

4.4. Tuning Rules for PID Controllers

In [P5], [P6] and [P7], the objective is to develop tuning methods for the PID controllers and to apply them for a wide range of processes in order to also derive tuning rules. The tuning rules should have a simple structure and they should be easily applicable for various process types. A desired property of the tuning rules for NCS is that the jitter margin would be an input parameter to the rules, so that the controller parameters would adapt directly to, for example, the network QoS characteristics (jitter, maximum delay). Another desirable property of the tuning rules is that they would provide robustness to noise and disturbances. These aspects are considered in the tuning methods and rules proposed in the thesis, and a short summary of them is given in this section.

4.4.1. Tuning Method

The PID controller tuning method proposed in [P6] is based on simultaneous maximization of performance and jitter margin using simulation based optimization with constraints regarding the robustness to noise and disturbances. In [P6], the focus is on stable processes with varying time-delays, and in [P5] the same tuning method is applied for marginally stable, integrating processes. In both publications, novel tuning rules for varying time-delay systems are also developed.

The multi-objective optimization based tuning method is formulated as follows:

$$J = \begin{bmatrix} J_{ITAE} & \dfrac{1}{\delta_{\max}} \end{bmatrix}, \tag{88}$$

$$h(x) = \min_{\omega} \left(\sqrt{\left[\mathrm{Re}\left(H(j\omega)\right) - c_R\right]^2 + \left[\mathrm{Im}\left(H(j\omega)\right)\right]^2} - r_R \right) > 0, \tag{89}$$

where $H(j\omega) = G(j\omega)C(j\omega)$ is the loop transfer function frequency response and c_R and r_R are the M-circle (75) center point and radius. Here the objective functions are the ITAE cost for a unit step reference and the inverse of the jitter margin, which should be minimized. Hence we aim at maximizing the performance and the jitter margin simultaneously. The constraint function (89) is formulated such that the tuning should satisfy the M-circle robustness. This would guarantee that the tuning is also robust in the traditional sense, to noise, disturbances and modeling errors.

In [P5] and [P6], the optimization problem is solved using the goal attainment method, which is one of the multi-objective optimization methods. The reason for using this method is that it allows specifying the goal values for the objective functions. For example, the goal value of the ITAE criterion can be easily set by calculating the criterion with some other tuning method, which has the desired and comparable performance. The goal value for the jitter margin depends on the application, but as it is concluded in [P6], for a FOTD process it might be reasonable to set the goal as $T + \tau$ (time-constant + delay), which corresponds to the *effective dead-time* (see [103]) of the process. In this case, the jitter margin would be relatively large compared to the total dynamics and speed of the process, and it does not matter if the process is delay-dominant ($\tau >> T$), lag dominated ($\tau << T$), or if it is a balanced lag and delay process ($\tau \approx T$). In [P5], the goal values are set similarly $T_f + \tau$ (filter time-constant + delay) for the same reasons.

4.4.2. Stable Processes

In [P6], the tuning problem (88) - (89) was solved for the FOTD process model with different process parameters (delay, time-constant) in the range $\tau, T \in [0.1 \ldots 10]$. The reason for choosing the FOTD model was that it can be used to characterize processes with higher order dynamics, and the parameters of the model are obtained via a simple step response experiment. It is also the basis of many well-known PID tuning rules (e.g. AMIGO tuning [103], Ziegler-Nichols tuning [100]). In fact, the AMIGO rules were taken as the reference tuning, since they provide good robustness properties for processes with constant delays. Nevertheless, it should be emphasized that

the AMIGO rules are designed primarily for load disturbance rejection, but as seen in the definition (88), the proposed method focuses on tracking the changing reference signal.

After solving the optimization problem, comparisons to the AMIGO tuning revealed that the jitter margin can be significantly improved without dramatic losses in either performance or robustness. The upper plot in Figure 17 presents the improvement of the jitter margin for different processes in percents compared to the AMIGO tuning, when the proposed tuning method is applied. The typical improvements to the jitter margin are around 50 – 150 %. At the same time, the performance (ITAE) of the closed-loop system has not decreased significantly, as can be seen in the lower plot of Figure 17. The graph presents the ITAE ratio of the AMIGO tuning with respect to the optimal tuning. Thus the values less than 100 % indicate that the performance of the proposed tuning is less than that of the AMIGO tuning. Although most of the dots are less than 100 % indicating losses in performance, the values are not, however, dramatically lower than 100 %. On the contrary, there are even cases where both the performance and the jitter margin are better with the optimal tuning.

To further develop tuning *rules* for the PID controller, the optimal controller parameter values were plotted with respect to time-constants and delays to obtain a surface, from which the tuning rules were derived using a procedure that we here define as *surface identification*. This procedure involves first determining the structure of the tuning rule, which is a function prototype with unknown parameters, and then estimating the parameters based on the optimal controller parameter surface and the tuning rule structure. The parameter estimation is done using the least squares method.

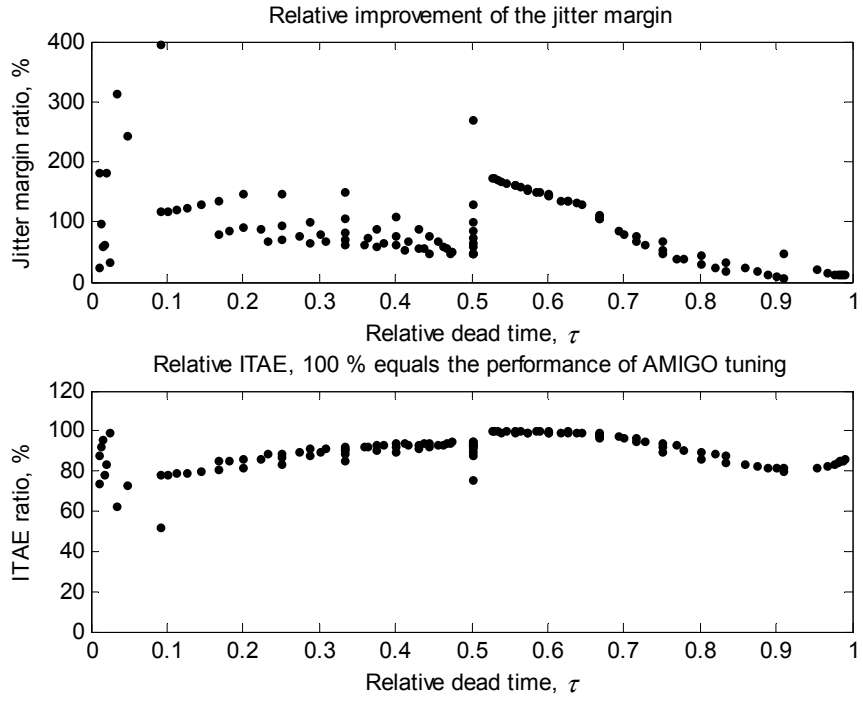

Figure 17. Performance of the optimized controllers with respect to the AMIGO tuning.

The *jitter-aware tuning rules* proposed for varying time-delay systems are

$$\begin{aligned} k_p &= \frac{1}{K_p}\left(\frac{0.4T-0.04}{\tau}+0.16\right), \\ k_i &= \frac{1}{100K_p}\left(\frac{-0.11T^3+1.5T^2-1.5}{\tau^2}+\frac{0.35T^2-4T+50}{\tau}\right), \\ k_d &= \frac{1}{100K_p}\left(0.4T^2+11T\right). \end{aligned} \tag{90}$$

Other controller parameters such as the filter time-constant and the set-point weights are chosen as in the AMIGO tuning (see (60)). For the pure FOTD process, the use of these rules results in improvements of approximately 100 % in the jitter margin and comparable performance, if the AMIGO tuning is used as the reference. The performance of the tuning rules is further discussed at the end of this section, where simulation results are presented.

Alternatively, tuning rules for stable processes may be developed by approaching the problem analytically. The idea presented in [P7], which considers the so called *extended plant approach*, is based on the assumption that the closed-loop system frequency response magnitude can be approximated by that of the open-loop system (the high-frequency approximation)

$$\left|\frac{G(j\omega)C(j\omega)}{1+G(j\omega)C(j\omega)}\right| \approx \left|G(j\omega)C(j\omega)\right|. \tag{91}$$

If a FOTD process is controlled with an AMIGO tuned PID controller with an n^{th} order filter and the approximation (91) is used, the jitter margin of the system becomes

$$\delta_{\max} \approx \inf_{\omega\in[0,\infty[}\left|3\frac{\left(1+j\omega T_f\right)^n}{j\omega}\right|. \tag{92}$$

If the infimum in (92) is solved by minimization with respect to frequency, the following tuning rules for the filter time-constant T_f are obtained.

$$T_f = \begin{cases} \dfrac{1}{3}\delta_{\max}, & n=1 \\ \dfrac{1}{3\sqrt{n}}\left(\dfrac{n}{n-1}\right)^{(1-n)/2}\delta_{\max}, & n>1. \end{cases} \tag{93}$$

Thus we may set the filter time-constant based on the jitter margin requirement, and using this filter in the control system should give the system the desired jitter margin. The steps in the tuning procedure to achieve jitter-aware tuning for the controller are:

1) Make a step experiment with the process
2) Design the filter using (93)
3) Extend the plant with the filter, i.e., filter the step response
4) Approximate the FOTD model from the extended plant (using the filtered step response)
5) Use the FOTD parameters to calculate the PID gains based on the AMIGO tuning rules except for the filter time-constant, which is already calculated

The steps of the extended plant approach are illustrated in Figure 18.

If these rules are applied, the resulting closed-loop system approximately satisfies the desired jitter margin that was used as the design parameter. There are certain limitations and modifications especially for delay-dominant systems ($\tau >> T$), and these are discussed in detail in [P7]. The performance of the proposed method is depicted in Figure 19, where the jitter margin is evaluated for the process test batch discussed in [103] with over 100 processes. The extended plant approach is applied to the processes in the test batch such that the jitter margin parameter in the tuning rules (93) is set twice as large as the jitter margin provided by the AMIGO tuning for each process. The proposed tuning procedure is then applied and the relative jitter margin with respect to the AMIGO tuning is plotted in Figure 19. Thus the values in Figure 19 should be aligned at 100 %.

In Figure 19, cases with the second and the third order filters are presented. As it can be seen, the performance of the method is relatively good despite the approximation (91) used in its derivation. In addition, some degradation of performance is observed when comparing the ITAE costs with the AMIGO tuning.

The performance of the tuning rules discussed above is compared next. In the comparison, both delay-dominant and lag dominated processes are considered. The process models are

$$G_{\exp,1}(s) = \frac{1}{0.3s+1}e^{-s}, \qquad G_{\exp,2}(s) = \frac{1}{(s+1)\left[(0.1s)^2 + 0.14s + 1\right]}. \tag{94}$$

$G_{\exp,1}$ is a pure FOTD process and $G_{\exp,2}$ is the first one of P_9-type processes in the test batch, see [103] for details.

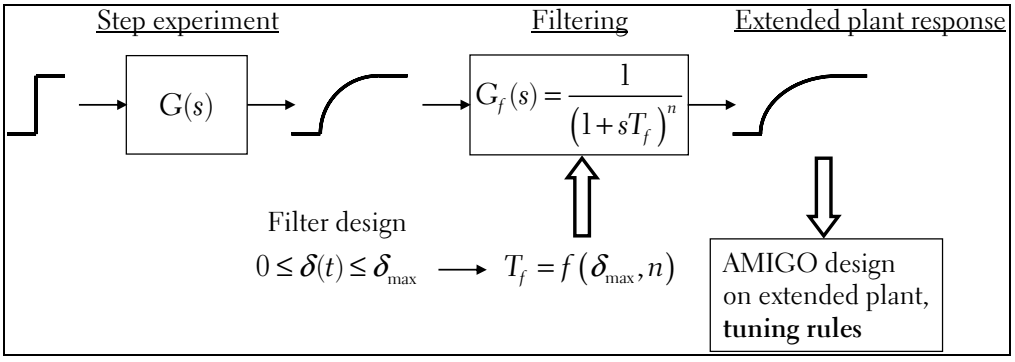

Figure 18. The steps in the extended plant approach.

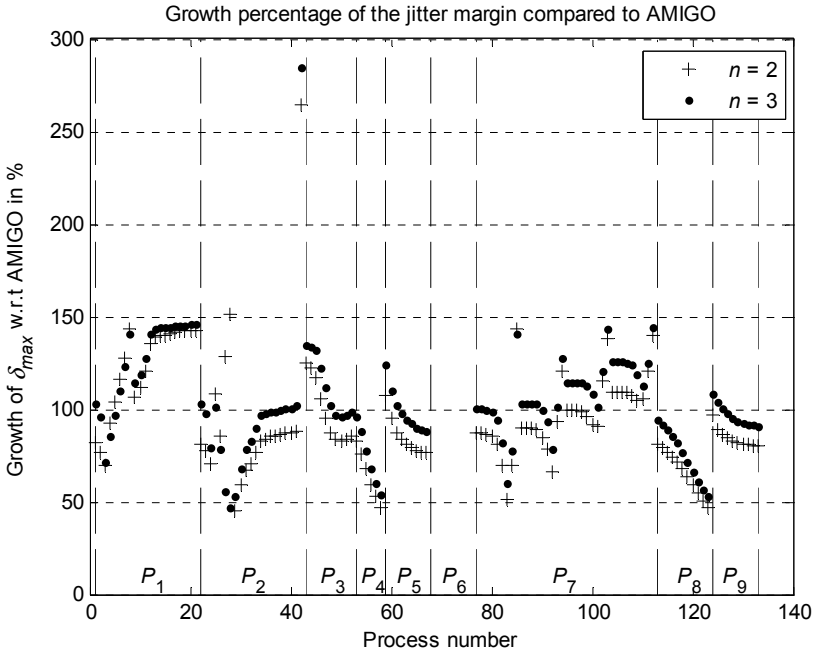

Figure 19. The jitter margin of a process test batch when using the extended plant approach (target value 100 %).

First, the FOTD parameters of the processes are derived. The FOTD parameters for the first process are the true process parameters (K_p = 1, T = 0.3 and τ = 1) the process being delay-dominant. For the second process, the parameters are K_p = 1, T = 1 and τ = 0.137 the process being lag dominated. For both processes the standard AMIGO tuning rules, the extended plant approach with n = 1,2,3 and the jitter-aware tuning rules (90) were calculated. For the delay-dominant process $G_{exp,1}$, only the second and the third order filter designs were applied, because as explained in [P7], the first order filter is not applicable for delay-dominant processes, when the extended plant approach is used. The AMIGO and the jitter-aware rules (90) use the second order measurement filter.

For the extended plant approach the jitter margin design parameter δ_{max} was set twice as large as the jitter margin provided by the AMIGO tuning. The resulting controller parameters are shown in Table 5 and Table 6, for $G_{exp,1}$ and $G_{exp,2}$, respectively. In the simulations, a varying time-delay (square-wave form) with maximum amplitude of 1.9 times the jitter margin of the AMIGO tuning was added after the true plant and the different controllers were tested in this setting. The unit step responses of the systems with various controllers can be seen in Figure 20 (delay-dominant) and Figure 21 (lag dominated). In the figures, the responses represent the process output variables measured before the variable delay, which is located between the process output and the controller measurement filter.

Table 5. Controller parameters for the delay-dominant process $G_{ext,1}$. The target value for δ_{max} is 1.3 (in extended plant).

Tuning	δ_{max}	k_p	k_i	k_d	T_f	ITAE
AMIGO	0.65	0.33	0.54	0.08	0.10	10.41
PID rules (90)	1.35	0.24	0.48	0.03	0.10	11.08
Ext. n = 2	1.33	0.56	0.43	0.27	0.50	9.60
Ext. n = 3	1.28	0.48	0.40	0.23	0.30	8.35

Table 6. Controller parameters for the lag dominated process $G_{ext,2}$. The target value for δ_{max} is 0.42 (in extended plant).

Tuning	δ_{max}	k_p	k_i	k_d	T_f	ITAE
AMIGO	0.21	3.47	7.04	0.23	0.014	∞
PID rules (90)	0.28	2.78	3.32	0.11	0.014	2.76
Ext. n = 1	0.39	2.22	3.44	0.24	0.14	1.62
Ext. n = 2	0.42	2.00	3.04	0.24	0.071	1.50
Ext. n = 3	0.44	1.83	2.69	0.24	0.054	1.64

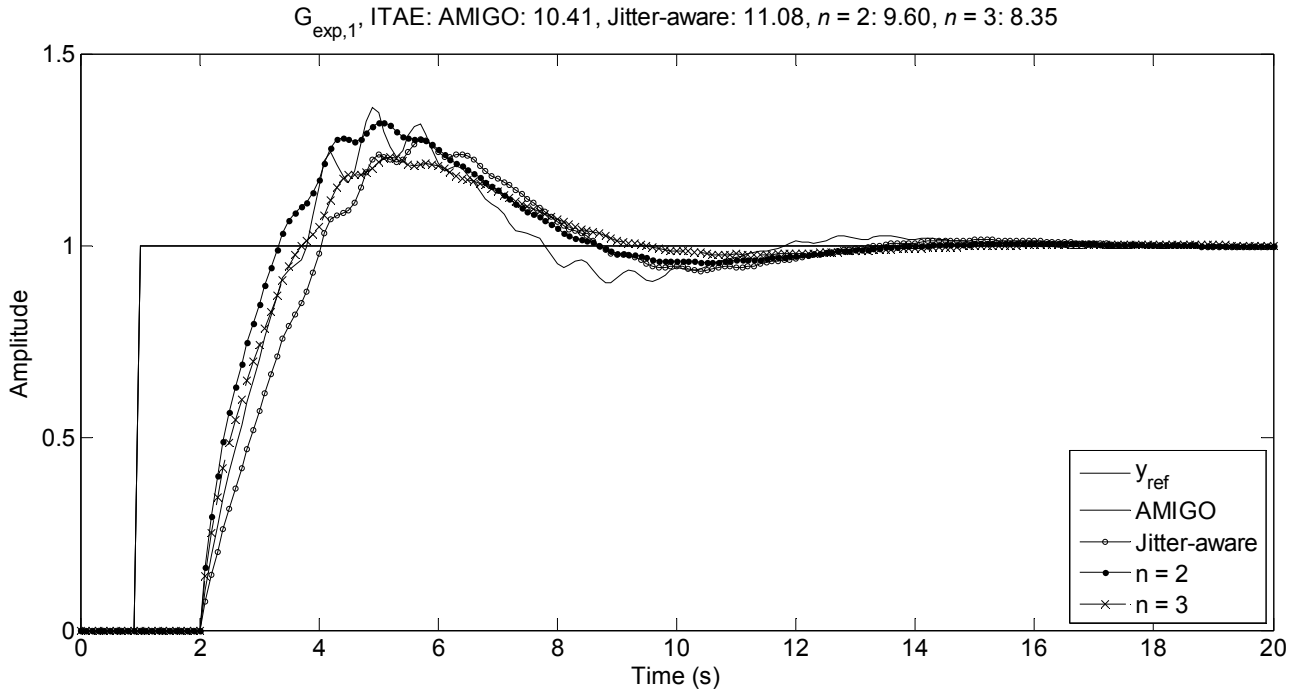

Figure 20. Simulation results from the delay-dominant process $G_{ext,1}$.

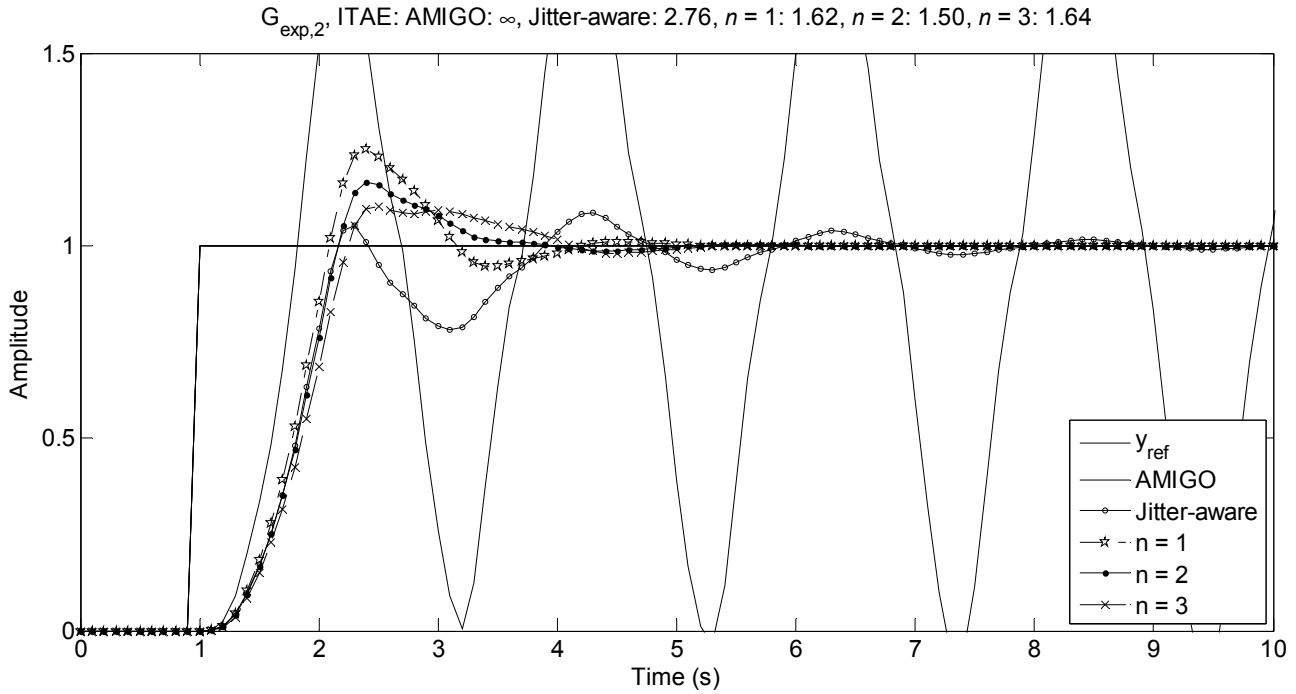

Figure 21. Simulation results from the lag dominated process $G_{ext,2}$.

The step responses of the delay-dominant process show that all the systems are stable for the specific realization of the varying time-delay, although the AMIGO and the jitter-aware tunings show signs of being close to unstable. According to (25), the AMIGO tuning does not guarantee stability with the used amplitude of the time-delay as seen in Table 5, but this does not mean that the system would be necessarily unstable. The controllers tuned using the extended plant approach and the jitter-aware rules are guaranteed to be stable for this case, and the extended plant approaches give the best ITAE values for the responses. The ITAE values are calculated from the process output variable (before the variable delay in the loop).

The lag dominated system with the standard AMIGO controller is unstable, but the extended plant approach gives fast and stable responses. Also the jitter-aware rules provide a stable response, although it is clearly affected by the varying time-delay. Of the extended plant designs the controller with the first order filter gives the largest overshoot. The controllers with the second and the third order filters give accurately the desired jitter margins (as seen in Table 6), but the requirement is not exactly satisfied with the first order filter. The jitter-aware tuning rules do not give as high jitter margin as the extended plant approach in this case, but this is expected, since the tuning rules do not take explicitly into account the requirement of the jitter margin. Nevertheless, for both processes the proposed tuning rules give excellent results with respect to both robustness to jitter and performance, compared to the standard AMIGO design.

4.4.3. Integrating Processes

In [P5], tuning of a PID controller for marginally stable integrating processes with varying time-delays and stringent robustness constraints is considered. In the paper, the novel controller tuning method discussed in Section 4.4.1 that uses the jitter margin and the ITAE performance as the optimized objectives simultaneously is used to develop robust tuning rules for integrating processes. The proposed tuning rules satisfy similar robustness constraints that were used in the AMIGO tuning rule development [103], [104]. Some of the recently proposed tuning rules for integrating processes are also analyzed in the paper, and based on the analysis the relationship between the jitter margin, process delay and the maximum controller gain is determined. As a result, the required jitter margin can be given as a parameter to the tuning rules and the maximum controller gain can be calculated.

The process model considered in the tuning rule development for integrating processes is

$$G(s) = \frac{K_v}{s(1+sT_f)} e^{-s\tau}. \tag{95}$$

This is the so called FOLIPD model, where K_v is the velocity gain and T_f could be considered either a lag in the process or a filter time-constant in the controller that filters the process output before the signal is processed by the controller. The PID controller has the form (58), but no additional filter is assumed (in spite of T_f).

First, it is noticed that many of the tuning rules for the FOLIPD processes have a similar form [63]

$$k_p = \frac{a}{K_v \tau},\ k_i = 0,\ k_d = \frac{aT_f}{K_v \tau}, \tag{96}$$

where a is the controller gain. By this tuning the open-loop system $G_l(s)$ becomes independent of T_f and K_v,

$$G_l(s) = \frac{a}{\tau s} e^{-s\tau}. \tag{97}$$

If such a tuning rule structure is applied, the jitter margin of the closed-loop system becomes

$$\delta_{\max} = \inf_{\omega \in [0,\infty[} \sqrt{\left(\frac{\tau}{a}\right)^2 - \frac{2\tau}{a\omega}\sin(\omega\tau) + \frac{1}{\omega^2}}. \tag{98}$$

Although the expression (98) seems quite simple, it does not have an analytical solution, which would also provide the inverse solution, that is how a (the controller gain) should be chosen based on the required jitter margin. Since such a solution would be beneficial in various applications, (98) is solved numerically for a very wide range of processes (different values of τ and a), and by surface identification it is possible to derive tuning rules for which the jitter margin becomes an input parameter. First, the jitter margin is solved with parameters $0.368 \leq a \leq 1.008$ and $0.01 \leq \tau \leq 100$. The range for a is justified, because the lower limit provides the control system jitter margins of approximately 2τ, which is considered a high value, and the upper limit gives so high a gain that the overshoot in the step response of the closed-loop system is approximately 50 % [63]. The range of parameters under investigation is further extended in [16] so that the analysis presented there covers cases, where the jitter margin may be given values as high as 10τ. The solutions of (98) for the above mentioned range of process parameters are depicted in Figure 22. Based on the numerical solutions seen in the figure, it is possible to identify the jitter margin surface, and the following relationship in the parameters is found.

$$\delta_{\max} = \left(\frac{0.9485}{a} - 0.6356\right)\tau \tag{99}$$

Thus the controller gain a may be chosen based on the jitter margin requirement and the constant process delay as

$$a = \frac{0.9485\tau}{\delta_{\max} + 0.6356\tau}, \tag{100}$$

from which the PID controller gains can be solved according to (96). The identification of the surface results in estimation error of less than ±3 % of the true value of the jitter margin, and hence the expression (100) can be well applied in practice.

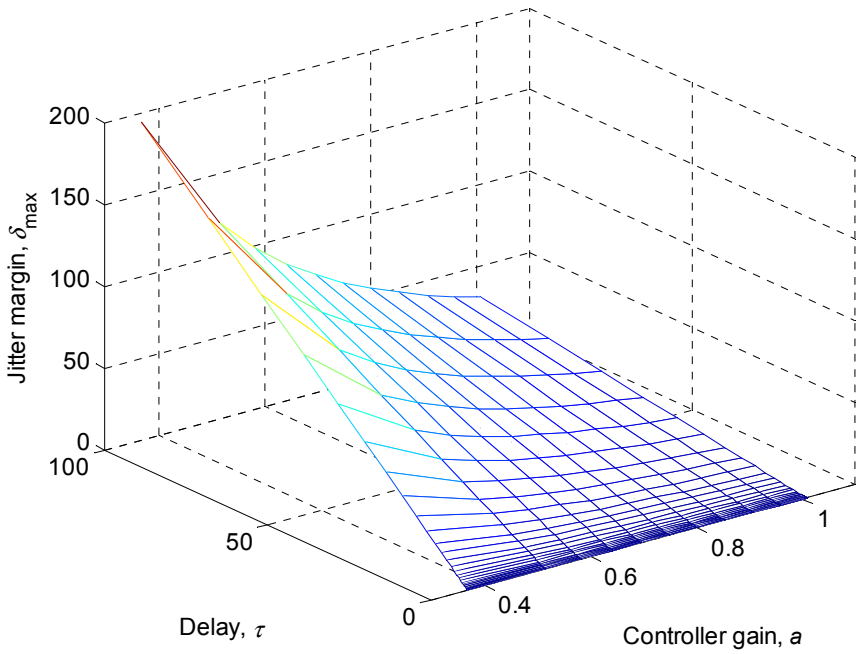

Figure 22. The jitter margin of the FOLIPD process with different values of controller gain a and process delay τ.

In [P5], an alternative tuning approach is also presented that is based on the optimization method discussed in Section 4.4.1. The reason for introducing this alternative tuning method is that the tuning rule (100) does not consider directly the robustness aspects such as the M-circle (75). The controller parameters are again optimized for a wide range of processes by solving the optimization problem (88) - (89) for $0.01 \le \tau$, $T_f \le 100$, whenever $0.1 \le T_f / \tau \le 10$. This restriction on the process parameters was motivated on one hand by simulation accuracy, since the system tends to become stiff for values outside of this range, and on the other hand by reasoning. One of the parameters easily becomes negligible outside of this range. The robustness of the tuning rules was guaranteed by using the constraint regarding the M-circle with M = 1.5. The optimized controller parameters are seen in Figure 23.

Based on the controller parameters in Figure 23 and with the aid of surface identification, the following tuning rules are proposed in [P5] for integrating processes with varying time-delays.

$$k_p = \frac{10^{p(\tau,T_f)}}{K_v \tau},\ k_i = 0,\ k_d = \frac{T_f^{q(\tau,T_f)}}{K_v} 10^{r(\tau)},\ \text{where} \tag{101}$$

$$\begin{aligned} p(\tau,T_f) &= 0.0027\left(T_f/\tau\right)^2 - 0.0794 T_f/\tau - 0.34, \\ q(\tau,T_f) &= 0.02 + \left(0.51 - 0.076\log_{10}(T_f)\right)\tau^{0.15}, \\ r(\tau) &= 0.97 - 1.48\tau^{0.15}. \end{aligned} \tag{102}$$

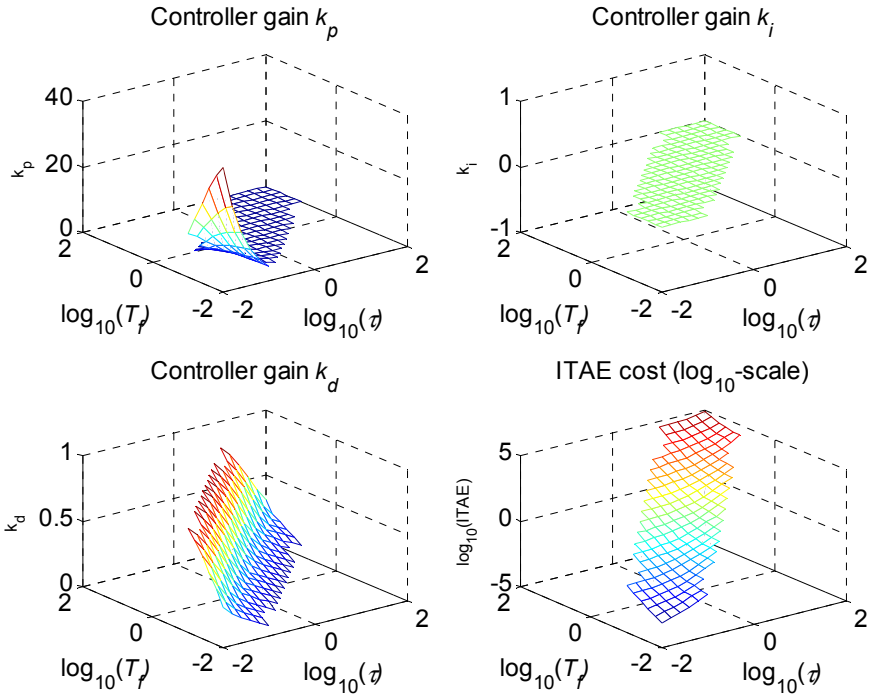

Figure 23. Optimized controller parameters and the ITAE cost.

The performance of the proposed tuning rules is compared in [P5] with other known tuning rules for FOLIPD and IPD processes using both simulations and a case process, which is a part of the hydraulic system of an agricultural tractor. The hydraulic system components are controlled over the CAN bus, and the delays in the control loop are in the range 200…300 ms. Hence the parameters for control design are τ = 0.2 s and δ_{max} = 0.1 s. The measurement filter time-constant was set by comparing the measured and filtered process output variables so that the filter would remove noise efficiently. By such qualitative analysis the measurement filter time-constant was given a value T_f = 0.15. The robustness properties and the simulated step responses obtained by various tuning rules for the case process model are shown in Figure 24. The M-circles in Figure 24 are plotted with M = 1.5 and the curves in the figures represent the loop transfer function Nyquist curves obtained by different tuning rules. It is seen that the proposed tuning rules provide both good performance and the desired robustness level (M = 1.5). From the robustness perspective, clearly the Viteckova (OS=0, overshoot 0 %) rule resembles the proposed tuning (The New Rules, figure in the bottom-right corner), but the proposed tuning provides a slightly faster response. If the settling times of the responses are considered, the Viteckova OS=10 rules provide as fast responses as the proposed tuning, but the robustness level obtained by the proposed tuning is simultaneously better.

The accuracy of the process model can be evaluated from Figure 25, where the step responses of the simulation model and the real process are shown. The accuracy of the model seems reasonable for control design. The differences between the simulated and measured responses are explained by the different values of delays, the non-ideal actuators and the nonlinearities of the real process. In any case, the proposed tuning gives smooth and fast response, which satisfies the requirements of control design.

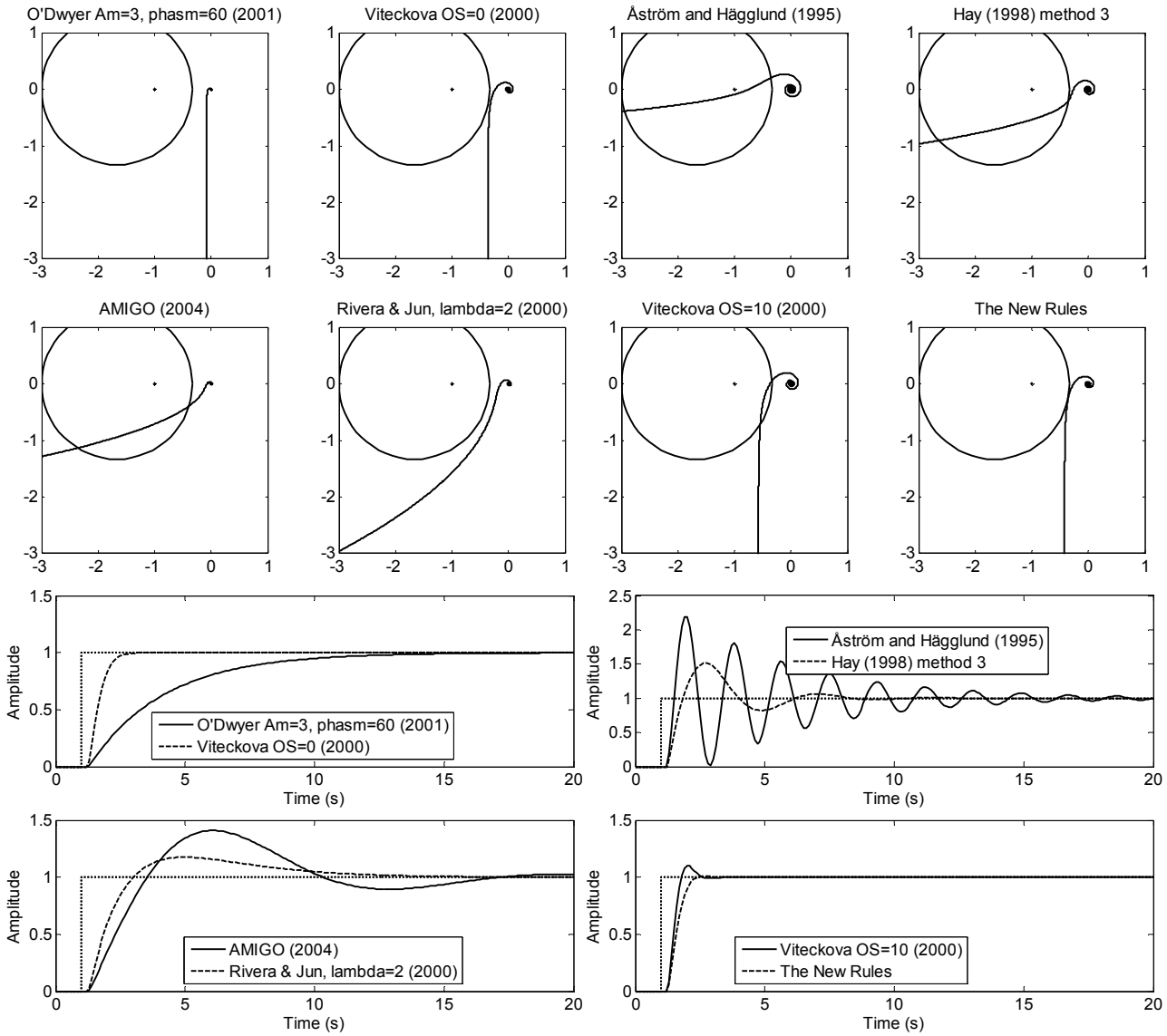

Figure 24. Comparison of the robustness and unit step responses of different tuning rules for the case process.

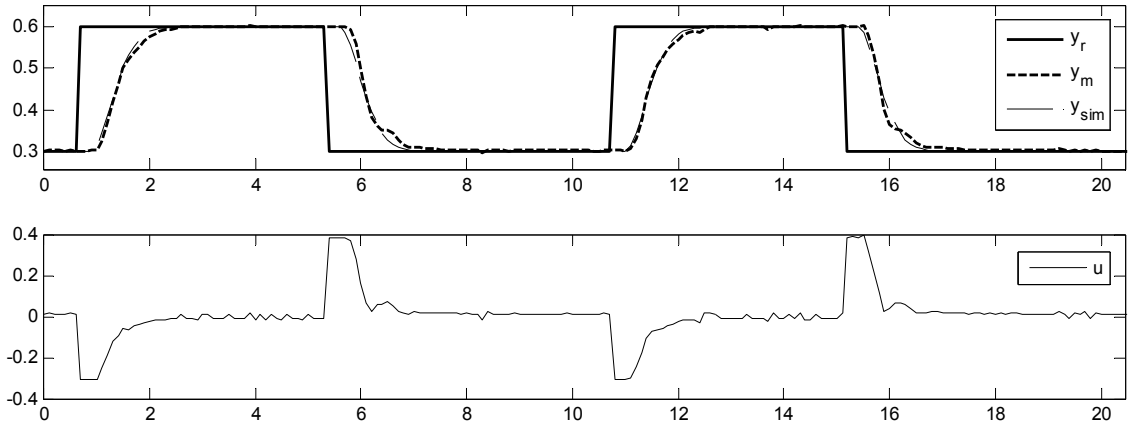

Figure 25. Step responses with the simulation model and the real process using the proposed tuning rules. The control signal (u, lower graph) is from the real process.

The methodology proposed in [P5] has been further developed in [16], where the numerical solution of (98) has been improved. A Newton's method based solver is proposed, which gives a

very accurate solution after only four iterations. The design methodology based on solving the jitter margin is further elaborated by presenting another case example of control design for varying time-delay systems. In this case, the control design takes the advantage of Mirkin's lemma [49] discussed already in Section 2.2.4. The idea is to design a controller for a DC motor that is controlled over a wireless link with packet losses. Varying time-delays are not considered here, and the focus is purely on packet loss. The requirement for the controller is to tolerate 10 consecutive packet losses, while maximizing the speed of the control system. The sample time of the system is h = 0.05 s. The control design is first done in CT, but the resulting controller is then approximated by its DT equivalent in the implementation phase.

The process model used is

$$G(s) = \frac{K_v}{s\left(1+sT_f\right)^2} e^{-s\tau}, \tag{103}$$

where T_f in this case represents the time-constant of a second order filter. Using the tuning in (96) for the PID controller, the jitter margin of the system becomes

$$\delta_{\max} = \inf_{\omega \in [0,\infty[} \sqrt{\left(\frac{\tau}{a}\right)^2 + \left(\frac{\omega\tau T_f}{a}\right)^2 + \frac{1}{\omega^2} - \frac{2\tau}{a}\left(\frac{1}{\omega}\sin(\omega\tau) + T_f\cos(\omega\tau)\right)}, \tag{104}$$

which is solved numerically with the process parameters (K_v = 0.015, T_f = 0.05, τ = 0.05) and the jitter margin requirement $\delta_{\max}$ = 0.5. If the original Kao-Lincoln jitter margin condition [36] is used, we obtain a = 0.091, which gives the controller parameters k_p = 121.4, k_i = 0 and k_d = 6.07. If Mirkin's lemma is applied, it allows using higher gain a in the tuning rules (96), so that we obtain a = 0.136 yielding to k_p = 181.5, k_i = 0 and k_d = 9.08. The performance of these two designs is illustrated in Figure 26 (Kao-Lincoln) and Figure 27 (Mirkin), where Simulink simulations of the control system are provided either with a) no packet loss, or b) with maximum packet loss. As expected, the control design based on Mirkin's lemma yields to better reference tracking, but it also appears to be more sensitive to the measurement noise that is present in the simulations.

4.5. Design and Experimenting Tools for PID Control in NCS

This section briefly summarizes the supporting tools for PID tuning and experimenting that are proposed in this thesis. Tuning tools make the application of the controller tuning methods much easier as the method is implemented in the software that anyone can use. This saves the burden of coding the algorithms and verifying their functionality from other users of the methods. In [P1], a software tool, implemented in the MATLAB environment, for PID tuning in NCS is presented. The graphical user interface of the developed tool provides an easy to use interface for controller tuning. The user can specify the process, reference signals, network delay types and/or distributions and disturbances affecting the system.

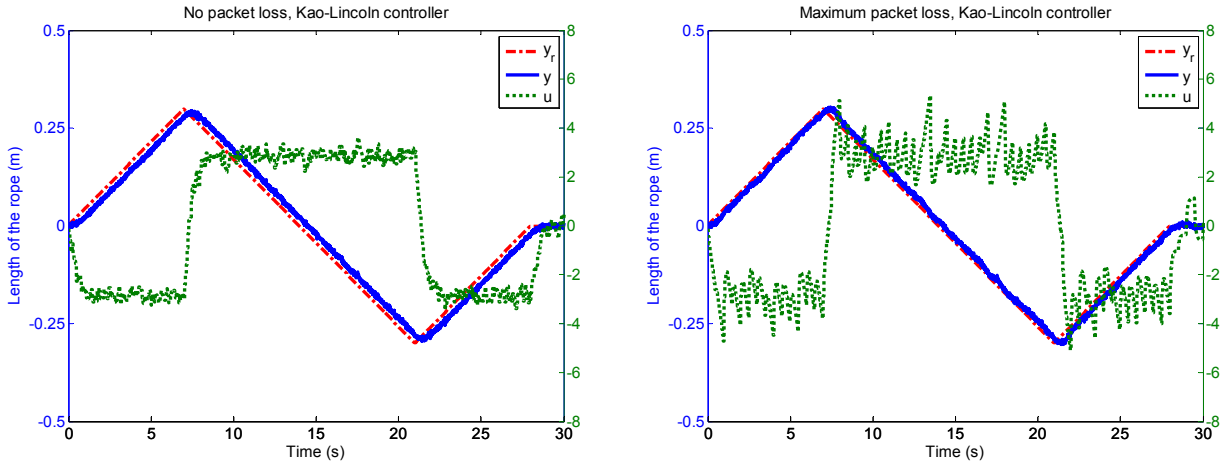

Figure 26. Ramp responses of the DC motor model with a) no packet loss (left), and b) with maximum packet loss (right). Controller parameters according to Kao-Lincoln stability criterion [36].

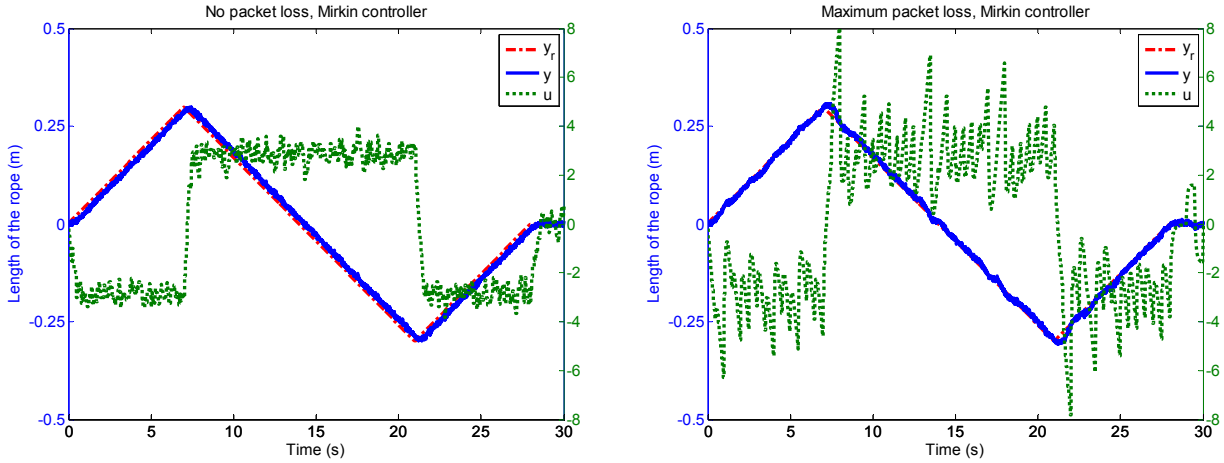

Figure 27. Ramp responses of the DC motor model with a) no packet loss (left), and b) with maximum packet loss (right). Controller parameters according to Mirkin's lemma [49].

The controller tuning can be made manually with sliders or by typing the parameter values into text boxes. Upon parameter change, the tool immediately draws the response of the system into a specified plot. The controller can also be optimally tuned using the tool. The user can specify the desired optimization criterion and the required gain and phase margins, and these are taken into account in optimization as described in [P2]. The tool also provides options for automatic tuning of a batch of processes, and in that case the optimal controller parameters can be analyzed with respect to the controller sample time or process time-constants. The results can be illustrated in several figures and saved on the disk. The tool makes the PID controller tuning for NCS easy and the graphical user interface relaxes the burden of having tens of parameters to be managed in simulations and optimization. The graphical user interface of the tool is illustrated in Figure 28.

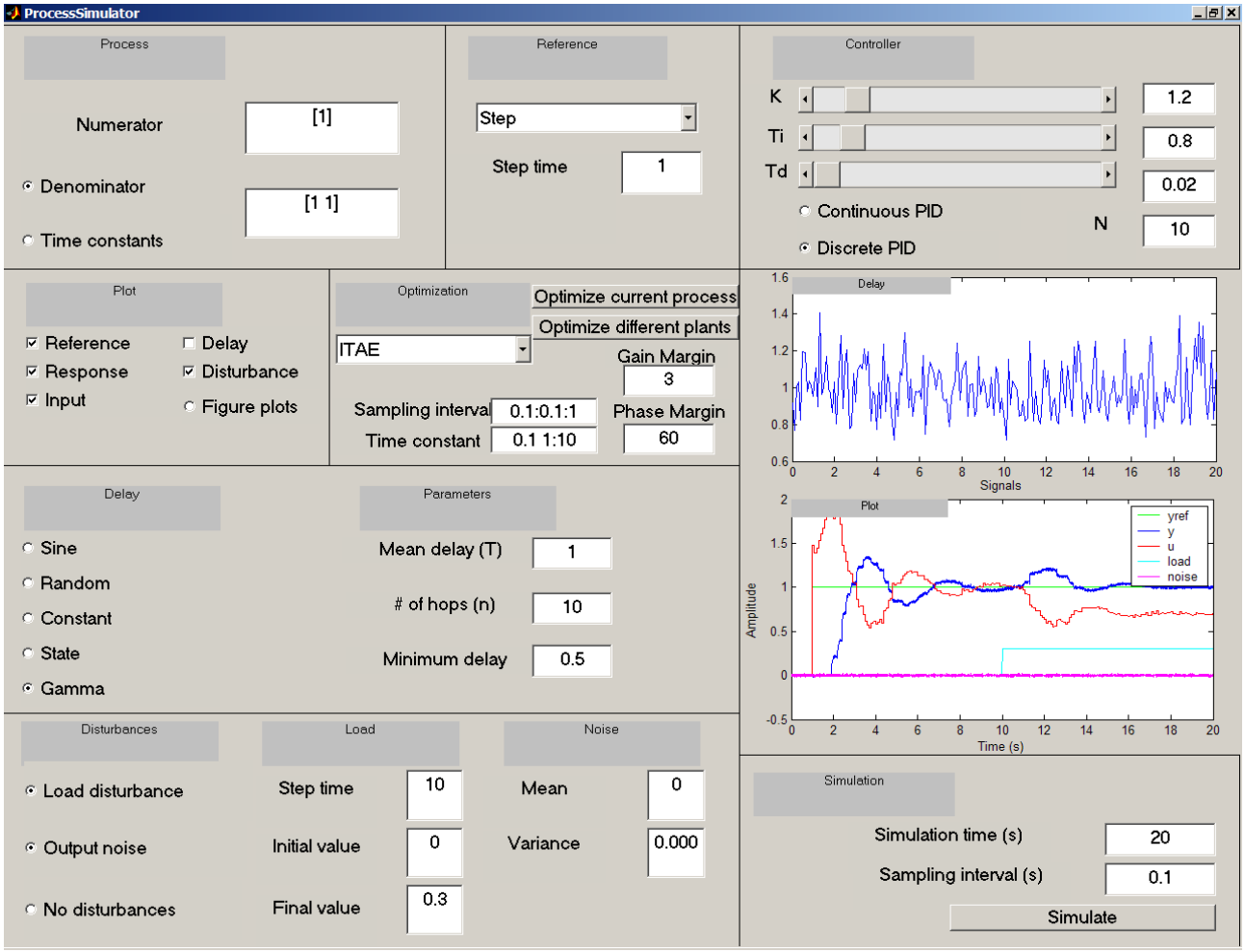

Figure 28. The PID controller tuning tool for NCS [P1].

Once the controller tuning is properly done, the control algorithm should be tested against the simulation model of the plant in a realistic NCS simulation environment. For that purpose, the thesis suggests using the MoCoNet platform [P3] that was already presented in Section 2.4.1. MoCoNet can be used for educational and research purposes in the remote testing of control systems. The MoCoNet platform differs from many remote laboratories in the sense that it is also designed for experimenting with NCS. This is done with a separate network simulator that delays the measurement and control packets according to given network specifications. The packet loss probability can also be adjusted by the user.

One important property implemented in MoCoNet is the ability to integrate the network simulator and the control algorithms with real processes. This is accomplished by implementing the controllers on a RTOS, which can communicate with the I/O board, through which the sensors and actuators are accessed. As the RTOS also supports UDP communications, the measurement and control packets can be directed to the network simulator, which emulates the network behavior by delaying the packets according to specified delay distributions. It can also drop packets randomly. These features make the system asynchronic and to resemble true NCS.

The MoCoNet platform has been extended in many ways after [P3] was published. It now has more properties, but also the network simulation part has been upgraded. Integration with the de facto standard communications simulator ns-2 [56] and experiments over real networks [69] prove the scalability and easy extendibility of the platform. The MoCoNet platform with its support for rapid control prototyping has become an extremely valuable research tool. The results of NCS control design may be easily tested with real processes on the platform.

5. Conclusions

Networked control systems are feedback control systems wherein the control loops are closed through a real-time network. The advantages of networking in control systems are obvious. Networks reduce the cabling costs and provide means for efficient information distribution in the automation system. The fault diagnosis of the system and its components is made easier as the information is available from all parts of the process. Besides the many things NCS provide, there are, however, issues to be taken into account in the control design. Networks induce delays that originate from the shared medium, computations, errors and packet loss in communication. Due to the distributed nature and the network-induced (often variable) time-delays, the NCS are asynchronic systems. The classical control theory assumes uniform sampling with constant sample times, constant time-delays and well-synchronized measurement systems, and the current theory for varying time-delay systems is not in a very mature state. There are, though, quite a few results on control design of NCS available. These include results on control of randomly varying time-delay systems and studies that consider packet dropping links in control systems. Furthermore, there are results on the stability of NCS, where both varying time-delays and packet loss are considered. The control design methodologies in NCS were reviewed in Chapter 2 of the thesis.

Although there are many theoretical results on stability and control design of NCS available, the practical implementation perspective is often missing in the studies. This thesis has taken another approach, and focused on the simple PID control algorithm, the de facto industry standard, and its tuning for NCS. The PID controller is attractive for industrial use, because the operators can understand the algorithm and it is also well justified for NCS, since the computational time and memory requirements are fairly low. These are the prerequisites for any algorithm that is supposed to be run on, for example, wireless sensor and actuator networks with very limited computational power, resources and communication capabilities. The motivation for focusing on the PID controller is that it is already the most common control algorithm in process control applications, and there is no doubt that it will also be that in the wireless automation systems of the future.

The thesis has shown that by using appropriate tuning methods the PID algorithm can be made robust against the deficiencies faced in NCS, such as varying time-delays and packet loss. The related robustness criteria can be adjusted according to the application requirements, for example, by tuning the controller for a specified jitter margin. The main contributions of the thesis are related to PID tuning methods and tuning rules that can be applied in NCS. In addition, the thesis

presented a comparison between the PID controller and other simple, but a little more advanced control algorithms in varying time-delay systems, to find out whether better control can be achieved, for instance, by using the internal model control approach or delay-based fuzzy gain scheduling. The results suggest that improvements are possible, but the difference between the basic PID algorithm and the more advanced controllers vanishes as the time-constant of the process starts to dominate the dynamics and the effect of network-induced delays is decreased.

Besides the tuning methods, the thesis proposed PID tuning rules for stable and integrating processes with varying time-delays. These rules aim at satisfying the given jitter margin, which may be an input parameter to the tuning rules, and maximizing the performance of the control system while providing robustness against disturbances. The proposed rules were compared with other tuning rules by simulations and experiments, and the benefits of using the proposed rules were clearly shown. For integrating processes, the thesis also showed the trade-off between the jitter margin and closed-loop system performance by deriving a formula for the maximum controller gain based on the jitter margin requirement. This formula is very useful in practical applications, where maximum performance is needed and the jitter margin requirement is known. The delay distributions of networks can often be measured in advance, and the control design can be based on these measurements, which give range for the delay values.

Other important aspects that the thesis has brought out are the design tools and environments for NCS. In order to make the control design easy, a PID tuning tool for NCS was proposed. The tool implements one of the tuning methods presented in the thesis. With an easy-to-use graphical user interface built on top of MATLAB, it is easy to tune the controllers either manually or through automated simulation-based optimization. The network properties may be defined in the tool by configuring the delay types and distributions. The tool also considers disturbances and measurement noise. The gain and phase margins can be given for the controller tuning procedure as design parameters, and the optimization algorithm takes these as constraints when solving the tuning problem.

The MoCoNet platform supporting remote experiments with real processes and simulated networks, developed at the Helsinki University of Technology, Finland, was also discussed in the thesis. With this platform, it is possible to implement various controllers with ease using the rapid control prototyping methodology. The platform is especially handy for experimenting control algorithms on real processes in NCS setup. The user of the system may choose from predefined networks or set the network parameters himself including delay distribution and packet loss parameters. The platform operates in a web browser with Java support, and does not require any additional software to be installed on client machines. Since everything is accessible over the Internet, the platform provides excellent opportunities, not only for educational use, but also for world wide research use as the researchers may implement their own control algorithms and test the performance in networked control systems with real processes.

Most of the research presented in this thesis has concentrated on the semi-distributed NCS architecture, which assumes that networks are used only in the sensor-to-controller link. Obviously, the results presented could be modified to support the fully-distributed architecture with networks in both sensor-to-controller and controller-to-actuator links. This would require replacing the jitter margin criterion used in many of the publications and defining suitable stability criteria for the fully-distributed architecture and revising the results from this point of view. Nevertheless, the simple form of the jitter margin criterion has allowed its use in different scenarios and also some analytical calculations have been based on it. Similar multiuse criteria are not generally available for the fully-distributed architecture. It should be emphasized that the simplification of the NCS model by assuming the semi-distributed architecture has also been applied in the literature by other authors. It might be possible to extend the jitter margin criteria for the fully-distributed architecture, but this would presumably result in either more complicated or more conservative criteria. Deeper investigation of these extensions is a good starting point for future research.

Another direction into which the research could be extended is investigating process parameter mapping when approximating a high-order process with the simple first order model with a delay (FOTD). In particular, it would be interesting to see how such an approximation affects the jitter margin if the controller tuning is based on the simplified model. For example, the extended plant approach proposed in this thesis is based on first extending the plant with a measurement filter and then approximating the extended plant with a FOTD model. Assuming a FOTD process model and a first-order measurement filter, it can be shown that if the measurement filter time-constant is significantly larger than that of the process and the proposed tuning rules are applied, the jitter margin of the control system could become relatively small ($\delta_{\max} \approx 3T$, but for small T this is also relatively small). The reason for this is that the jitter margin depends on the ratio of the true time-constant of the process and the time-constant of the approximated FOTD model. Thus the jitter margin depends on how the parameters of the process and the filter map into the FOTD model in the approximation phase, so it is necessary to further investigate the parameter mappings from this perspective. For example, large filter time-constants tend to map into large time-constants in the estimated FOTD model, which might in certain cases prevent achieving high jitter margins with this approach. These findings are based on preliminary investigations of parameter mappings in the FOTD approximations and their effects on the jitter margin, but more exact conclusions would still require further research. In any case, the extended plant approach is a promising method for setting the controller gains based on the desired jitter margin and all future research should take advantage of this approach.

References

[1] I. F. Akyildiz, I. H. Kasimoglu, "Wireless sensor and actor networks: research challenges", *Ad Hoc Networks*, Vol. 2, pp. 351-367, 2004.

[2] A. Al-Hammouri, D. Agrawal, V. Liberatore, H. Al-Omari, Z. Al-Qudah, M. S. Branicky, "A Co-Simulation Platform for Actuator Networks", Demo Abstract, *The 5th ACM Conference on Embedded Networked Sensor Systems*, Sydney, Australia, Nov. 6-9, 2007.

[3] N. B. Almutairi, M.-Y. Chow, Y. Tipsuwan, "Network-Based Controlled DC motor with Fuzzy Compensation", in *Proc. The 27th Annual Conference of the IEEE Industrial Electronics Society*, Vol. 3, pp. 1844-1849, Denver, USA, 2001.

[4] M. Andersson, D. Henriksson, A. Cervin, K.-E. Årzén, "Simulation of Wireless Networked Control Systems", in *Proc. 44th IEEE Conference on Decision and Control, and the European Control Conference 2005*, Seville, Spain, Dec. 2005.

[5] G. Bachman, L. Narici, *Functional analysis*, Academic Press, 1966.

[6] M. S. Branicky, S. M. Phillips, W. Zhang, "Stability of Networked Control Systems: Explicit Analysis of Delay", In *Proc. 2000 American Control Conference*, Chicago, USA, pp. 2352-2357, Jun. 2000.

[7] Y.-Y. Cao, Y.-X. Sun, C. Cheng, "Delay-Dependent Robust Stabilization of Uncertain Systems with Multiple State Delays", *IEEE Transactions on Automatic Control*, Vol. 43, No. 11, pp. 1608-1612, Nov. 1998.

[8] M. Casini, D. Prattichizzo, A. Vicino, "The Automatic Control Telelab", *IEEE Control Systems Magazine*, Vol. 24, No. 1, Jun. 2004.

[9] A. Cervin, B. Lincoln, J. Eker, K.-E. Årzén, G. Buttazzo, "The Jitter Margin and Its Application in the Design of Real-Time Control Systems", in *Proc. 10th International Conference on Real-Time and Embedded Computing Systems and Applications (RTCSA)*, Göteborg, Sweden, Aug. 2004.

[10] G. H. Cohen, G. A. Coon, "Theoretical Consideration of Retarded Control", *Transactions of the ASME*, Vol. 75, pp. 827-834, 1953.

[11] E. B. Dahlin, "Designing and tuning digital controllers", *Instruments and Control Systems*, Vol. 42, pp. 77-83, 1968.

[12] L. Davis, *Handbook of Genetic Algorithms*, Van Nostrand Reinhold Company, 1991.

[13] J. Dorsey, *Continuous and Discrete Control Systems: Modeling, Identification, Design, and Implementation*, McGraw-Hill, 2002.

[14] M. Elmusrati, L. Eriksson, K. Gribanova, M. Pohjola, R. Jäntti, H. Koivo, M. Johansson, J. Zander, "Wireless Automation: Opportunities and Challenges", In *Proc. Automaatio Seminaaripäivät 2007*, Helsinki, Finland, Mar. 27-28, 2007.

[15] L. Eriksson, M. Elmusrati, M. Pohjola (eds.), *Introduction to Wireless Automation*, Helsinki University of Technology, Control Engineering, Report 155, Espoo, Apr. 2008.

[16] L. Eriksson, T. Oksanen, K. Mikkola, "*PID Controller Tuning Rules for Integrating Processes with Varying Time-Delays*", submitted, 2008.

[17] L. Fang, Z. Wu, A. Wu, A. Zheng, "Fuzzy immune Self-regulating PID Control of Networked Control System", in *Proc. International Conference on Computational Intelligence for Modelling Control and Automation, and International Conference on Intelligent Agents, Web Technologies and Internet Commerce*, Sydney, Australia, Nov. 29-Dec. 1, 2006.

[18] E. Fridman, U. Shaked, "Delay-dependent stability and H_∞ control: constant and time-varying delays", *International Journal of Control*, Vol. 76, No. 1, pp. 48-60, 2003.

[19] C. E. Garcia, M. Morari, "Internal Model Control. 1. A Unifying Review and Some New Results", *Industrial & Engineering Chemistry Process Design and Development*, Vol. 21, pp. 308-323, 1982.

[20] K. Gu, V. L. Kharitonov, J. Chen, *Stability of Time-Delay Systems*, Birkhäuser, 2003.

[21] V. Gupta, D. Spanos, B. Hassibi, R. M. Murria, "On LQG Control Across a Stochastic Packet-Dropping Link", in *Proc. 2005 American Control Conference*, pp. 360-365, Portland, USA, Jun. 2005.

[22] F. Göktas, J. M. Smith, R. Bajcsy, "μ-Synthesis For Distributed Control Systems with Network-Induced Delays", in *Proc. 35th Conference on Decision and Control*, pp. 813-814, Kobe, Japan, Dec. 1996.

[23] Y. Halevi, A. Ray, "Integrated communication and control systems: Part I-analysis", *Journal of Dynamic Systems, Measurement and Control*, Vol. 110, pp. 367-373, 1988.

[24] F. Halsall, *Data communications, computer networks and open systems*, 3rd ed., Addison-Wesley Publishing Company, 1992.

[25] V. Hasu, *Radio Resource Management in Wireless Communication: Beamforming, Transmission Power Control, and Rate Allocation*, Doctoral thesis, Department of Automation and Systems Technology, Helsinki University of Technology, Jun. 2007.

[26] T. Henningsson, A. Cervin, "Event-based control over networks: Some research questions and preliminary results", in *Proc. Reglermöte 2006*, Stockholm, Sweden, May 2006.

[27] D. Henriksson, A. Cervin, K.-E. Årzén, "TrueTime: Simulation of Control Loops Under Shared Computer Resources", in *Proc. 15th IFAC World Congress on Automatic Control*, Barcelona, Spain, Jul. 2002.

[28] D. Henriksson, O. Redell, J. El-Khoury, M. Törngren, K.-E. Årzén, *Tools for Real-Time Control Systems Co-Design – A Survey*, Department of Automatic Control, Lund Institute of Technology, Apr. 2005.

[29] K. Hirai, Y. Satoh, "Stability of a system with variable time delay", *IEEE Transactions on Automatic Control*, Vol. AC-25, No. 3, Jun. 1980.

[30] D. Hristu-Varsakelis, P. R. Kumar, "Interrupt-based feedback control over a shared communication medium", in *Proc. 41st IEEE Conference on Decision and Control*, pp. 3223-3228, Las Vegas, USA, Dec. 2002.

[31] D. Hristu-Varsakelis, W. S. Levine, "An Undergraduate Laboratory for Networked Digital Control Systems", *IEEE Control Systems Magazine*, Vol. 25, Issue 1, Feb. 2005.

[32] V. Hölttä, L. Palmroth, L. Eriksson, "Rapid Control Prototyping Tutorial with Application Examples", *Sim-Serv*, 2004, http://www.sim-serv.com/pdf/whitepapers/whitepaper_88.pdf, referred on May 5, 2008.

[33] O. C. Imer, S. Yuksel, T. Basar, "Optimal Control of LTI Systems over Unreliable Communication Links", *Automatica*, Vol. 42, Issue 9, pp. 1429-1439, Sep. 2006.

[34] ISA100, Wireless Systems for Automation, www.isa.org/ISA100/, referred on May 8, 2008.

[35] X. Jiang, Q.-L. Han, X. Yu, "Stability Criteria for Linear Discrete-Time Systems with Interval Like Time-Varying Delay", in *Proc. 2005 American Control Conference*, Portland, USA, Jun. 8-10, 2005.

[36] C.-Y. Kao, B. Lincoln, "Simple stability criteria for systems with time-varying delays", *Automatica*, Vol. 40, pp. 1429-1434, 2004.

[37] H. N. Koivo, A. Reijonen, "Tuning of PID Controllers for Varying Time-Delay Systems", in *Proc. IEEE International Conference on Mechatronics 2004*, Istanbul, 2004.

[38] H. Koivo, P. Virtanen, M. Pusenius, "A Self-Tuning Controller for Processes with Time-Varying Delay: Application to a Paper Machine Head Box", *Preprint of IFAC International Symposium of Adaptive Control of Chemical Processes*, Copenhagen, Denmark, Aug. 1988.

[39] E. Kreyszig, *Advanced engineering mathematics*, 7th ed., John Wiley & Sons Inc., 1993.

[40] K. C. Lee, S. Lee, H. H. Lee, "Implementation and PID tuning of network-based control systems via Profibus polling network", *Computer Standards & Interfaces*, Vol. 26, pp. 229-240, Elsevier, 2004.

[41] S. Li, Z. Wang, Y. Sun, "A Novel Auto-tuning Robust PID Controller for Networked Control System", in *Proc. The 29th Annual Conference of the IEEE Industrial Electronics Society*, Vol. 1, pp. 836-841, Roanoke, USA, Nov. 2-6, 2003.

[42] X. Li, C. E. de Souza, "Criteria for Robust Stability and Stabilization of Uncertain Linear Systems with State Delay", *Automatica*, Vol. 33, No. 9, pp. 1657-1662, 1997.

[43] B. Lincoln, *Dynamic programming and Time-Varying Delay Systems*, Ph.D. dissertation, Lund Institute of Technology, 2003.

[44] B. Lincoln, B. Bernhardsson, "Optimal control over networks with long random delays", in *Proc. International Symposium on Mathematical Theory of Networks and Systems*, Jan. 2000.

[45] B. Lincoln, A. Cervin, "Jitterbug: A Tool for Analysis of Real-Time Control Performance", in *Proc. 41st IEEE Conference on Decision and Control*, Las Vegas, USA, Dec. 2002.

[46] G. P. Liu, J. B. Yang, J. F. Whidborne, *Multiobjective Optimisation and Control*, Research Studies Press Ltd., 2003.

[47] R. Luck, A. Ray, "An observer-based compensator for distributed delays", *Automatica*, Vol. 26, Issue 5, pp. 903-908, 1990.

[48] J. M. Maciejowski, *Predictive Control with Constraints*, Pearson Education Ltd., 2002.

[49] L. Mirkin, "Some Remarks on the Use of Time-Varying Delay to Model Sample-and-Hold Circuits", *IEEE Transactions on Automatic Control*, Vol. 52, Issue 6, pp. 1109-1112, 2007.

[50] Modelica and the Modelica Association, http://www.modelica.org/, referred on May 8, 2008.

[51] L. A. Montestruque, P. Antsaklis, "Stability of Model-Based Networked Control Systems With Time-Varying Transmission Times", *IEEE Transactions on Automatic Control*, Vol. 49, No. 9, pp. 1562-1572, Sep. 2004.

[52] Y. Mostofi, R. Murray, "Effect of Time-Varying Fading Channels on the Control Performance of a Mobile Sensor Node", in *Proc. IEEE 1st International Conference on Sensor and Ad Hoc Communications and Networks*, Santa Clara, USA, Oct. 2004.

[53] R. M. Murray, Z. X. Li, S. S. Sastry, *A Mathematical Introduction to Robotic Manipulation*, CRC Press, 1994.

[54] P. Naghshtabrizi, J. P. Hespanha, "Designing an observer-based controller for a network control system", in *Proc. 44th IEEE Conference on Decision and Control, and the European Control Conference 2005*, Seville, Spain, December 2005.

[55] S. Nethi, C. Gao, R. Jäntti, M. Pohjola, "Localized Multiple Next-hop Routing Protocol", in *Proc. 7th International conference on ITS Telecommunication*, Paris, France, Jun. 2007.

[56] S. Nethi, M. Pohjola, L. Eriksson, R. Jäntti, "Platform for Emulating Networked Control Systems in Laboratory Environments", in *Proc. IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks*, Helsinki, Finland, Jun. 18-21, 2007.

[57] S. Nethi, M. Pohjola, L. Eriksson, R. Jäntti, "Simulation case studies for wireless networked control systems", in *Proc. The 10th ACM/IEEE International Symposium on Modeling, Analysis and Simulation of Wireless and Mobile Systems*, Chania, Greece, Oct. 22-26, 2007.

[58] X. Nian, "Stability of Linear Systems with Time-Varying Delays: An Lyapunov Functional Approach", in *Proc. American Control Conference*, Denver, USA, Jun. 4-6, 2003.

[59] G. Nikolakopoulos, A. Panousopoulou, A. Tzes, J. Lygeros, "Multi-hopping Induced Gain Scheduling for Wireless Networked Controlled Systems", in *Proc. 44th IEEE Conference on Decision and Control, and the European Control Conference 2005*, Seville, Spain, Dec. 2005.

[60] J. Nilsson, *Real-time control systems with delays*, Ph.D. dissertation, Lund Institute of Technology, 1998.

[61] R. Oboe, P. Fiorini, "Issues on Internet-Based Teleoperation", in *Proc. 5th IFAC Symposium on Robot Control*, pp. 611-617, Nantes, France, 1997.

[62] A. O'Dwyer, *PI and PID controller tuning rules for time delay processes: a summary*, Technical Report AOD-00-01, Ed. 1, School of Control Systems and Electrical Engineering, Dublin Institute of Technology, 2000.

[63] A. O'Dwyer, *Handbook of PI and PID Controller Tuning Rules*, Imperial College Press, London, 2003.

[64] T. Olsen, B. Bialkowski, *Lambda Tuning as a Promising Controller Tuning Method for the Refinery*, Presentation at 2002 AIChE Spring National Meeting in New Orleans, 2002.

[65] J. W. Overstreet, A. Tzes, "An Internet-Based Real-Time Control Engineering Laboratory", *IEEE Control Systems Magazine*, Vol. 19, Issue 5, Oct. 1999.

[66] Y.-J. Pan, H. J. Marquez, T. Chen, "Stabilization of remote control systems with unknown time varying delays by LMI techniques", *International Journal of Control*, Vol. 79, No. 7, pp. 752-763, Jul. 2006.

[67] C. E. Perkins, E. M. Royer, *Ad hoc on-demand distance vector (AODV) routing*, tech. rep., IETF Internet Draft, 2001.

[68] C. L. Phillips, R. D. Harbor, *Feedback control systems*, Prentice-Hall International Inc., 1988.

[69] M. Pohjola, *Delay and Control over Internet*, Control Engineering Laboratory, Helsinki University of Technology, Jun. 2006.

[70] M. Pohjola, *PID controller design in networked control systems*, Master's thesis, Helsinki University of Technology, Jan. 2006.

[71] M. Pohjola, S. Nethi, R. Jäntti, "Wireless Control of Mobile Robot Squad with Link Failure", in *Proc. The First Workshop on Wireless Multihop Communications in Networked Robotics*, Berlin, Germany, Apr. 4, 2008.

[72] J. Pulkkinen, H. N. Koivo, K. Mäkelä, "Tuning of a robust PID controller – application to heating process in extruder", in *Proc. Second IEEE Conference on Control Applications*, Vancouver, Canada, Sep. 13-16, 1993.

[73] B. Sinopoli, L. Schenato, M. Franceschetti, K. Poolla, M. I. Jordan, S. S. Sastry, "Kalman Filtering With Intermittent Observations", *IEEE Transactions on Automatic Control*, Vol. 49, No. 9, Sep. 2004.

[74] B. Sinopoli, L. Schenato, M. Franceschetti, K. Poolla, S. Sastry, "An LQG Optimal Linear Controller for Control Systems with Packet Losses", in *Proc. 44th IEEE Conference on Decision and Control, and the European Control Conference 2005*, Seville, Spain, Dec. 2005.

[75] B. Sinopoli, C. Sharp, L. Schenato, S. Schaffert, S. S. Sastry, "Distributed Control Applications Within Sensor Networks", *Proc. IEEE*, Vol. 91, No. 8, Aug. 2003.

[76] J. Song, A. K. Mok, D. Chen, M. Nixon, T. Blevins, W. Wojsznis, *Improving PID Control with Unreliable Communications*, presentation at ISA EXPO 2006, Houston, USA, Oct. 2006.

[77] Y. Tan, "Time-Varying Time-Delay Estimation for Nonlinear Systems Using Neural Networks", *International Journal of Applied Mathematics and Computer Science*, Vol. 14, No. 1, pp. 63-68, 2004.

[78] B. Tavassoli, P. J. Maralani, "Robust Design of Networked Control Systems with Randomly Varying Delays and Packet Losses", in *Proc. 44th IEEE Conference on Decision and Control, and the European Control Conference 2005*, Seville, Spain, Dec. 2005.

[79] The Network Simulator – ns-2, http://www.isi.edu/nsnam/ns/, referred on May 12, 2007.

[80] X. Tian, X. Wang, Y. Cheng, "A Self-tuning Fuzzy Controller for Networked Control Systems", *International Journal of Computer Science and Network Security*, Vol. 7, No. 1, Jan. 2007.

[81] Y. Tipsuwan, M.-Y. Chow, "Network-Based Controller Adaptation Based on QoS Negotiation and Deterioration", in *Proc. The 27th Annual Conference of the IEEE Industrial Electronics Society*, Vol. 3, pp. 1794-1799, Denver, USA, 2001.

[82] Y. Tipsuwan, M.-Y. Chow, "Control methodologies in networked control systems", *Control Engineering Practice*, Vol. 11, pp. 1099-1111, 2003.

[83] Y. Tipsuwan, M.-Y. Chow, "On the Gain Scheduling for Networked PI Controller Over IP Network", *IEEE/ASME Transactions on Mechatronics*, Vol. 9, No. 3, Sep. 2004.

[84] J. G. VanAntwerp, R. D. Braatz, "A tutorial on linear and bilinear matrix inequalities", *Journal of Process Control*, Vol. 10, pp. 363-385, 2000.

[85] E. Verriest, M. Egerstedt, "Control with Delayed and Limited Information: A First Look", in *Proc. 41st IEEE Conference on Decision and Control*, pp. 1231-1236, Las Vegas, USA, Dec. 2002.

[86] P. Viljamaa, *Fuzzy Gain Scheduling and Tuning of Multivariable Fuzzy Control – Methods of Fuzzy Computing in Control Systems*, Doctoral thesis, Tampere University of Technology, Finland, 2000.

[87] G. C. Walsh, O. Beldiman, L. Bushnell, "Asymptotic Behavior of Networked Control Systems", in *Proc. 1999 IEEE International Conference on Control Applications*, Vol. 2, pp. 1448-1453, Kohala, Hawaii, USA, Aug. 1999.

[88] A. Willig, K. Matheus, A. Wolisz, "Wireless Technology in Industrial Networks", *Proc. IEEE*, Vol. 93, No. 6, Jun. 2005.

[89] WirelessHART, http://www.hartcomm2.org/hart_protocol/wireless_hart/wireless_hart_main .html, referred on May 8, 2008.

[90] E. Witrant, P. G. Park, M. Johansson, C. Fischione, K. H. Johansson, "Predictive control over wireless multi-hop networks", in *Proc. IEEE Conference on Control Applications*, Singapore, Oct. 1-3, 2007.

[91] M. Wu, Y. He, J.-H. She, G.-P. Liu, "Delay-dependent criteria for robust stability of time-varying delay systems", *Automatica*, Vol. 40, pp. 1435-1439, 2004.

[92] J. Yen and R. Langari, *Fuzzy Logic – Intelligence, Control, and Information*, Prentice Hall, USA, 1999.

[93] J. K. Yook, D. M. Tilbury, H. S. Wong, N. R. Soparkar, "Trading computation for bandwidth: state estimators for reduced communication in distributed control systems", in *Proc. 2000 Japan-USA Symposium on Flexible Automation*, Ann Arbor, USA, Jul. 2000.

[94] G. Zames, "On the Input-Output Stability of Time-Varying Nonlinear Feedback Systems - Part I: Conditions Derived Using Concepts of Loop gain, Conicity, and Positivity," *IEEE Transactions on Automatic Control*, Vol. AC-11, No. 2, pp. 228-238, 1966.

[95] W. Zhang, *Stability Analysis of Networked Control Systems*, Ph.D. thesis, Department of Electrical Engineering and Computer Science, Case Western Reserve University, Aug. 2001.

[96] W. Zhang, M. S. Branicky, S. M. Phillips, "Stability of Networked Control Systems", *IEEE Control Systems Magazine*, Vol. 21, Issue 1, Feb. 2001.

[97] L. Zhang, D. Hristu-Varsakelis, "LQG Control under Limited Communication", in *Proc. 44th IEEE Conference on Decision and Control, and the European Control Conference 2005*, Seville, Spain, Dec. 2005.

[98] T. Zhang, Y.-C. Li, "A Fuzzy Smith Control of Time-Varying Delay Systems based on Time Delay Identification", in *Proc. Second International Conference on Machine Learning and Cybernetics*, Xi'an, Nov. 2-5, 2003.

[99] X.-M. Zhang, M. Wu, J.-H. She, Y. He, "Delay-dependent stabilization of linear systems with time-varying state and input delays", *Automatica*, Vol. 41, pp. 1405-1412, 2005.

[100] J. G. Ziegler, N. B. Nichols, "Optimum settings for automatic controllers", *Transactions of the ASME*, Vol. 64, pp. 759-768, 1942.

[101] K.-E. Årzén, A Simple Event-Based PID Controller, in *Preprints 14th World Congress of IFAC*, Beijing, China, 1999.

[102] K. J. Åström, T. Hägglund, *PID Controllers: Theory, Design, and Tuning*, 2nd ed., Instrument Society of America, 1995.

[103] K. J. Åström, T. Hägglund, "Revisiting the Ziegler-Nichols step response method for PID control", *Journal of Process Control*, Vol. 14, pp. 635-650, 2004.

[104] K. J. Åström, T. Hägglund, *Advanced PID Control*, Instrument Society of America, 2005.

[105] K. J. Åström, H. Panagopoulos, T. Hägglund, "Design of PI controllers based on non-convex optimization", *Automatica*, Vol. 34, pp. 585-601, May 1998.

[106] K. J. Åström, B. Wittenmark, *Adaptive Control*, Second Edition, Addison-Wesley, 1995.

[107] K. J. Åström, B. Wittenmark, *Computer-controlled Systems – Theory and Design*, 3rd ed., Prentice Hall, 1997.

HELSINKI UNIVERSITY OF TECHNOLOGY CONTROL ENGINEERING

Editor: H. Koivo

Report 145 Hyötyniemi, H. (ed.)
Complex Systems: Science at the Edge of Chaos - Collected papers of the Spring 2003 postgraduate seminar. October 2004.

Report 146 Paanasalo, J.
Modelling and Control of Printing Paper Surface Winding. June 2005.

Report 147 Mohamed, F.
Microgrid Modelling and Simulation. March 2006.

Report 148 Mäenpää, T.
Robust Model Predictive Control for Cross-Directional Processes. May 2006.

Report 149 Kantola, K.
Modelling, Estimation and Control of Electroless Nickel Plating Process of Printed Circuit Board Manufacturing. March 2006.

Report 150 Virtanen, T.
Fault Diagnostics and Vibration Control of Paper Winders. June 2006.

Report 151 Hyötyniemi, H.
Neocybernetics in Biological Systems. August 2006.

Report 152 Hasu, V.
Radio Resource Management in Wireless Communication: Beamforming, Transmission Power Control, and Rate Allocation. June 2007.

Report 153 Hrb•ek, J.
Active Control of Rotor Vibration by Model Predictive Control - A simulation study. May 2007.

Report 154 Mohamed, F. A.
Microgrid Modelling and Online Management. January 2008.

Report 155 Eriksson, L., Elmusrati, M., Pohjola, M. (eds.)
Introduction to Wireless Automation - Collected papers of the spring 2007 postgraduate seminar. April 2008.

Report 156 Korkiakoski, V.
Improving the Performance of Adaptive Optics Systems with Optimized Control Methods. April 2008.

Report 157 Al.Towati, A.
Dynamic Analysis and QFT-Based Robust Control Design of Switched-Mode Power Converters. September 2008.

Report 158 Eriksson, L.
PID Controller Design and Tuning in Networked Control Systems. October 2008.

ISBN 978-951-22-9633-0

ISSN 0356-0872

Yliopistopaino, Helsinki 2008