University of Alberta

Multirate Control Over Networks

by

Amir Kamal Dehghani Mohammad Abadi ©

A thesis submitted to the Faculty of Graduate Studies and Research in partial fulfillment of the requirements for the degree of Master of Science.

Department of Electrical and Computer Engineering

Edmonton, Alberta
Spring 2005

University of Alberta

Library Release Form

Name of Author: Amir Kamal Dehghani Mohammad Abadi

Title of Thesis: Multirate Control Over Networks

Degree: Master of Science

Year this Degree Granted: 2005

Amir Kamal Dehghani Mohammad Abadi
53 Golestan 3 St.,
Farahzadi Ave., Tehran,
Iran

Date: Nov. 9, 2004

Abstract

This thesis studies design of Network Control Systems, focusing on the design of the network along with the controller to increase the closed loop performance in the presence of the capacity constraints of communication channel. In the literature, optimal network resource allocation has been used to to improve the system performance. In the first section we use the optimal allocation along with the design of the associated controller to reduce the network traffic. The resource allocation problem is a convex optimization problem, which is computationally tractable. The numerical results show a significant improvement in the system performance compared to the uniform resource allocation.

Observing that not all of the control signals need to be transmitted at the same rate, we divided the control signals into two categories: fast-rate signals and slow-rate signals. We designed a dual-rate $\mathcal{H}_2$-optimal controller while the time-slots were optimally allocated. The achieved results motivated us for a more general case. We suggest an algorithm to further decreasing the network traffic; a combination of a multirate controller together with optimal resource allocation is proposed. Simulation results verify effectiveness of the suggested algorithm.

Acknowledgements

This thesis would not transpire to exist without the help, support and love of a number of people to whom I will always be indebted.

First and foremost, I would like to express my sincere gratitude to my supervisor Professor Tongwen Chen for his valuable guidance, encouragement, technical advice and financial support. I thank Professor Fraser Forbes for his advice on optimization algorithms.

I thank Hamid Reza Saligheh Rad for his friendship and support over many years. I thank Mehrdad Sahebsara, Amir Masoud Rabiei, Iman Izadi, Sasan Haghani and their families for their friendship and support that made my stay at Edmonton so enjoyable.

Last, but certainly not least, I wish to thank my dearest parents and family, who I am indebted to them for their unconditional supports and endless love.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Closing control loops over communication networks appeared as a consequence of the developments in the microelectronic and the telecommunication areas. The idea introduced a new terminology, Networked Control Systems or NCS. Compared to traditional control systems, it suggests several advantages such as lower cost, higher flexibility, easier maintenance and higher compactness. Therefore, it suits numerous applications such as process and automotive industries, aircrafts and spacecrafts. Although NCS introduces several advantages, it initiates some new challenges too. In summary, inserting a network in a previously-designed system, affects the system performance and can destabilize the system. Therefore, the system requires to be re-designed considering the network presence. This chapter briefly reviews the control system implementation schemes, introduces the status of the research and presents an outline for the remainder of the thesis.

1.1 History of Control System Implementation

In the course of the developments in the control system implementation technology, there exist several turning points. But from information transmission point of view, it is mainly divided into two eras, namely traditional implementation and network-based implementation. In this section, we briefly introduce the aforementioned categories and compare their advantages and disadvantages.

1.1.1 Traditional Control Systems

Directly connecting control elements is the traditional way of implementing control systems. Figure 1.1 depicts a traditional control system in general setup, where sensor information is transmitted to the controller by individual connections. Similarly, the controller sends the control sequences via individual connections. Earlier, these peer-to-peer connections were pneumatic links, however, hard-wired electrical connections replaced them later. In this setup, control elements such as sensors, controllers and actuators are interconnected by means of individual wirings. Several advantages prompted its vast utilization in the industry.

- First of all, its theories are well established [10]; it is assumed that the measurements are transmitted almost perfectly, the classical control results suit the applications.
- Moreover, they are well integrated, i.e. components from different manufacturers could be integrated to maintain the system functionality. This property allows the system integrators to have a wider range of options to choose the required equipments.
- Furthermore, it is easy to use and in the case of emergency , the operator can override the equipment and operate the system manually.

The benefits of the traditional control systems are unquestionable, however, today's competitive market demands more features. Revisiting Figure 1.1, one observes that establishing each connection needs an individual wiring, i.e. the system requires a high amount of wirings. These wirings are huge, time-consuming and more importantly, expensive to install and to maintain. Moreover, modern production platforms

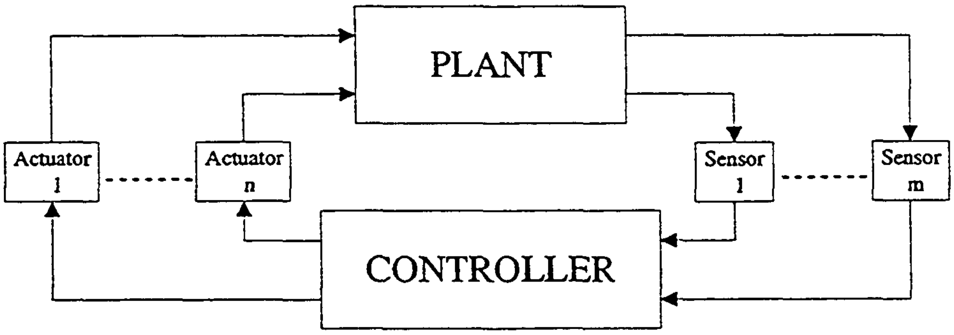

Figure 1.1: Traditional control system.

require the ability of producing different products using the same facilities. To achieve this goal, the platform must be easily and quickly reconfigurable. However, one observes that any reconfiguration or expansion in this setup will need new wirings that decreases the flexibility. The aforementioned drawbacks raise a question: Is there another technology to implement the control system and remove these restrictions? Network-based control systems appeared to be the answer.

1.1.2 Network-Based Control Systems

Transmitting control signals over a shared medium appeared as an aftereffect of the recent advances in the computer networks. Advances in the microelectronics allowed producing control equipments with built-in network interfaces; the new control elements resemble the network users. These elements are interconnected over an industrial network such as Fieldbus Foundation®, Profibus®, CANbus®, etc. It is noteworthy that these networks can be interfaced with one another. Although each network suits a specific range of applications, they introduce comparable advantages. Figure 1.1.2 shows an NCS in general setup. Observe that unlike the traditional control systems, Figure 1.1, the controller receives all sensors information using a single communication link. Moreover, control sequences are transmitted using the similar connection. In addition to the control information, other traffic such as diagnostic data and managerial information is also transmitted over this communication link.

The immediate consequence of using the NCS is a significant decrease in the required amount of the wirings that facilitates and expedites the installation and the maintenance at a significantly lower cost. Moreover, the system offers higher flexibility, i.e. the system can be reconfigured more easily or updated without interrupting the system. This added flexibility allows the manufacturers to produce several products using the same production line or to quickly reconfiguring the system to manufacture new products. Last but not least, NCS allows collecting more information from the plant that contributes to the system maintenance, production inventory and management. Although NCS offers irrefutable benefits and abilities to the control system, further examinations show some drawbacks.

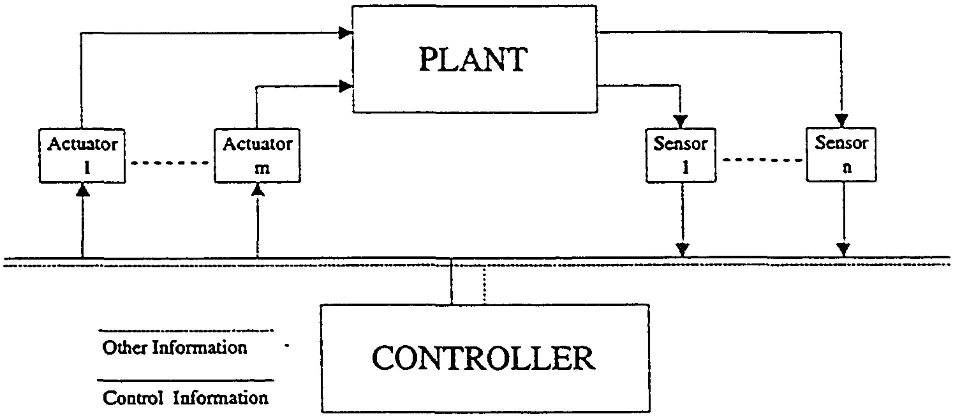

Figure 1.2: Networked control system.

Transmitting information through the network induces some delays. In specific, these delays are *the processing*, *the propagation* and *the queuing delays* that degrade the NCS performance or may cause instability [12]. The data requires to be processed before the transmission, i.e. the data is split up into several packets. To ensure that the data is correctly transmitted and delivered to the recipient, other information such as the recipient address and redundancy information should be added to the data packets; this information must be properly removed at the recipient side. The required time for the processing operation addresses the processing delay that is constant by its nature. Depending on the physical length of the network and the type of the media that is being used to propagate the physical signals, e.g. optical fiber, the signal requires a certain amount of time to arrive at the recipient side. This is the so-called propagation delay that is constant. Because the communication medium is shared among several users, all users post their data packets in a unique queue, where the network protocol determines the management policy for the queue.

Observe that the first-two delays are constant and once the network is set up they will not change. However, queuing delay can be either time-invariant or time-variant in a deterministic or stochastic fashion. Another issue that can contribute to the delay is the network sampling time. Consider that as the network sampling time increases, more data is queued, i.e. queuing delay increases. Another issue in NCS is the packet loss, i.e. while a data packet is being transmitted over the network it might be lost. In fact, because a data packet is a sequence of binary signals, some of the signal values may change because of the noise interference or collision with the other packets and the changes can not be recovered, i.e. the data packet is lost. Moreover, because the data is split into several packets, they may be routed through different paths. Therefore, the recipient user may not receive them at the order that they have been sent. This phenomenon addresses packet-mismatch. Capacity constraint is another obstacle. Observe that the users share limited amount of resources among each other. Next section presents the specific problem that we are interested in.

1.2 The Problem

In the previous section some of the major drawbacks associated with the network-based control systems were addressed. However, in this work, we focus on the effects of the network resource restrictions. Observe that regardless of the network protocol or the physical medium that is being used, network resources are limited. For example, amount of bit rate[1] that is available to the network is limited. Because in the network environment each user obtains a fraction of the available resources, resource allocation has a substantial impact on the network performance and in turn on the control system performance. In [38], the suggested optimal resource allocation policy improves the NCS performance.

Although carefully allotted resources to the network users ameliorate~~s~~ the system performance, but these improvements are restricted by the total amount of the available resources. This restriction raises a question: Can we further improve the system performance at the presence of the optimal resource allocation? With fixed amounts of resources, the only alternative to improve the system performance is to decrease the network traffic. Because if we attempt to transmit the control information at a higher rate, to increase the performance, the network capacity constraints cause the performance to reach a maximum. Transmitting information at a higher rate than the maximum, reduces the performance. Apparently, to reduce the traffic, one needs to reduce the amount of information that is acquired from the plant or sent to the plant. However, along with the concession in the resource allocation, there exist another trade-off: The more traffic we reduce, the less quality information we receive. Therefore, the method to decrease the traffic should be chosen carefully. We propose a multirate sampling scheme to decrease data transmissions.

[1] Bit rate is a measure of how many bits of information are passed across the communication channel in the unit time; units are bps, Kbps, etc.

1.3 Thesis Scope

In this thesis we address problem of the optimal resource allocation combined [38] with a new method to decrease the network traffic. The presented network resource allocation is a convex optimization problem, which is solved numerically. Effect of the resource constraints is modelled as the communication noise. Simulation results show effectiveness of the optimal resource allocation.

To further improve the network utilization, we suggest to design an optimal multirate controller for the plant. We present an algorithm to design the network and the controller parameters to improve the NCS performance. Numerical examples demonstrate the effectiveness of the proposed algorithm.

1.4 Thesis Organization

The outline of this thesis is as follows. Chapter 2 presents a background to NCS. It discusses proposed analysis and design methods along with the current status of the research. It reviews methodologies in three categories, namely network-design methods, controller-design methods and joint-design methods. Chapter 3 presents simultaneous design of the network and the controller parameters. The NCS design is suggested as a multi-objective optimization problem in which the objective is to minimize the system output variance. The decision values are the controller and the network parameters. Some technical issues are collected here. Chapter 4 presents the thesis contribution. The proposed design algorithm is presented in this chapter. To decrease the network traffic load we suggest to design an $\mathcal{H}_2$-optimal multirate controller. We give an algorithm to design the controller that is subject to the causality constraint. Numerical examples illustrate effectiveness of the method. A summary of the thesis and its contribution are given in Chapter 5.

Chapter 2

Background and Literature Review

Network control systems challenges including network delay, packet loss, packet mismatch and miss-synchronization have been addressed in several pieces of research. However, from the point of view of designing parameters, we may divide them into three categories: network-based design methods, controller-based design methods and Joint design methods. In this chapter we review some of the featured works done in each above mentioned categories.

To achieve a better understanding of the NCS design and analysis problem we review network protocols briefly; the first section is devoted to this issue. In section 2, we review some of the controller-based NCS designs. The last section represents the network-based methodologies.

2.1 Network Protocols

Unlike data networks, which deliver large data packets with a sporadic nature of the transfer load, control networks transfer small-sized information packets that must be delivered in a timely manner. In other words, the most important feature of control networks compared with data networks is that it is time-critical, i.e. data must be delivered in a timely fashion.

2.1.1 Token-Passing Networks

One way to guarantee the on-time arrival of the control-related information is to set up a token; the token indicates the possession of the communication bus and rejects any other simultaneous access request. In other words, token's owner is guaranteed to transmit its information freely, without any congestion, in certain duration of the time over the network. Network protocols IEEE 802.4 and IEEE 802.5 have been built on this concept. Regardless of the physical arrangement of the nodes, network clients, each of them are placed into a queue, and access the network one-by-one and consecutively. Figure 2.1 shows two examples of the different topologies that could be used to deploy the clients. In the bus topology, each user connects to the bus, which is shared among all users. Unlike bus topology, in the star topology all users have a common node. Real-world networks combine different topologies to benefit their advantages and to reduce the disadvantages. Although the token mechanism controls the network function regardless of its physical arrangement, each topology has its own advantages and disadvantages [9].

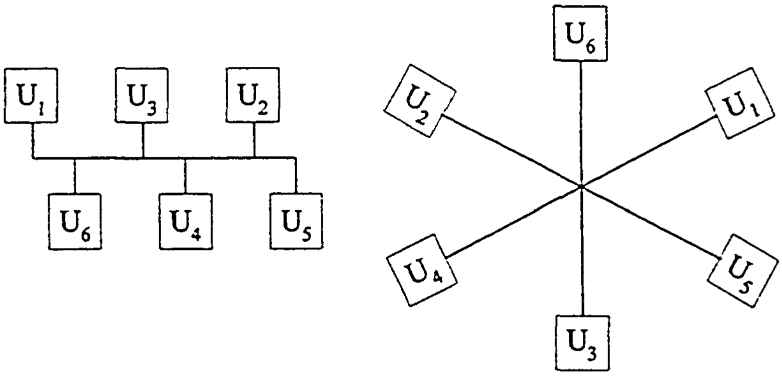

Figure 2.1: Different Topologies, BUS and STAR.

PROFIBUS

PROFIBUS was developed by a group of German companies and is now a German standard. The information is transmitted at different speeds, from 9.6 Kbps up to 500 Kbps. The maximum length of the bus is 1200 m, on which up to 127 stations can be connected. PROFIBUS is a token-passing network. The nodes are divided into *active* and *passive* nodes. The node which holds the token has the permission to send data on the network. The token is passed around in the network between the active nodes. Active nodes can transmit when they hold the token. However, passive nodes are not able to transmit unless they are called by an active node to send data on the network. For example, actuator could be an passive link while the controller is an active node.

Foundation Fieldbus

Foundation Fieldbus was developed by Fieldbus Foundation organization; the foundation is a not-for-profit corporation that consists of nearly 120 of the world's leading suppliers and end users of process control and manufacturing automation products. Based on the bus speed, there are two fieldbuses. The low speed bus, H1, has been designed to use in the field, i.e. sensors, controllers and actuators are connected via this link. The maximum number of the instruments that can be connected to the H1 is 32 at transmission rate of 31.25 Kbps. However, this may be limited to a lower number, based on the required 'supply current' of the instrument. High Speed Ethernet, HSE, is the foundation fieldbus backbone. It connects host computers at 2.5 Mbps. The bus access control is done through Link Active Scheduler, LAS. LAS is a deterministic, centralized bus scheduler that maintains a list of transmission times for all data buffers in all devices that need to be cyclically transmitted. Only one device on an H1 fieldbus, Link Master or LM, can be functioning as that link's LAS.

At the system start-up each node informs LAS its required information and the time to access the network. During the system run time, LAS broadcasts a message over the link and every node receives the schedule; the scheduled node broadcasts its information and all nodes will receive it. There is also a spare time for unscheduled messages. Therefore each link is able to transmit its information in a deterministic fashion.

2.1.2 Congestion Control Methods

The congestion control method is another way to control the network access. In this method, several users compete to send their information simultaneously; a congestion-control mechanism will assign which node possesses the network. Ethernet (IEEE 802.3) includes a group of protocols that employ congestion control mechanism. They provide transmission rates from 1 Mbps to 1 Gbps. Ethernet-based protocols have been employed widely both in the commercial and industrial applications. Although they have not been intended to deploy in the real-time applications, a large number of installations has made this group an attractive choice to many system designers.

Ethernet is controlled by a distributed mechanism to control bus access, which is called CSMA/CD, Carrier Sense Multiple Access with Collision Detection. Each node before sending its information verifies bus status to be idle, i.e., it listens to the bus. As soon as the bus is idle, it transmits its information over the bus. However, if several stations transfer their information simultaneously a collision occurs. In this case the detection mechanism informs the collision to clients. Each client waits for a random time; this time is calculated individually for each of them. After waiting, they begin to retransmit the information. If a node was unable to transmit its information after 16 trials, an error message will be produced. An almost unlimited number of the clients can connect via an Ethernet network, though, the possibility of the network transmission decreases while unscheduled messages can be sent during the allocated spare time.

CAN

The Controller Area Network (CAN), was developed by Robert Bosch GmbH [1]. It was developed mainly for automotive industries. Although CAN was one of the first fieldbuses in the market, it is used widely in the automobile industries. It can be programmed in different speeds. Depending on the bus length, it may be programmed for 1 Mbps for the lengths less than 50 meters and 500 Kbps for the bus length over 50 meters. Unlimited number of users can connect to the bus. Every node can start transmitting its information at any time. If a congestion happens, an arbitration mechanism begins. Since each node has a priority number, the node with the highest priority wins the competition and retransmits its information. In this way, it is almost guaranteed that a high priority node will access the communication channel.

In the next section we review some of the controller design methods for the NCS problem.

2.2 Controller Design Methods

Controller-design methods refer the research that analyzes or designs controllers for an NCS assuming that the network has been already designed.

Controller-design based methods can be traced back to 1970's. Isle in [13] stated the problem of the miss-synchronization between the sampling of the plant states and arrival time of this required information to the controller. This phenomenon represents a randomly time-varying delay in the outer loop.

Furthermore, because a central computer is used to compute the control sequences, these signals are put into a buffer. This latter increases the delay. Figure 2.2 shows

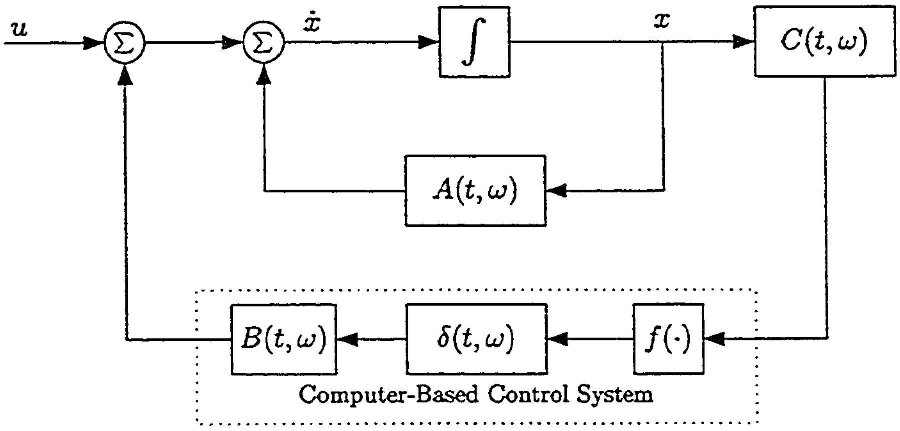

Figure 2.2: Computer-Based Control System.

the considered setup. The inner loop shows the plant dynamic. The outer loop shows the computer-based control, where $\delta(t,\omega)$ represent the induced delay, $f(\cdot)$ models the D/A converter nonlinearities, $A(t,\omega)$, $B(t,\omega)$ and $C(t,\omega)$ represent system dynamic matrices and ω belongs to the probability set Ω.

Definition 2.2.1 *Almost-sure stability: A stochastic discrete-time system,*

$$x(k+1) = Ax(k) + Bu(k),$$

is said to be almost-surely stable, if for every initial states and time (x_0, r_0) *we have*

$$P\{\lim_{k\to\infty} \mid x_k(x_0, r_0) \mid = 0\} = 1. \tag{2.1}$$

Isle shows the almost-sure stability is maintained. Although his work does not directly address the NCS problem, the investigated problem carries a similar concept: The considered delays resemble the network delays. In the remainder of this section, we review several controller-based analysis and design methods.

2.2.1 State and Output Feedback

Luck *et al.* [22] considered the NCS problem with the setup shown in Figure 2.3. The sensor information and control sequences are transmitted over the network. The sensor-controller and the controller-actuator delays are shown by δ_{sc} and δ_{ca}, respectively; both are randomly changing.

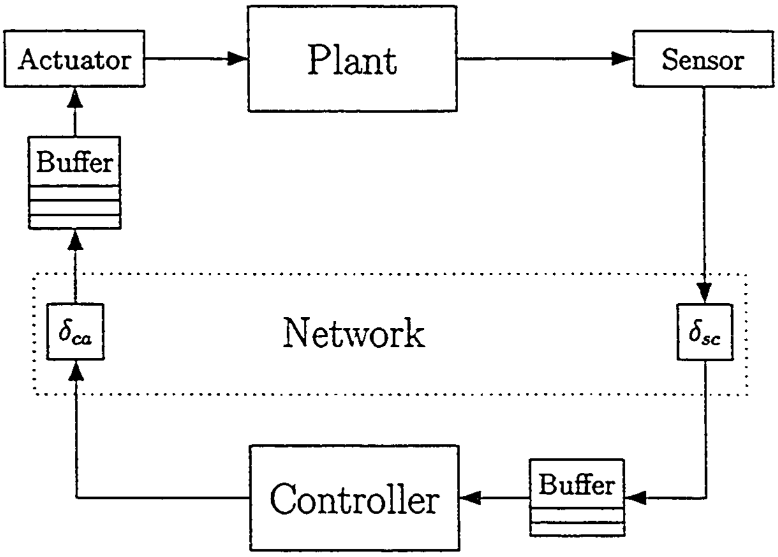

Figure 2.3: Buffered Time-Invariant NCS.

Because the delays are randomly changing in the system, the system is time-varying, which makes design and analysis of the system harder. To remove this obstacle, they added two buffers before each recipient node, i.e., the buffers interface actuator-network and sensor-network connections. These buffers are longer than the worst-case delays. Suppose Δ_1 and Δ_2 are the buffer lengths of the actuator and the controller nodes, then the system equations can be written:

$$
\begin{aligned}
x(k+1) &= Ax(k) + Bu(k-\Delta_1), && (2.2)\\
y(k) &= Cx(k), && (2.3)\\
\omega(k) &= y(k-\Delta_2), && (2.4)
\end{aligned}
$$

where $\omega(k)$ is the process output available to the controller. The reformulated problem, controller design for the system stated in (2.2) and (2.3), is an standard sampled-data control problem, where an LQG-optimal controller has been designed for. Although this method can handle delays longer than the sampling period, but it has a major disadvantage. Revisiting Figure 2.3, it is observed that even if the new data is available, the old data would be still used, i.e. the delays are longer than necessary.

In Liou and Ray [20, 21] a stochastic approach is considered. The sensors and the controllers are sampled at identical sampling rates, T_s, with a time-skew Δ_s. The time-skew, Δ_s, is a slowly time-varying parameter, which is periodically adjusted. Therefore it may be assumed as a selectable constant parameter. Figure 2.4 shows the investigated setup. The sensor-controller and controller-actuator delays are indicated by δ_{sc} and δ_{ca}, which are random variables. The current controller output value is u_k. The probability that the new sensor value has reached the controller when the control output, $u(k)$, is computed is $P(\delta_{sc}^k < \Delta_s)$ and is assumed to be known. However, if the controller does not receive the new sensor information, the control signal is calculated without knowledge of the new measurement signal. The actuator D/A converter converts the new control signal as soon as it receives the data, i.e. the actuator is event-driven.

An augmented state-space model given in (2.5) is associated with the system that takes the delay into account:

$$
x(k+1) = A(k)x(k) + B(k)u(k), \qquad (2.5)
$$

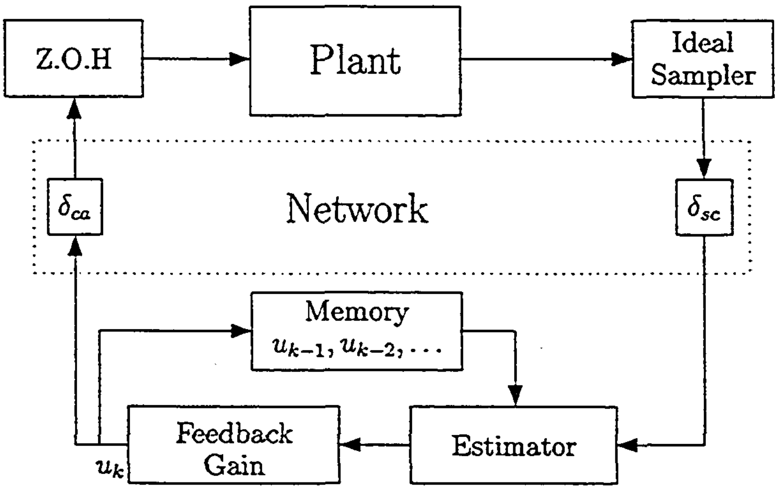

Figure 2.4: Time-Varying NCS with a Constant Time-Skew.

where $A(k)$ and $B(k)$ are stochastic matrices due to random communication delays. The LQ-optimal controller is derived for the system shown in (2.5). A state estimator is suggested in the case where all states are not available to measure. To achieve a more accurate estimation, it is suggested to time-stamp measurements. However, combination of the state estimator and the LQ-optimal controller is a suboptimal controller. Furthermore achieved controller does not guarantee an almost-sure performance. A similar setup is suggested in [28] with a stochastic state estimator. The estimator is to minimize the prediction error variance.

2.2.2 Jump Linear Systems

Jump linear systems in discrete-time can be written as

$$x(k+1) = A(r_k)x(k) + B(r_k)u(k), \tag{2.6}$$

where $A(r_k)$ and $B(r_k)$ are real-valued matrix functions of the random process $\{r_k\}$ [6]. Assuming r_k to be a time homogenous Markov chain taking values in the finite set $\{1, 2, \ldots, n\}$. The Markov chain has the transition probabilities:

$$P(r_{k+1} = j \mid r_k = i) = q_{ij} \geq 0, \text{ where} \sum_{j=1}^{n} q_{ij} = 1, \tag{2.7}$$

with the state transition matrix:

$$Q = \begin{bmatrix} q_{11} & q_{12} & \cdots & q_{1n} \\ q_{21} & q_{22} & \cdots & q_{2n} \\ \vdots & \vdots & & \vdots \\ q_{n1} & q_{n2} & \cdots & q_{nn} \end{bmatrix}. \tag{2.8}$$

Different notations of stability have been considered for jump linear systems. Ji *et al.* [14] presented the following definitions of three different stability notions. For the system stated in (2.6) with the inputs identically equal to zero the equilibrium point 0 is:

Definition 2.2.2 *Stochastically Stable if for every initial state* (x_0, r_0)

$$E\left\{\sum_{k=0}^{\infty} |x_k(x_0, r_0)|^2 \middle| x_0, r_0\right\} < \infty, \tag{2.9}$$

Definition 2.2.3 *Mean Square Stable if for every initial state* (x_0, r_0)

$$\lim_{k \to \infty} E\left\{|x_k(x_0, r_0)|^2 \middle| x_0, r_0\right\} = 0, \tag{2.10}$$

and

Definition 2.2.4 *Exponentially Mean Square Stable if for every initial state* (x_0, r_0)*, there exist constants* $0 < \alpha < 1$ *and* $\beta > 0$ *such that for all* $k \geq 0$

$$E\left\{|x_k(x_0, r_0)|^2 \middle| x_0, r_0\right\} \leq \beta \alpha^k |x_0|^2. \tag{2.11}$$

The authors in [14] showed that the three stability definitions, stated in (2.9), (2.10) and (2.11), are equivalent.

Revisiting *Almost Sure Stability* in (2.1), a less conservative stability notion, for the jump linear systems described by (2.6) and (2.7), with the inputs identically equal to zero, the equilibrium point 0 is:

Definition 2.2.5 *Stochastically Stable if for every initial state* (x_0, r_0)

$$P\left\{\lim_{k\to\infty} |x_k(x_0, r_0)| = 0\right\} = 1. \tag{2.12}$$

The stability concepts stated in (2.9), (2.10) and (2.11) imply almost sure stability, but almost sure stability does not imply them. An example clarifies this concept.

Consider a Markovian jump linear system with two states with a state transition matrix:

$$Q_a = \begin{bmatrix} 1-\alpha & \alpha \\ 0 & 1 \end{bmatrix}, \alpha \leq \frac{1}{2}. \tag{2.13}$$

The system has the property that once it jumps into absorbing state 2, it remains in there forever. System matrices for these two states are given in:

$$A(1) = 2 \quad , \quad B(1) = 0, \tag{2.14}$$

$$A(2) = 0 \quad , \quad B(2) = 0. \tag{2.15}$$

The state x_k grows exponentially as long as it stays in absorbing state 1, i.e. the system is determined by system matrices in 2.14. Once it jumps into absorbing state 2, x_k is reset to zero, while the system stays in this absorbing state forever.

$$\lim_{k\to\infty} E\left\{|x_k(x_0, y_0)|^2 \middle| x_0, r_0 = 1\right\} = \lim_{k\to\infty} (1-\alpha)^k (2)^k x_0 = \infty, \tag{2.16}$$

which shows that the system is not mean square stable. However, we have

$$P\left\{\lim_{k\to\infty} |x_k(x_0, r_0)|\right\} = 0, \tag{2.17}$$

which shows the system is almost surely stable.

Chan and Özgüner in [5] modelled the NCS with upper-bounded delay as a jump linear system. Because the delay has a bionomial distribution, it is assumed that the random parameters are governed by a Markov chain [9]. They assume queue, with length m, which always keeps the latest m samples. However, if the queue is full and no sample has been transmitted then the new data replaces the old data. This mechanism guarantees that always newest data is used. Figure 2.5 shows the considered configuration.

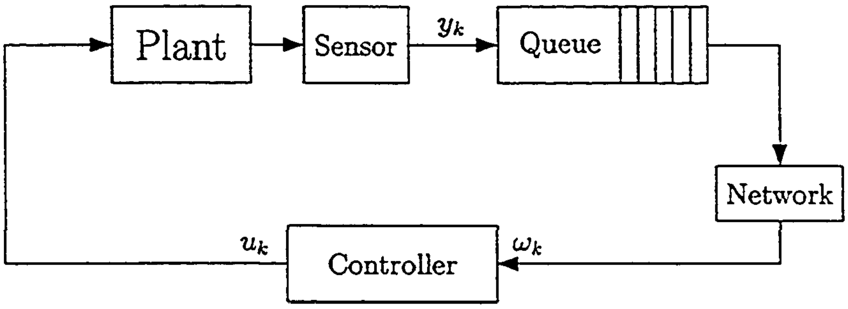

Figure 2.5: Block Diagram of the Controlled System Involving Queueand Network Block.

The random delay has been taken into account by augmenting the states:

$$z_k = \left[\begin{array}{cccccc} x_k^T & x_{k-1}^T & x_{k-2}^T & \cdots & x_{k-m}^T & \omega_{k-1}^T \end{array}\right]^T. \tag{2.18}$$

The system equation is also given by

$$z_{k+1} = \begin{bmatrix} A & 0 & \cdots & 0 & 0 & 0 \\ I & 0 & \cdots & 0 & 0 & 0 \\ 0 & I & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & 0 & \cdots & I & 0 & 0 \\ a_1 & a_2 & \cdots & a_m & a_{m+1} & a_{m+2} \end{bmatrix} z_k + \begin{bmatrix} B \\ 0 \\ 0 \\ \vdots \\ 0 \\ 0 \end{bmatrix} u_k, \tag{2.19}$$

$$\omega = \left[\begin{array}{cccccc} c_1 & c_2 & \cdots & c_m & c_{m+1} & c_{m+2} \end{array}\right] z_k. \tag{2.20}$$

Coefficients $a_1, a_2, \cdots, a_{m+2}$ and $c_1, c_2, \cdots, c_{m+2}$ are determined based on the delays as follows: for $1 \leq i \leq m+1$

$$a_i \text{ and } c_i = \begin{cases} C, & \text{if there is a } (i-1)\text{-time-step delay,} \\ 0, & \text{otherwise,} \end{cases}$$

and

$$a_{m+2} \text{ and } c_{m+2} = \begin{cases} I, & \text{if there is no data received within one sampling} \\ & \text{interval, } (\omega_k = \omega_{k-1}). \\ 0, & \text{otherwise.} \end{cases}$$

One observes that depending on the delay, the jump parameter (r_k), the system has $m+2$ modes. For example $r_k = 1$ indicates no-delay mode case and $r_k = 2$ represents one-sample delay case. To achieve the mode-independent optimal controller, the following performance measure is minimized:

$$J = E\left\{\sum_{k=0}^{N-1} [u_k' R u_k + z_k' Q z_k] + z_N' \tilde{Q}(r_N) z_N\right\}. \tag{2.21}$$

The achieved controller is mode-independent. However, this leads to a more conservative solution. A switching controller yields a better answer, i.e. a control law is associated with each delay case. So, depending on the delay, control laws are switched while the system is running. However, it is necessary to time-stamp the samples for the controller to determine the delay.

2.2.3 Inferential Control

Zhivoglyadov and Middleton in [40] posed NCS design problem as an inferential control design problem. They considered the communication delay and data loss issues. The model was assumed to be given. Then a switching mechanism determines to update the data from model output, in the case that data was not transmitted in a timely manner. They also showed the closed-loop system achieves robust stability.

Alberto and Crespo in [2] considered problem of real-time control of a non-uniformly sampled-data system. Although the constraint that results in a non-uniform sampling arises from the computation resource shortage, but the phenomenon is similar to the NCS problem. In other words, although the delay is due to the computation resource limits, it is comparable with the communication delay. This random delay results in a non-uniform sampling. To compute the intersamples, they assumed an observer to estimate the states while the observer parameters are modified adaptively using a least-square algorithm, based on the latest received input/output pair. The input is interpolated, for example using a linear interpolation, and the model output is computed and is placed as the current output sample.

2.3 Network Design Methods

Existence of a wide range of the network protocols to implement the NCS motivates network design approaches. In these treatments the controller is assumed to be already designed and the designer designs the network parameters. To design a network there are several parameters to be determined, however, an ample amount of research has been conducted to design suitable network schedules. Loosely speaking, in the time-division multiple access (TDMA) networks where each user or each client is entitled with a certain period of time to access the communication media, network schedule determines who should access the network and for how long. This also guarantees that all users will access the network.

Another issue in transferring the data over the network is coding. To transmit information more efficiently information is coded. In other word, the data is coded to reduce the data redundancies. However, coding would also insert some noises into data. For example, analog-to-digital conversion is a coding scheme. Observer that as the number of the quantization bits decreases, the quantization noise increases. Therefore, information coding is a very important issue in transferring the data. This prompts another approach in the network design methodologies, i.e. to design a coding scheme to recover or improve the NCS performance or its stability to the traditional one. The remainder of this section represents some of the network design research on the NCS problem.

2.3.1 Network-Scheduling Methods

Earlier in Section 2.3, we introduced an NCS design methodology; network scheduling methods. Specifically, network schedule indicates allocated network resource with each user, here time-slot (we define later). Figure 2.6 represents a typical network schedule. In this schedule users $U_1, \ldots, U_4$ access the network respectively and transfer their information through the allocated time-slots. However, it is observed that each user has been assigned with a different period of time. This difference is mostly because of the different amount of the information that each user transmits. It is also observed that user U_1 has accessed two times within the same data-frame. This indicates the priority issue; some users may access the network more than the others. More specifically, network schedule guarantees that each user will access the network in *a priori* manner or as the system is running. A time-slot is the period of time that a user is assigned to transmit its information over the network. For example, interval T_2 shows the time-slot that user U_2 transmits its information.

There are two major methods to assign the time-slots to the users: Static Scheduling and Dynamic Scheduling.

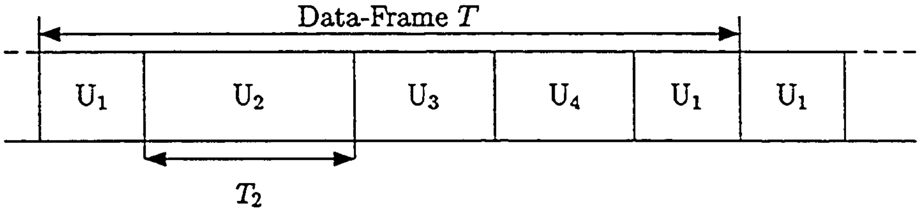

Figure 2.6: Typical TDMA-based network schedule.

Definition 2.3.1 *Static Schedule: Each client is assigned with a priori designed* *numbers of the accesses and the access time-slots, i.e. the time-slots in which the users access the network. This schedule will not change while the system is running.*

Definition 2.3.2 *Dynamic Schedule: The network is scheduled while it is running, i.e. the clients are assigned with the communication resources based on the present and the last network information; this information is usually distributed.*

In the remainder of this part we review some of the proposed dynamic and static scheduling solutions for the NCS problem.

Shin in [29] proposed a dynamic scheduling method which guarantees to minimize the probability of the messages missing their deadlines. This method was based on the Token Ring protocol, IEEE 802.5. In this protocol, the node that currently posses the token is able to transmit its data, and the token is released upon completion of the data transfer. Figure 2.7 shows a generic model for the Token Ring protocol. In this setup users capture and release the token consecutively to transfer their information. However, there exist different mechanisms to determine which node or user would receive the token.

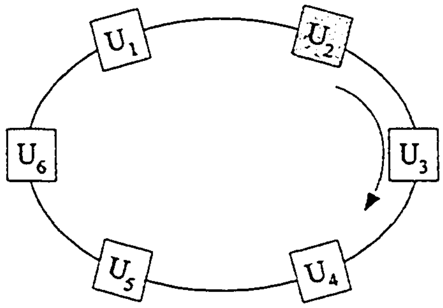

Figure 2.7: A Token-Ring Network.

Shin in [29] suggests a bus access mechanism using a *poll number*, which is uniquely computed based on the deadlines of the corresponding messages and the tasks. In this method, each node computes a poll number based on the deadlines and the user-defined priorities independently; the priorities do not change. The poll number has been designed such that the task with the earliest message deadline will have the largest poll number. After finishing the current service, the node with the largest poll number will access the bus. The poll number method achieves a better performance, in term of meeting the deadlines, compared to that of the token ring. However, as the number of the messages increases, the number of the messages missing their deadlines also increases. Additionally, it is not flexible in changing the message priorities. Moreover, the amount of the data (polling number) that was added to the data packet (packet overhead) does not allow to implement this method on commercial and comparably less expensive protocols such as Controller Area Network, CAN. Preceding issues motivated further research in this area.

Zuberi and Shin in [41] further developed the previous method to become more flexible. The messages were classified into three categories:

- *High-speed Messages*: Hard-deadline periodic messages.
- *Low-speed Messages*: Hard-deadline sporadic messages.
- *Non-priority Messages*: Nonreal-time (best-effort) aperiodic messages.

First category includes messages that their deadlines are not tolerable, i.e. they must be delivered within a certain period of time. These messages come out periodically. The next category is composed of those messages that although they are not periodic, they still need to meet their deadlines, such as the event-driven sensors. The last group contains those messages that do not have a specific deadline, such as the inventory and the diagnosis data.

For each category a specific unique ID, message header, is produced. Figure 2.8 shows a typical data packet which is transmitted over the network. In other words,

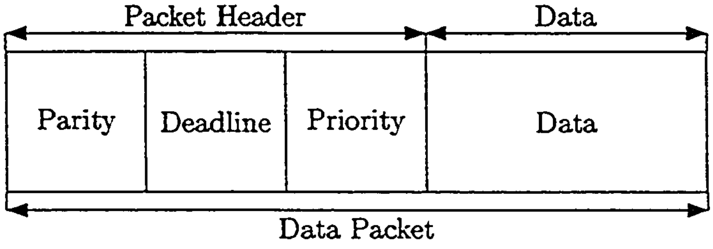

Figure 2.8: A Typical Data Packet.

high-speed messages have the highest priority, non-priority messages have the lowest priority and low-speed messages priority is placed in between. They give schedulability condition based on the category and the deadline of the message (if applicable), i.e. it determines if the network is able to transfer the data prior to the deadlines. The presented simulations in [41] shows there is a significant improvement in the network utilization. Number of the messages missing their deadlines decreases, compared to the method in [29]. However, because this method is using high performance features of CAN, short worst-case bus access delay and the bus acquisition scheme based on the priority messages, it is not possible to use it in the other network protocols.

An analytic analysis and design approach was suggested in [33] by Walsh *et al.* They modelled the network effect as a perturbation on the original system.

Figure 2.9 shows the considered setup, where $\hat{y}$ indicates the input to the controller, y denotes the plant output, x_c represents the controller states and x_p shows the plant states. The controller is described by

$$\begin{cases} \dot{x}_c = & f_c(t, x_c, \hat{y}), \\ u_p = & g_c(t, x_c). \end{cases} \tag{2.22}$$

With $x = [x_c^T, x_p^T]^T$ we arrive at the closed-loop system dynamics, below:

$$\begin{cases} \dot{x} = & f(t, x, e), \\ \dot{e} = & g(t, x, e). \end{cases} \tag{2.23}$$

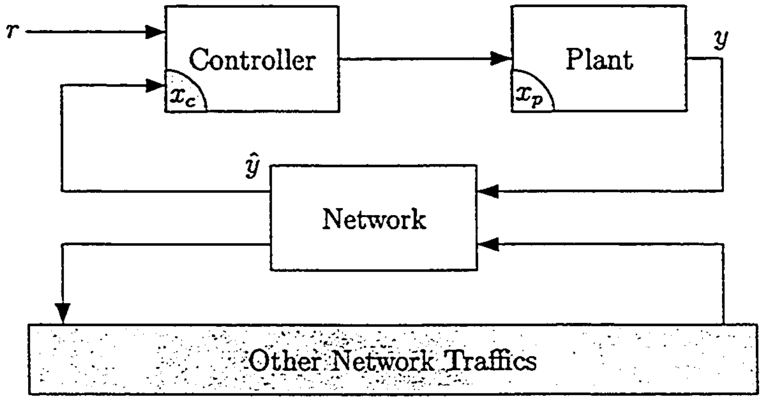

Figure 2.9: Networked Control System Configuration.

As it was mentioned before, the network effect, e, has been modelled as a perturbation on the system. It has been also assumed that the network delay is upper-bounded. They used dynamic scheduling method, Try-Once-Discard (TOD), i.e. the node with the largest difference between its current value and its last transmitted value accesses the network. After one node accesses the network bus, other nodes discard their current values and replace new data. They show that using aforementioned scheduling method or a static scheduling method that guarantees each node will access the network at least once p cycles, with a maximum allowable transfer interval τ seconds between two consecutively successful transmissions and with the maximum growth in $e(t)$ in τ seconds strictly bounded by $\beta \in (0, \infty)$ then for any time $t \geq t_0 + p\tau$, the error is bounded by

$$\|e(t)\| < \frac{\beta p(p-1)}{2}. \tag{2.24}$$

Although the upper bound is calculated analytically, but the resulting answer is barely practical.

2.3.2 Coding Methods

To transfer data in a more efficient and safer fashion, information is coded. In other words, coding removes the information redundancies to transfer the messages more efficiently; it also enables the sender and the recipient of the message to ensure the message has been transferred and received correctly. Therefore, an efficient coding scheme improves the communication quality. As we earlier mentioned, another approach to the NCS problem is to design a coding scheme to solve the problem.

Wong and Brockett in [36] and [37] investigated NCS problem in a stochastic framework. Assuming the channel specification to be known, they suggest an optimal coding state estimator by designing an optimal coding-estimator pair.

2.4 Summary

In this chapter we reviewed some research in the NCS area. Based on the design parameters, we categorized them into two groups, controller-based design and network-based design methods. However, this review raises a question: can we design both the network parameters and the controller parameters together. In this way the designer, intuitively, would have more freedom. Although we meet new challenges, this question motivates us for the next chapter, in which we will review a joint-design technique in detail.

Chapter 3

Joint Control and Network Design

Ample research has been conducted on NCS problems to design either the controller or the network; we reviewed some briefly in the previous chapter. However, simultaneous design of the controller and the network appears to be more challenging; it gives more design freedom and in turn, there exist more parameters to be determined, i.e. both the network parameters and the controller parameters should be identified.

In the next section we motivate the research on simultaneous design, where both the controller and the network parameters are determined. Section 2 covers some required preliminaries, including quantization noise, network modelling and closed loop system performance. In section 3 the problem is posed as an optimization problem and the proposed heuristic is presented. Simulation results validate our discussions. Section 4 presents a summary of this chapter.

3.1 Motivation

Recalling traditional digital control design and analysis practice, to resemble performance of the digital control system to that of the analog implementation, one should increase the sampling frequency. However, increasing the sampling frequency can lead to increase of the quantization noise, for further detail reader should consult [25]. Therefore, there will be an upper bound for increasing the sampling frequency.

In network control systems, increasing the sampling frequency, however, has different consequence.

Lian *et al.* [18] investigated performance of NCS, digital control systems and analog control systems versus the sampling time. They established a test bed to monitor the control system performance in aforementioned modes. The controller has been designed without considering effect of the network. A discretized version of the analog controller has been used in the digital and the network-based control schemes. Figure 3.1 shows the performance for the fixed plant-controller system. In the analog and the digital implementation, the signals from the sensors to the controllers and from the controllers to the actuators are transmitted via the hard-wired connections, i.e. there is no network connection. In the network-based implementation, though, the sensor information and the control sequences are transmitted over the network. The media access is controlled by a congestion-control-based protocol. Apparently, in

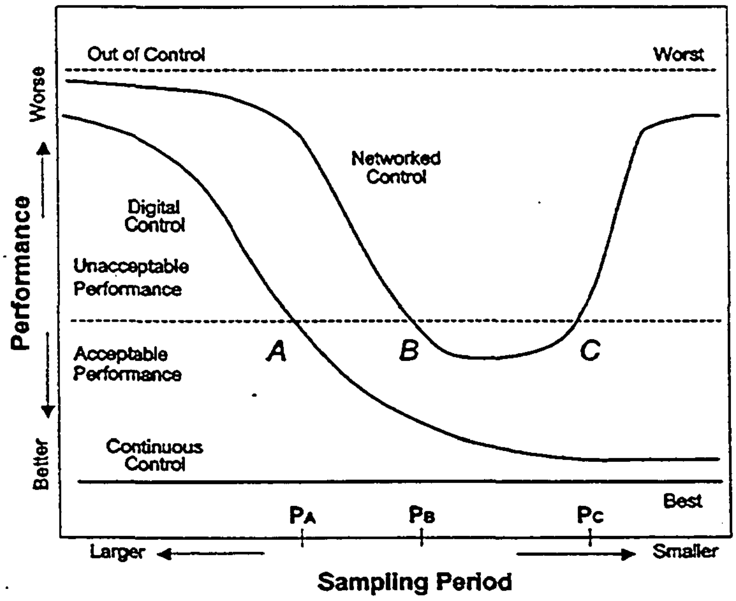

Figure 3.1: Control System Performance vs. Sampling Time

the analog implementation of the control system, the system performance is constant with respect to the sampling period. For the digital implementation, however, we see that the performance varies while the sampling time changes. As sampling period decreases the system performance is recovered to that of the analog scheme, which is desired. But in the network implementation, we observe a major difference. In Figure 3.1 from the very left hand-side while the sampling period is decreasing the performance is improving, passing point B, and it is acceptable. However, after passing a mid-point between B and C, unlike the digital scheme, the performance worsens and after passing point C it is not even acceptable anymore.

To describe this phenomenon we mention the limited communication resources. With increasing the sampling time, the network users, the sensors and the controllers, send more data packets in one unit time. In other words, probability of sending information and equally making the bus busy increases.

Definition 3.1.1 *Network Throughput: The ratio of the offered traffic to the transmitted traffic is the the network throughput.*

For a CSMA/CD network the throughput is defined by [16]

$$S = \frac{Ge^{-aG}}{G(1+2a)+e^{-aG}}, \tag{3.1}$$

where a is the ratio of the propagation delay to the packet transmission time and G is the offered traffic rate. One observes in Figure 3.2 while the packet arrival traffic rate, G, increases as a result of increasing the sampling frequency, network throughput increases until a maximum value G_{max}. With increasing the offered traffic more than G_{max} the network throughput decreases. This decrement in the network throughput degrades the NCS performance which verifies the empirical result in [18].

Loosely speaking, to increase the network throughput one must increase the network capacity. In [17], an NCS has been implemented over a switched Ethernet net-

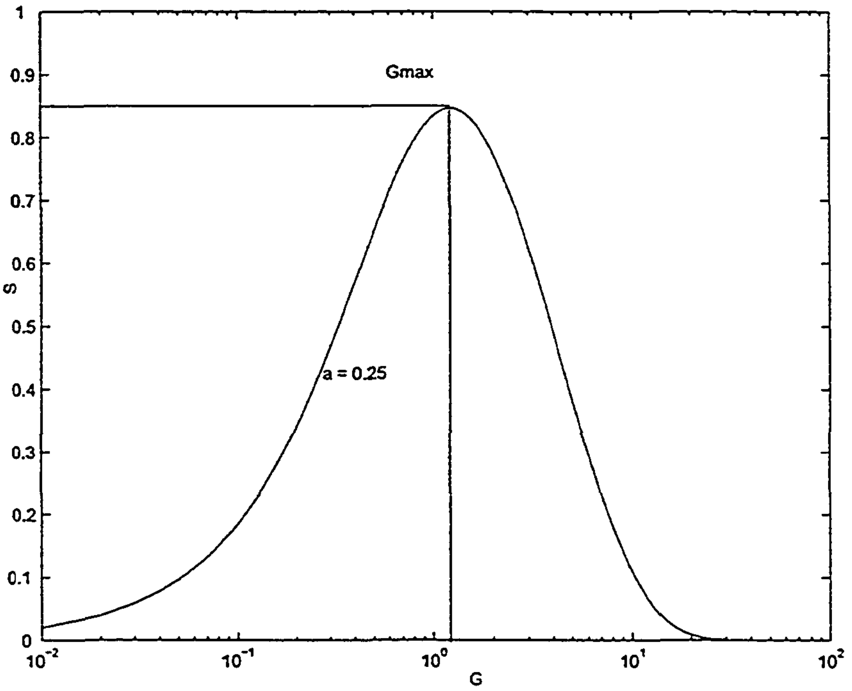

Figure 3.2: Throughput in the CDMA/CD Network.

work, which operates at 100 Mbps; this network type has been used to increase the network capacity. The experimental result in this work reveals that performance of the NCS is recovered to that of the digital system provided the network transmission capacity is large enough. However, switched-Ethernet technology is still developing and yet expensive to use.

Because the network resources are scarce, it is very important to improve the efficiency of the system, i.e. increasing the network utilization. There exist ways to further improve the efficiency of network based control systems:

- Optimizing the *Offered Traffic.*

- Optimizing the *Offered Capacity.*

- Combined Optimization.

The offered traffic indicates how the device-related information affects the system performance. The offered capacity shows how the network resource should be allocated to each device to transfer an offered volume of the traffic. Therefore, simultaneous optimization of the network offered traffic and the network resource allocation would improve the overall system performance. This motivates further research on the simultaneous optimization of the network and the controller parameters. In the next section, we give some preliminaries to model the NCS.

3.2 Preliminaries

Xiao *et al.* in [39] considered to simultaneously design the optimal controller and achieve an optimal resource allocation. They considered to control the plant over the wireless communication network. Figure 3.3 shows the assumed control system. The sensor information, y, and control sequences, u, are transmitted over two wireless links, which have limited bit-rates. They studied problem of the optimal resource allocation of the network resources combined with the optimal estimated state-feedback

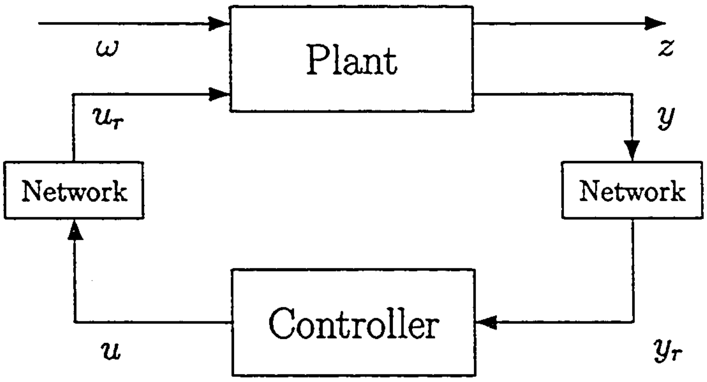

Figure 3.3: NCS in Standard Setup.

control. In this study network effect has been translated to the quantization white noise. To see this we notice that the communication channel bit-rate can be translated to the number of bits by which a signal sample is quantized with

$$b = \alpha r, \ \alpha = c_s/f_s. \tag{3.2}$$

In equation (3.2), b is the number of quantization bits, r is the channel bit-rate in bps, c_s is the coding efficiency (source bits per transmitted bits) and f_s is the sampling frequency in Hz. Therefore we can explain the transmission rate by the number of bits the signal samples are quantized with. In the next part we review basic concepts of the quantization and the required assumptions.

3.2.1 Quantization

In this part we review the fundamental concepts in the quantization area and the necessary conditions. To transfer or process an analog signal we need to first sample the signal at a proper rate and then quantize the samples, i.e. we represent the samples by finite-length digital words.

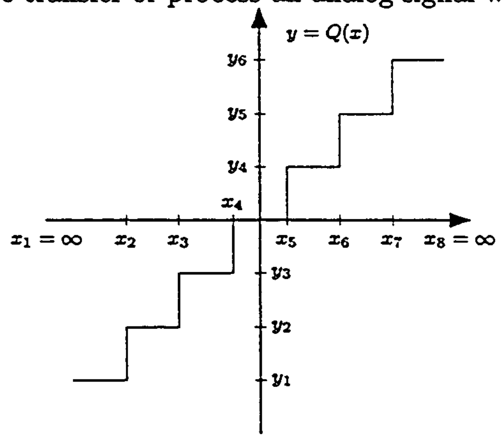

Figure 3.4: Uniform Quantization Scheme.

There exist two major quantization schemes, *uniform* quantization and *nonuniform* quantization. Figure 3.4 shows a uniform quantizer with intervals

$$[x_1, x_2), \dots, [x_{k-1}, x_k),$$

where $x_1, x_2, \dots, x_k$ are quantization intervals and $y_1, y_2, \dots, y_k$ are quantization levels.

Definition 3.2.1 *Uniform quantization: If the lengths of the quantization intervals are all equal except for the outer most positive or most negative levels and the quantization levels are separated by the same length, then the quantization is uniform [11].*

Definition 3.2.2 *Nonuniform quantization: If the quantization intervals or the quantization levels are not equal, then it is called nonuniform quantization.*

Although uniform quantization is not optimal but it is easily implemented and it gives a simple yet fairly accurate model for the network. It is assumed that the signals' amplitudes lie in $[-1, 1]$. Therefore with b quantization intervals we have

$$y = Q_b(x) = \frac{\mathbf{round}(2^{b-1}x)}{2^{b-1}}, \tag{3.3}$$

,where y is the quantized signal, x is the original signal sample, b is the number of quantization bits, and $\mathbf{round}(z)$ is the integer nearest to z. From (3.3) we can compute the quantization error:

$$E_b(x) = y - x = \frac{\mathbf{round}(2^{b-1}x) - 2^{b-1}x}{2^{b-1}}. \tag{3.4}$$

The numerator in the right-hand side of the expression (3.4) lies always between ±1/2. It follows that

$$|E_b(u)| \leq 2^{-b}. \tag{3.5}$$

However, it is not always the case that the signal is properly ranged. Therefore we need to scale the signal samples prior to the quantization. To handle this problem we use a pre-scaling factor s, where s is chosen appropriately:

$$y = sQ_b(\frac{x}{s}). \tag{3.6}$$

For stochastic signals, One way to ensure that overflow would rarely happen is to use *3σ* rule, i.e. we choose $s = 3\mathbf{rms}(x)$, where

$$\mathbf{rms}(x) = \lim_{t\to\infty}(\mathrm{E}x(t)^2)^{1/2}. \tag{3.7}$$

Then we can calculate quantization error by the following

$$E_b(x) = y - x = sE_b(\frac{x}{s}). \tag{3.8}$$

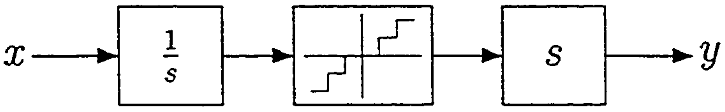

Figure 3.5: Scaled Quantization.

Figure 3.5 shows quantization with pre-scaling, where the scaling factor is s. After the quantization the value is multiplied by the scaling factor. Assuming that the signal sample x varies several levels in each sample, the quantization error is modelled as an independent random variable, uniformly distributed on the interval:

$$s[-2^{-b}, 2^{-b}]. \tag{3.9}$$

Figure 3.6 shows the quantization effect on the samples, which has been modelled as an additive white noise source. One observes that the variance of this noise is a function of the number of quantization bits.

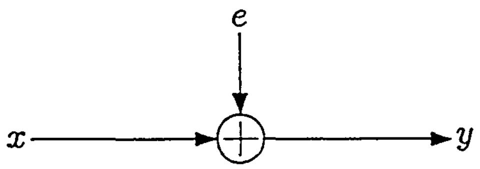

Figure 3.6: Uniform Quantization Scheme.

From (3.9), we observe that increasing the number of quantization bits decreases the quantization noise variance, and decreasing the number of quantization bits will further increase the quantization noise variance. In the next section we review the network model.

3.2.2 Network Model

In this part we exploit the TDMA network model. We implement the control system over two TDMA links. In the TDMA networks, each user accesses the network for a specific time-length, namely time-slot, though these time-slots may not have equal lengths.

Xiao *et al.* [39] proposed a convex model for the communication channel. We adopt our network to this model. Shannon [8] showed that capacity of a Gaussian channel has the following upper bound:

$$\gamma = W \log_2(1 + \frac{P}{NW}), \tag{3.10}$$

$$r \leq \gamma. \tag{3.11}$$

Equation (3.10) shows the upper bound of the Gaussian channel capacity with the bandwidth W, the transmission power P and the noise power spectral density N at the receiver side. We rewrite inequality (3.11), considering $b = \alpha r$:

$$b - \alpha T \gamma \leq 0. \tag{3.12}$$

Inequality (3.12) shows the connection between the specified bits and time-slot length T. For different users, we can write down:

$$b_i - \alpha T_i \gamma \leq 0, \qquad i = 1, ..., M, \tag{3.13}$$

$$b_{min} \leq b_i \leq b_{max}, \qquad i = 1, ..., M, \tag{3.14}$$

$$T_i \geq 0, \qquad i = 1, ..., M, \tag{3.15}$$

$$\sum_{i=1}^{M} T_i \leq T. \tag{3.16}$$

Inequalities (3.13),(3.14),(3.15), and (3.16) show the communication constraints, where b_i is the specified bits, T_i is the time-slot to the i^{th} user; observer that b_i is integer. Inequality (3.13) shows that the number of the quantization bits, b_i, is upper bounded by the specified time-slot to each user. Inequality (3.14) presents higher and lower bound, b_{min} and b_{max}, for the quantization bits, where the higher bound is imposed by the hardware constraints and the lower bound is to validate the assumption that the quantization error noise is an additive white noise.

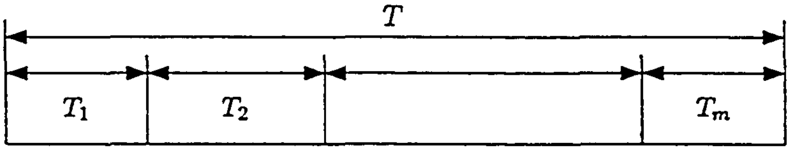

Figure 3.7: General TDMA Time-Slot.

Equality (3.16) represents the time-slots constraints, by which the summation of all time slots must be less than or equal to the main time-slot, T. Figure 3.7 shows constraint on the summation of the time-slots, where T_i is the i^{th} time-slot length and T is the maximum access time for all of the users. Inequality (3.15) represents the non-negativity constraints.

3.2.3 Network Effect

In this section we propose a model for the NCS to capture the network effect on the system. We translated the constraint due to bit-rate limitation of the communication link to the number of bits by which the signal samples are quantized, see equation (3.2). We also observed that quantization-induced error can be considered as an additive white noise source [11], $e(t)$, with zero mean and variance $\mathrm{E}e(t)^2 = 1/3s^2 2^{-2b}$, where s and b are the scaling factor and the number of quantization bits, respectively. However as it was earlier mentioned we take $s = 3\mathbf{rms}(y)$.

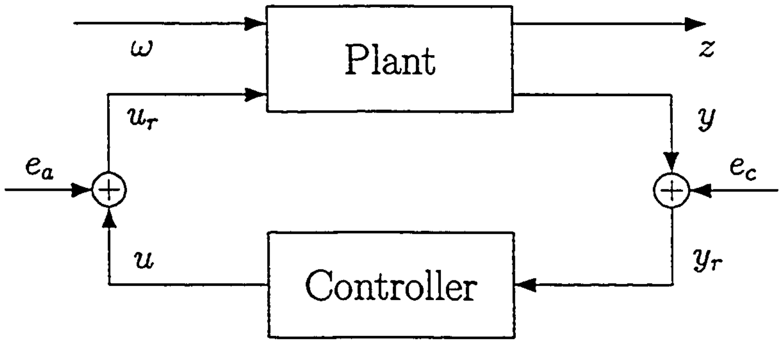

Figure 3.8: Network Effect, Modelled as White-Noise Sources.

Therefore we consider the network effect as an exogenous input. Figure 3.8 depicts this case, where ω is the exogenous input, z is the output, e_a and e_c are the effects of the networks in the controller-actuator side and the sensor-controller side, respectively. As a result we can analyze the effect of the network on the closed-loop performance more easily. In the next part we show how the network affects the closed-loop performance of the system.

3.2.4 Closed-loop Performance

In this part we analyze the effect of the network-induced error on the closed-loop system performance. For simplicity of presentation we assume the network existence in either actuator link or sensor link. However, in the case of presence of networks at both sides, the same approach is applied. As it was earlier shown, effect of bit-rate limitation due to the network presence can be modelled as an exogenous input.

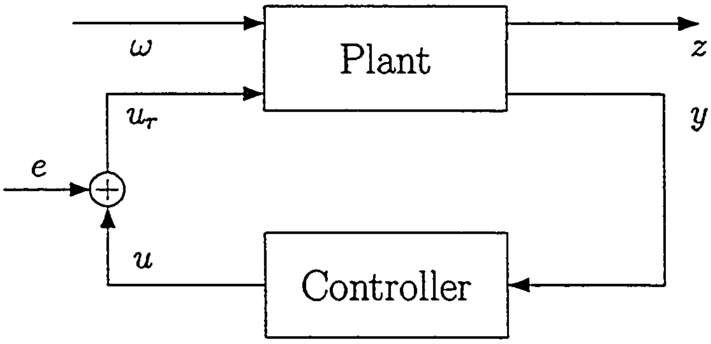

Figure 3.9: Resource Allocation.

$$z = G_{zw}w + G_{ze}e, \tag{3.17}$$

$$y = G_{yw}w + G_{ye}e. \tag{3.18}$$

Equations (3.17) and (3.18) show the closed-loop system dynamics. G_{zw} is expressing the effect of ω on the output in the absence of the network and G_{ze} represents the network effect on the output. G_{yw} and G_{ye} express effects of w and e on the output in the absence of the other one, respectively.

Remember that we aim to reduce the effect of the network on the output, z. A proper criterion to measure this effect is to measure the output variance, $\mathbf{rms}(z)$, see

[3]. We calculate the variance of z due to e, namely the network error, as follows:

$$V_e = \mathrm{E}\|G_{ze}e\|^2, \tag{3.19}$$

$$= \sum_{i=1}^{M} \underbrace{\|G_{ze_i}\|^2(1/3)s_i^2}_{a_i} 2^{-2b_i}, \tag{3.20}$$

$$= \sum_{i=1}^{M} a_i 2^{-2b_i}. \tag{3.21}$$

Equation (3.20) shows the contribution of the network error to the output variance, where $\|.\|$ is the standard 2-norm, s_i is the scaling factor ($3\mathrm{rms}(u_i)$), u_i is the i^{th} controller output, b_i is the number of the quantization bits and M is the number of actuators. One can observe that output variance depends on the scaling factor, s_i, which is determined by the controller parameters. We see that output variance also depends on the number of the quantization bits, b_i, i.e. the output variance relies on the specific bit-rate allocation to each individual user.

3.3 Problem Statement

We addressed in NCS design that we aim to minimize the output variance due to e_a and w:

minimize rms(z)

subject to:

a. Communication constraints,

b. Control constraints.

We break the above problem into two parts:

1. Controller Design.

2. Communication Network Design.

In the controller design step we design an observer-based, estimated state-feedback, controller. We require to solve an inverse LQG problem, i.e. given **rms** values of the error(s) and the actuator effort(s) find the proper controller parameters, which achieve the given values:

$$\begin{aligned} &\text{minimize } \mathbf{rms}(z) \\ &\text{subject to:} \\ &\qquad \mathbf{rms}(u_i) \leq \beta_i, \quad i = 1, ..., M. \end{aligned} \tag{3.22}$$

Optimization problem (3.22) has been addressed as a multi-criterion LQG design in [32], where β_i is the upper limit of the actuator effort. This problem can be solved using variety of convex optimization algorithms, where we used Analytical Center Cutting Plane Method (ACCPM), see [26, 27].

The second problem to solve is the optimal allocation of the communication resources:

$$\text{minimize } \sum_{i=1}^{M} a_i 2^{-2b_i}$$

Subject to:

$$b_i - \alpha T_i \gamma \leq 0, \qquad i = 1, ..., M, \tag{3.23}$$

$$b_{min} \leq b_i \leq b_{max}, \qquad i = 1, ..., M, \tag{3.24}$$

$$T_i \geq 0, \qquad i = 1, ..., M, \tag{3.25}$$

$$\sum_{i=1}^{M} T_i \leq T. \tag{3.26}$$

Although the controller design problem and the network design problem are both convex, but the joint problem is not convex, see [39], and it is mainly related to r. In other words, the lower range of the transmission rate has a significant impact, i.e. the joint problem is only convex for large transmission rates. However, a heuristic method has been proposed in [39, 3] to solve the problem, though it does not necessarily converge. It is suggested to solve two mentioned problems sequentially:

1. Allocate all resources equally and design the controller.
2. Calculate variances, $\mathbf{rms}(y_i)$ and $\mathbf{rms}(u_i)$, to determine the scaling factor.
3. Solve the network resource allocation problem.
4. Redesign the controller.
5. Repeat step 2 until convergence.

3.3.1 Simulation Results

We consider a system of five mass-spring in series [39], where sensors on the masses transfer the mass locations to the controller, an LQG-optimal controller, and the controller transfers the control sequences to the actuators on the masses, both using TDMA-baesd networks. The mechanical system parameters are shown in the table below:

Mass-Damper No.	1	2	3	4	5
Mass, M (kg)	10	5	20	2	15
Spring, K (N/m)	1	1	1	1	1

Table 3.1: System Parameters: Masses and Spring Constants.

We selected sampling time, $T_s = 1$ *sec.* and time slot is equal to 0.5 *sec.* Other network parameters are $W = 12$ bps (bandwidth),$P = 7$ Watts (transmitter power)

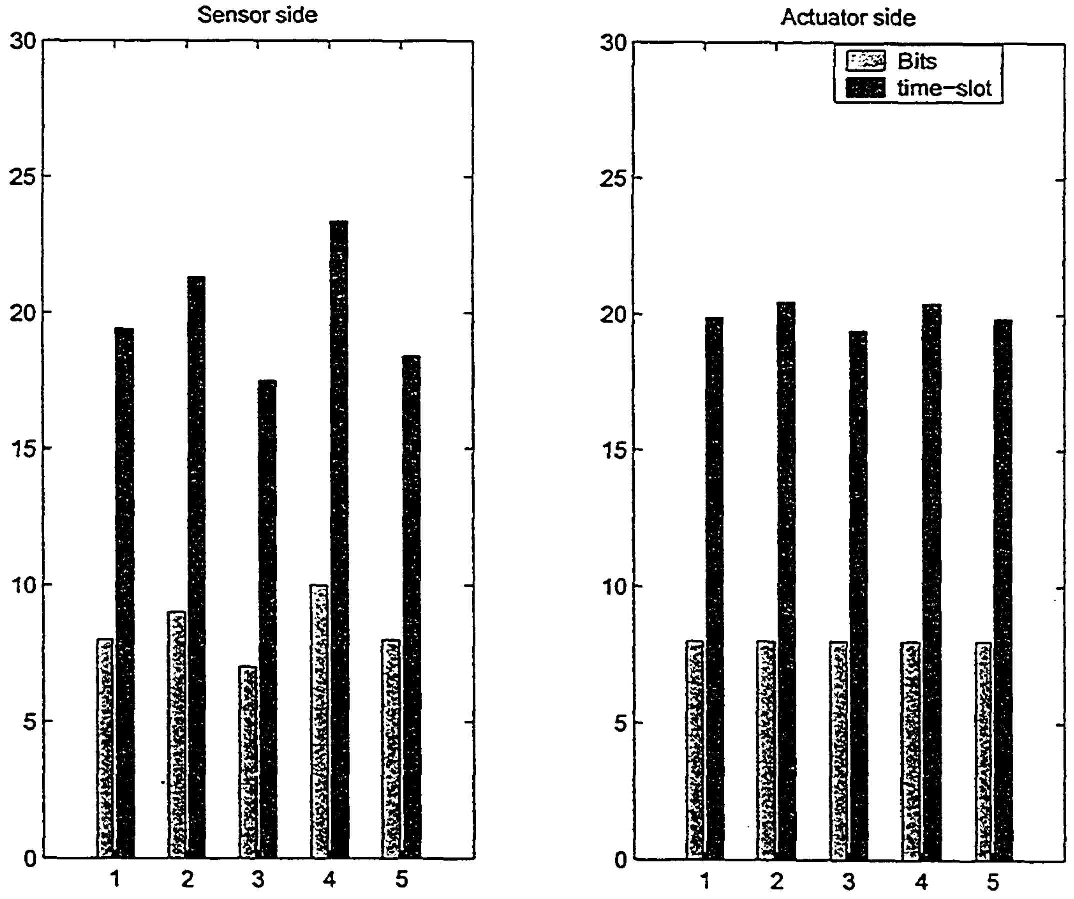

Figure 3.10: Resource Allocation.

Closed-Loop Performance	Uniform Allocation	Joint-Design Allocation
RMS(z)	0.36095	0.2219

Table 3.2: System Performance, Uniform Allocation and Joint Design.

and $N = 0.1$ (noise) at the receiver side. Figure 3.10 shows the network resource allocation for the above example. However, because the allocated bits are integer, the achieved allocation is sub-optimal. Table 3.2 shows the closed-loop performance for the uniform resource allocation and the joint design. As we were expecting, the performance has improved significantly. To explain this phenomenon we remind that each input has different impact on the system performance, therefore a proper resource allocation can improve the performance significantly. One can also observe that there is more variation on the sensor side. This is because the performance is more sensitive to the sensor noise than the actuator noise.

3.4 Summary

In this chapter we reviewed the proposed simultaneous design approach and reproduced the results in [39]. Bit-rate limitation of the network was captured as the quantization noise. Also, a convex model was extracted for the network. We observed that the simultaneous design of the network and the controller increased the system performance. Following the discussion in section 3.1, to increase the system performance we need to reduce the offered traffic and increase the network utilization. This result raises a question: Is it possible to further improve the system performance? In other words, how we can decrease the network traffic while increasing the network utilization? This will be addressed in the next chapter.

Chapter 4

Multirate Design

Although optimal allocation of network resources is important to improve the system performance, experimental results show that the most effective strategy to achieve this goal is to decrease the network traffic [24, 19]. Observe that the aim is to convert the network to a less active element in determining the system performance, i.e. to reduce its effect on the performance. Several methods have been suggested to accomplish this idea. In [35], the authors suggest to transfer only the variable which has the maximum difference with respect to its latest successfully transmitted value. An alternative is to only transmit those variables that their difference with their latest successfully transmitted samples exceed a certain threshold [34]. Another way to decrease the traffic is to use a system model to estimate the data and sending the information less frequently [2].

Unfortunately all of the above-mentioned methods have some drawbacks. For example in the method given in [2], the estimation procedure used to produce the data increases the computation load, i.e. it needs a higher level of intelligence at the node level.

In the remainder of this chapter, in section 4.1, we design an $\mathcal{H}_2$-optimal controller combined with the network resource allocation optimization problem to decrease the output variance of the close-loop system. In section 4.2, to further ameliorate the performance, a network traffic-reduction method is introduced. This method is based on the dual-rate control. To handle the dual-rate problem, a brief review of the lifting technique and some of the associated properties are also given in this section. In section 4.3, we solve the problem in the general case, i.e. we deploy a multirate control scheme. The causality constraint is discussed and the associated design procedure is introduced. Simulation results verify effectiveness of the suggested method.

4.1 Single-Rate Design

In this section we design an $\mathcal{H}_2$-optimal controller for the NCS. We consider the network bit-rate limitation as an exogenous input or standard white-noise source. Next, we show that the network-resource allocation problem is indeed a convex problem. In the next part we review the $\mathcal{H}_2$-optimal control design procedure briefly.

4.1.1 $\mathcal{H}_2$-optimal Control

In this section we introduce the standard $\mathcal{H}_2$-optimal control problem and review the associated state-space solution. Figure 4.1 shows a control system in the standard setup where ω, z, u and y are the exogenous input, the desired output, the control sequence and the controller input, respectively. The plant model and the controller are shown by P and K, respectively.

Assume the state-space model for the system P in discrete time is

$$P = \left[\begin{array}{c|cc} A & B_1 & B_2 \\ \hline C_1 & D_{11} & D_{12} \\ C_2 & D_{21} & D_{22} \end{array}\right]. \tag{4.1}$$

In the $\mathcal{H}_2$-optimal controller design, it is aimed to design a stabilizing controller, K, to minimize $\|T_{zw}\|_2$, where T_{zw} is the closed-loop system and $\|\cdot\|_2$ represents the $\mathcal{H}_2$-norm of the system. In other words, we are interested in the solution to the following

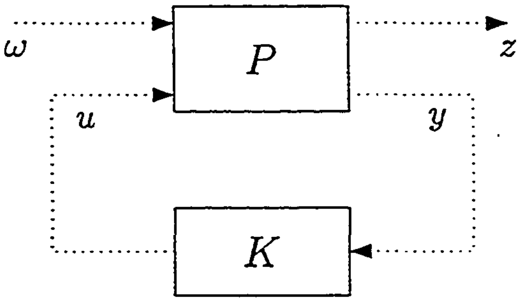

Figure 4.1: Control System in the Standard Setup.

optimization problem:

$$\min_{K \text{ stabilizing}} \|T_{zw}\|_2. \tag{4.2}$$

It is always assumed that the system is wellposed, i.e. D_{22} is equal to zero [30]. Even though this is a reasonable assumption for most of the physical systems, as we will show later, it is not applicable for general multirate systems. Other standard assumptions to this problem are:

(i) (A, B_2) is stabilizable and (C_2, A) is detectable.

(ii) $M := D_{12}'D_{12}$ and $N := D_{21}D_{21}'$ are nonsingular.

(iii) The matrices

$$\begin{bmatrix} A-\lambda & B_2 \\ C_1 & D_{12} \end{bmatrix}, \begin{bmatrix} A-\lambda & B_1 \\ C_2 & D_{21} \end{bmatrix},$$

have full column and full row ranks, respectively, $\forall \lambda \in \partial D$[1].

Under these assumptions, the $\mathcal{H}_2$-optimal control problem has a unique solution which can be obtained by solving two Riccati equations. Then the unique optimal

[1] By ∂D we denote the set of all $s \in C$ such that $|s| = 1$, where C denotes the complex numbers set.

controller, K_{opt}, is determined by

$$K_{opt} := \left[\begin{array}{c|c} A + B_2F + LC_2 - B_2L_0C_2 & L - B_2L_0 \\ \hline L_0C_2 - F & L_0 \end{array}\right], \tag{4.3}$$

where we define

$$\begin{aligned}
H &= \left(\begin{bmatrix} A - B_2M^{-1}D_{12}'C_1 & 0 \\ -C_1'(I - D_{12}M^{-1}D_{12}') & I \end{bmatrix}, \begin{bmatrix} I & B_2M^{-1}B_2' \\ 0 & (A - B_2M^{-1}D_{12}'C_1)' \end{bmatrix} \right), \\
X &= Ric\,(H), \\
R &= (M + B_2'XB_2), \qquad (4.4) \\
F &= -R^{-1}(B_2'XA + D_{12}'C_1), \\
F_0 &= -R^{-1}(B_2'XB_1 + D_{12}'D_{11}), \\
T &= \left(\begin{bmatrix} (A - B_1D_{21}'N^{-1}C_2)' & 0 \\ -B_1(I - D_{21}'N^{-1}D_{21})B_1' & I \end{bmatrix}, \begin{bmatrix} I & C_2'N^{-1}C_2 \\ 0 & A - B_1D_{21}'N^{-1}C_2 \end{bmatrix} \right), \\
Y &= Ric\,(T), \\
S &= (N + C_2YC_2'), \qquad (4.5) \\
L &= -(AYC_2' + B_1D_{21}')S^{-1}, \\
L_0 &= (FYC_2' + F_0D_{21}')S^{-1}.
\end{aligned}$$

The detailed derivation is available in [7]. An example illustrates the design procedure.

Example 4.1.1

Consider a system of two pairs of mass-spring's, which are connected in series, Figure 4.2. The mechanical parameters of the system are shown in Table 4.1.1

Mass-Spring No.	1	2
Mass, M (kg)	10	15
Spring, K (N/m)	1	1

Table 4.1: Mechanical Parameters of the System.

Starting with the continuous-time system model:

$$P_c(s) = \left[\begin{array}{c|c} A_c & B_c \\ \hline C_c & D_c \end{array}\right], \tag{4.6}$$

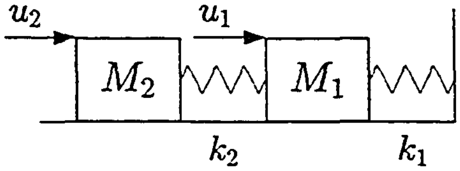

Figure 4.2: Two Mass-Spring Pairs in Series.

we compute the discretized model:

$$P(\lambda) = \left[\begin{array}{c|c} A & B \\ \hline C & D \end{array}\right], \tag{4.7}$$

where $(A, B)^2$ are discretization of the continuous-time system in (4.6) with the sampling time h^3. We aim to regulate the mass locations, y, in the existence of the exogenous noise, ω, using the actuator efforts, u. The desired output signal, z, is:

$$z = \begin{bmatrix} u \\ y \end{bmatrix}.$$

With partitioning the input and output of G:

$$\begin{bmatrix} \omega \\ u \end{bmatrix}, \begin{bmatrix} z \\ y \end{bmatrix}, \tag{4.8}$$

we arrive at realization of P in the standard setup:

$$P = \left[\begin{array}{c|cc} A & [0 \;\; B] & B \\ \hline C & [0 \;\; D] & D \\ C & [I \;\; D] & D \end{array}\right]. \tag{4.9}$$

The $\mathcal{H}_2$-optimal controller of the system in (4.9) is

$$K_{opt} := \left[\begin{array}{c|c} A_k & B_k \\ \hline C_k & D_k \end{array}\right], \tag{4.10}$$

where A_k, B_k, C_k and D_k are

[2] We may use Matlab-like expression for simplicity,$(A, B) = c2d(A_c, B_c, h)$, (A_c, B_c) indicates the continuos-time system model.

[3] Sampling time, h, is $0.1 \times \frac{2\pi}{\lambda}$, where λ is the largest eigen-value of A_c.

$$K_{opt} = \left[\begin{array}{cccc|cc} -4.0877 & -0.0016 & -0.7834 & -0.0001 & -0.0409 & -0.0078 \\ -2.7767 & -1.0014 & -0.7888 & -0.0001 & -0.0192 & -0.0075 \\ -1.5229 & -0.0001 & -4.2174 & -0.0008 & -0.0152 & -0.0422 \\ -1.5363 & -0.0002 & -2.8532 & -1.0007 & -0.0146 & -0.0200 \\ \hline 12.1336 & 8.5761 & 2.2662 & 0.3700 & 0.0944 & 0.0145 \\ 2.2259 & 0.3697 & 6.2132 & 4.2861 & 0.0141 & 0.0487 \end{array}\right].$$

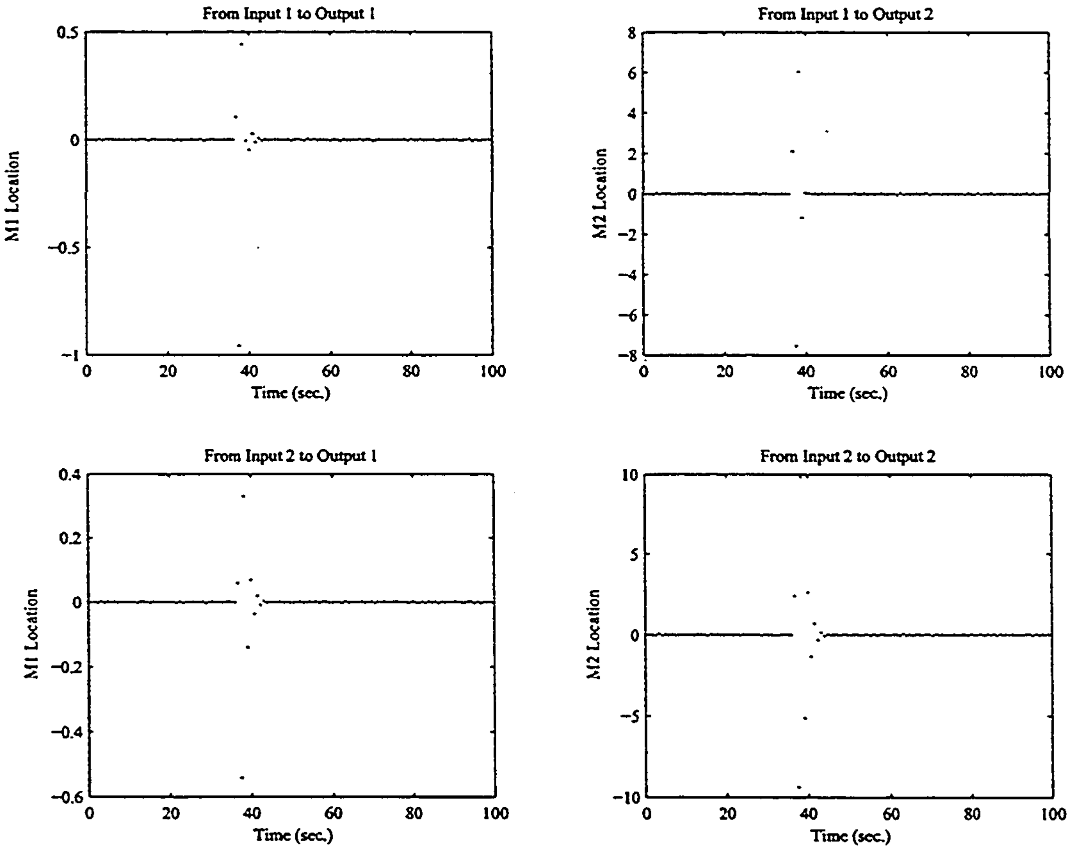

Figure 4.3: Impulse-response of the Mass-spring system.

Figure 4.3 shows the impulse response of the closed system in In the next section, we examine convexity of the network resource allocation problem.

4.1.2 Allocation Problem

Recall from the previous chapter, we aimed at allocating the time-slots optimally. For this purpose, we solve the following optimization problem

$$\min_{b_i,T_i} \sum_{i=1}^{M} a_i 2^{-2b_i} \quad i = 1, ..., M, \tag{4.11}$$

Subject to:

$$b_i - \alpha T_i \gamma \leq 0, \qquad i = 1, ..., M, \tag{4.12}$$

$$b_{min} \leq b_i \leq b_{max}, \qquad i = 1, ..., M, \tag{4.13}$$

$$T_i \geq 0, \qquad i = 1, ..., M, \tag{4.14}$$

$$\sum_{i=1}^{M} T_i \leq T, \tag{4.15}$$

where a_i, b_i and T_i are as introduced before. The allocation problem is indeed a convex optimization problem; this makes the problem computationally tractable when the problem size increases. To verify this, we show that the objective function along with the associated constraints are convex.

To show the objective function convexity, we need only to show that $a_i 2^{-2b_i}$ is a convex function, because the summation of $a_i 2^{-2b_i}$ terms is also a convex function [4]. The function, $a_i 2^{-2b_i}$, is a single variable function b_i, therefore to show the convexity we only need to show

$$\frac{d^2(a_i 2^{-2b_i})}{d{b_i}^2} \geq 0, \tag{4.16}$$

where this is simply verified; a_i and b_i are positive. The convexity condition for the constraints is similarly verified.

To verify the effect of the communication channel capacity on the closed-loop system performance, we setup a benchmark example. We return to this problem frequently.

Example 4.1.2

Consider the system in example 4.1.1, which the sensor information and control sequences are transmitted over a wireless channel. Communication channel parameters are shown in Table 4.1.2. The results have been collected for three different channel bit-rates; the rest of the parameters remain unchanged.

Channel Parameters	P(watt)	SNR(dB)
Value	15	15

Table 4.2: Communication Channel parameters.

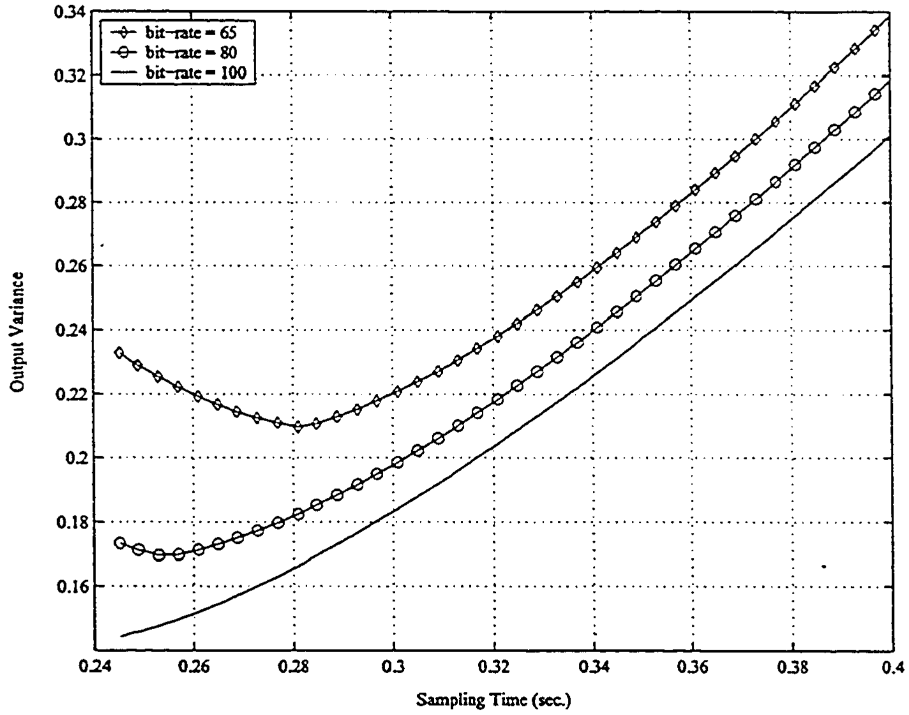

Figure 4.4: Performance Curves for the Single-rate System vs. the Sampling Time.

The system performance for three different channel bit-rates versus the sampling time are shown in Figure 4.4. For the bit-rate values 65 bps and 80 bps, from the very left-hand side, by increasing the sampling time, the output variances decrease until they achieve their minima. After passing the minima, while the sampling time increases, the variance also increases. Observe that as the channel capacity increases, the output variance decreases. However, if the communication channel capacity is large enough, e.g. the curve for bit-rate 100 bps, the output variance increases as the sampling time increases. Indeed, corresponding to the first two cases, the network-induced noise dominates the process and the sensor noises, for small enough sampling times. However, when the channel capacity is large enough, the network-induced noise is dominated by the process and the sensor noises.

The results shown in Figure 4.4 verify the effectiveness of the idea to reduce the network traffic. In the next section, we propose a strategy to reduce the traffic.

4.2 Dual-Rate Setup

Observe that not all the information needs to be either sampled or updated at the same rate, i.e. we can update some of the variables faster than the others, where this resembles a *multirate* system. In the industry, multirate systems usually arise because of the constraints in the sampling or updating the control information, e.g. an analyzer produces its output at, say, every 30 minutes, while a pressure transmitter can return its measurement at fractions of a second. However, in our case, we suggest to deliberately send information at different rates to decrease the network traffic. To verify our conjecture, first we verify this method for a *dual-rate* system. We

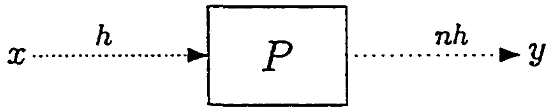

Figure 4.5: Multirate Setup.

assume that the information is sampled at two different rates; the control sequences are updated every h seconds, the fast rate, and the sensors are sampled at every nh second, the slow rate.

Figure 4.5 presents a typical single-input single-output dual-rate system, where the input, x, is updated every h seconds , shown by high-frequency dots, and the output, y, is updated every nh seconds, shown by low-frequency dots. Observe that the above system is not time-invariant, i.e. we are not able to achieve an LTI model for the system. As a result, we can not use many control schemes that mainly belong to the time-invariant framework. A standard way to handle this problem is to use the so-called *lifting* technique. In the next part, we briefly review this technique and collect some of the properties related to our work.

4.2.1 Discrete Lifting

In this section, we introduce the lifting technique in the discrete domain and collect some of the associated properties. We first explain lifting for the signals and next summarize the system lifting. Loosely speaking, the term lifting addresses unifying signal rates of the signals which are sampled at different rates by stacking up them. The lifting technique [15] was originally derived from the Kranc's idea of switch decomposition. An example clarifies this concept. Consider two signals $x(\cdot)$ and $y(\cdot)$, which are sampled every h and $2h$ seconds, respectively:

$$x(k) = \{x(0), x(h), x(2h), \ldots\}, \; y(k) = \{y(0), y(2h), y(4h), \ldots\}. \tag{4.17}$$

Using the lifting technique, we arrive at two signals, which have a unique sampling time equal to $2h$, $l.c.m\ (h, 2h)$[4]. Underlining signal represents lifting.

$$\underline{x} \;=\; \left\{ \begin{bmatrix} x(0) \\ x(h) \end{bmatrix}, \begin{bmatrix} x(2h) \\ x(3h) \end{bmatrix}, \ldots, \begin{bmatrix} x(nh) \\ x((n+1)h) \end{bmatrix}, \ldots \right\}, \tag{4.18}$$

$$y \;=\; \{ [\, y(0) \,], [\, y(h) \,],\ \ldots, [\, y(nh) \,],\ \ldots \} \tag{4.19}$$

The above rearrangement is summarized using the lifting operator, L_N, where N shows the order of the lifting; the inverse lifting is also shown by L_N^{-1}. For example, we can rewrite (4.18)

$$\underline{x} = L_2 x\ , \quad x = L_2^{-1}\underline{x}.$$

To keep the notation simple we drop the lifting order unless it is not clear. The system matrix representation of the lifting and the inverse lifting operator, for $N = 2$, are

$$L = \left[\begin{array}{c|c|c|c|c|c} I & 0 & 0 & 0 & 0 & \cdots \\ 0 & I & 0 & 0 & 0 & \cdots \\ \hline 0 & 0 & I & 0 & 0 & \cdots \\ 0 & 0 & 0 & I & 0 & \cdots \\ \hline \vdots & \vdots & \vdots & \vdots & \vdots & \end{array}\right], \quad L^{-1} = \left[\begin{array}{cc|cc|cc|c} I & 0 & 0 & 0 & 0 & 0 & \cdots \\ \hline 0 & I & 0 & 0 & 0 & 0 & \cdots \\ \hline 0 & 0 & I & 0 & 0 & 0 & \cdots \\ \hline 0 & 0 & 0 & I & 0 & 0 & \cdots \\ \hline \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \end{array}\right], \tag{4.20}$$

where I is the identity matrix of a proper size. To achieve the lifted system, we lift the input signals and the output signals appropriately, which is illustrated in the block diagram below. It is assumed that the input is updated at the fast-rate, h, and the output is sampled at the slow-rate, nh. Figure 4.6 represents the assumed block diagram to compute the lifted system, where H_f and S_f are the fast-rate hold, and the fast-rate sampler, respectively, and S represents slow-rate sampler. Then the lifted model is

$$\underline{P} = \left[\begin{array}{c|cccc} A^n & A^{n-1}B & A^{n-2}B & \cdots & B \\ \hline C & D & 0 & \cdots & 0 \end{array}\right], \tag{4.21}$$

[4] $l.c.m(a, b)$ denotes the least common multiple of the numbers a and b.

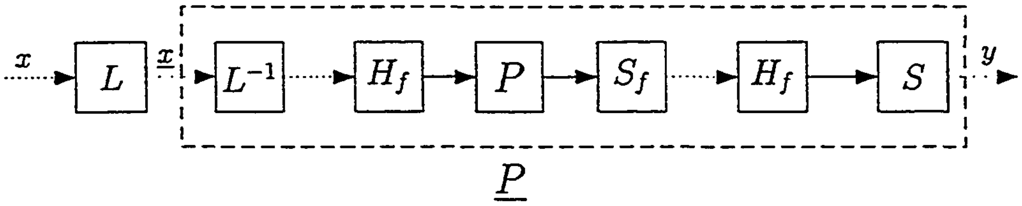

Figure 4.6: Lifted Multirate Setup.

where $(A,B)=$ $c2d(A_c,B_c,\ h)$, for detailed derivations reader may consult [7].

Remark **4.2.1** *Lifting a system does not increase the system order, the importance becomes clearer when an optimal controller is designed for the plant, which is of the system order.*

Remark **4.2.2** *Lifting a system preserves the system stability, i.e. if A is a stable matrix, so is A^n.*

Remark **4.2.3** *The lifting operator is norm-preserving. In the case of $\mathcal{H}_2$-norm, which is of interest to us, it can be shown that*

$$\|P\|_2^2 = n\|\underline{P}\|_2^2.$$

Remark **4.2.4** *Lifting is not a causal operator, observe that (4.18) is neither lower-triangular nor Toeplitz. However for the dual-rate case, it does not limit us. We will address this issue later on.*

The next example verifies effectiveness of our conjecture. We solve the benchmark problem in a dual-rate setup.

Example 4.2.1

Sampling-Scheme	Fast-rate	Dual-rate	Slow-rate
RMS(z)	0.2630	0.2219	0.36095

Table 4.3: Close-loop System Output-variance with Different Sampling Schemes.

Consider the benchmark problem, example 4.1.2. The sensor measurements are sampled at every $2h$ seconds, and the control sequences are updated at every h seconds. Communication channel specifications and the sampling time are similar to those in that example. The process and the sensors noise covariance matrices are

$$R_v = 0.6 \times 10^{-6} I_1, \quad R_\omega = 0.6 \times 10^{-6} I_2,$$

where I_1 and I_2 are identity matrices with proper dimensions. The controller and the network parameters are computed in three steps:

(I) Compute the lifted state-space model of the plant, $n = 2$.

(II) Design the dual-rate $\mathcal{H}_2$-optimal controller.

(III) Solve the resource allocation optimization problem.

To compare the closed-loop system performances in the fast-rate, slow-rate and dual-rate modes, we solved the problem in these setups, separately. Table 4.3 presents the simulation results. Observe that unlike the sampled-data control systems, which the dual-rate system performance is placed between those of the fast-rate and slow-rate schemes, here dual-rate control achieves lowest output variance, i.e. it has the highest performance among the three candidates. The achieved result agrees with our conjecture. Observe while the sampling-time is shortened, simultaneously the channel capacity decreases, which results in increasing the network-induced noise, i.e. the network noise dominates the other exogenous noises.

In the next section we investigate this idea in the general framework, i.e. we consider a multirate setup.

4.3 Multirate Design

The term multirate is applied to systems whose inputs are updated at different rates and outputs are sampled also at different rates. Figure 4.7 depicts a general multirate sampled-data system, where its i^{th} input is updated every $n_i h$ seconds and its j^{th} output is sampled every $k_j h$ seconds. The lifted system will have $\tilde{N}$ inputs and $\tilde{K}$ outputs, where $\tilde{N}$ and $\tilde{K}$ are

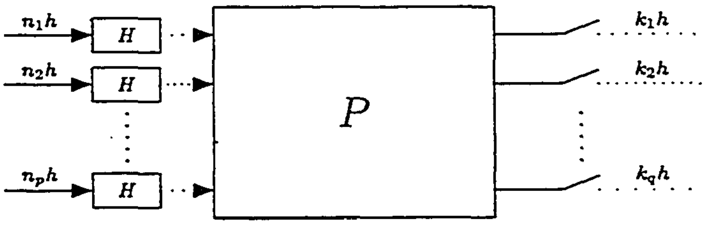

Figure 4.7: General Multirate Sampled-date System.

$$\begin{aligned}
\tilde{N} &= \sum_{i=1}^{p} \tilde{N}_i,\ \tilde{N}_i = \frac{N}{n_i} \quad ,i = 1, 2, \ldots, p, \\
\tilde{K} &= \sum_{j=1}^{q} \tilde{K}_j,\ \tilde{K}_j = \frac{N}{k_j} \quad ,j = 1, 2, \ldots, q, \\
N &= lcm(n_1, n_2, \ldots, n_p, k_1, k_2, \ldots, k_q).
\end{aligned}$$

The lifted plant in the standard form is

$$P = \left[\begin{array}{c|cc} \underline{A} & \underline{B}_1 & \underline{B}_2 \\ \hline \underline{C}_1 & \underline{D}_{11} & \underline{D}_{12} \\ \underline{C}_2 & \underline{D}_{21} & \underline{D}_{22} \end{array}\right]. \tag{4.22}$$

Unlike the dual-rate case, the associated $\mathcal{H}_2$-optimal multirate controller is not causal. To see this, we lift the controller inputs and the controller outputs as follows:

$$\underline{y}(k) = \begin{bmatrix} \underline{y}_1(k) \\ \vdots \\ \underline{y}_p(k) \end{bmatrix}_{\bar{K}\times 1}, \quad \underline{y}_j(k) = \begin{bmatrix} y_j(k_jk) \\ \vdots \\ y_j(k_jk+k_j-1) \end{bmatrix}_{\bar{K}_j\times 1}, j = 1, 2, \ldots, q, \quad (4.23)$$

$$\underline{u}(k) = \begin{bmatrix} \underline{u}_1(k) \\ \vdots \\ \underline{u}_q(k) \end{bmatrix}_{\bar{N}\times 1}, \quad \underline{u}_i(k) = \begin{bmatrix} u_i(n_ik) \\ \vdots \\ u_i(n_ik+n_i-1) \end{bmatrix}_{\bar{N}_i\times 1}, i = 1, 2, \ldots, p, \quad (4.24)$$

Observe that the elements of the two vectors, may occur at different time instants. Therefore, to preserve the controller causality, a special structure for the controller feedthrough term, $\underline{D_k}$, is required. Indeed, $\underline{D_k}$ must be a block lower-triangular matrix, after some coordinate transformations, to hold this property:

$$\underline{D_k} = \begin{bmatrix} \underline{D_{k_{11}}} & 0 & \cdots & 0 \\ \underline{D_{k_{21}}} & \underline{D_{k_{22}}} & \cdots & 0 \\ \vdots & \vdots & & \vdots \\ \underline{D_{k_{q1}}} & \underline{D_{k_{q2}}} & \cdots & \underline{D_{k_{qp}}} \end{bmatrix}_{\bar{N}\times\bar{K}}. \quad (4.25)$$

Shu and Chen in [30] have suggested a procedure to design the $\mathcal{H}_2$-optimal multirate controller, subject to the causality constraint, which we summarize it here. Assume that the lifted plant model (4.22) in the standard form, with the assumptions and definitions similar to the standard $\mathcal{H}_2$-optimal controller that were introduced in section 4.1.1. They proposed the following procedure:

(I) Decompose R (4.4) to $R_1'R_1$ and S (4.5) to S_1S_1' such that R_1 and S_1 are lower block-triangular.

(II) Decompose $R_1L_0S_1$ to $W + W_\perp$, where W and $W_\perp$ are lower and upper block-triangular matrices, respectively.

(III) Define $V = L_0 - R^{-1}W_\perp S^{-1}$.

Then the unique optimal controller $\underline{K}_{opt}$ is

$$\underline{K}_{opt} := \left[\begin{array}{c|c} A + \underline{B}_2 F + LC_2 - \underline{B}_2 V \underline{C}_2 & -L + \underline{B}_2 V \\ \hline F - V\underline{C}_2 & V \end{array}\right]. \tag{4.26}$$

The proposed procedure, however, is not applicable in general. Revisiting the standard $\mathcal{H}_2$-optimal controller design procedure, observe that unlike the single-rate case (4.1), which has $D_{22} = 0$, the lifted model (4.22) does not satisfy this. In the next part, we suggest a design procedure to handle this issue.

Alternative Design

As we showed in the previous section, although the fast-rate model has $D_{22} = 0$ but the lifted model does not hold this property. To remove this obstacle, we decompose the system into two subsystems, namely P_1 and P_2, and absorb P_2 into the controller K, as shown in Figure 4.8.

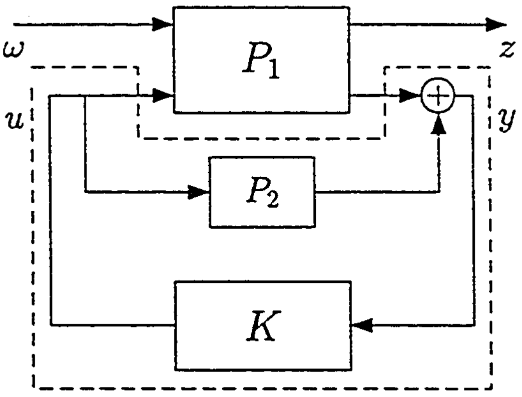

Figure 4.8: Decomposed System Block Diagram.

$$P = \left[\begin{array}{c|cc} A & B_1 & B_2 \\ \hline C_1 & D_{11} & D_{12} \\ C_2 & D_{21} & D_{22} \end{array}\right] = \underbrace{\left[\begin{array}{c|cc} A & B_1 & B_2 \\ \hline C_1 & D_{11} & D_{12} \\ C_2 & D_{21} & 0 \end{array}\right]}_{P_1} + \underbrace{\left[\begin{array}{c|cc} 0 & 0 & 0 \\ \hline 0 & 0 & 0 \\ 0 & 0 & D_{22} \end{array}\right]}_{P_2}. \tag{4.27}$$

By this new formulation, we arrive at the new system which is shown in Figure 4.9.

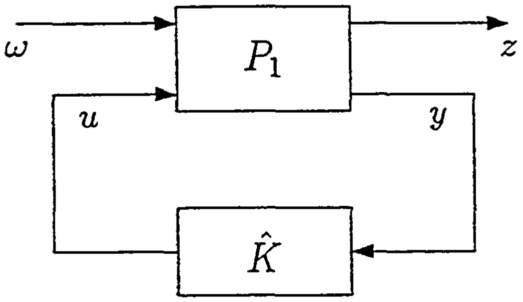

Figure 4.9: Reformulated Problem.

One can observe that the new system P_1 satisfies the condition that $D_{22} = 0$.

***Corollary* 4.3.1** *Given system P (4.27), and controller $\hat{K}$:*

$$\hat{K} = \left[\begin{array}{c|c} \hat{A}_K & \hat{B}_K \\ \hline \hat{C}_K & \hat{D}_K \end{array}\right], \tag{4.28}$$

the optimal controller for the subsystem P_1, the unique optimal controller for the system P is given by

$$\underline{K}_{opt} = \left[\begin{array}{c|c} \hat{A}_K - \hat{B}_K \underline{D}_{22}(I + \hat{D}_K \underline{D}_{22})^{-1}\hat{C}_K & \hat{B}_K - \hat{B}_K \underline{D}_{22}(I + \hat{D}_K \underline{D}_{22})^{-1}\hat{D}_K \\ \hline (I + \hat{D}_K \underline{D}_{22})^{-1}\hat{C}_K & (I + \hat{D}_K \underline{D}_{22})^{-1}\hat{D}_K \end{array}\right].$$

Proof *It is directly resulted by some algebra.*

An example crystalizes the design procedure.

Example 4.3.1

Consider the benchmark problem, example 4.1.2. Consider the following sampling schedules:

$$K = [2 \quad 3] \text{ and } L = [3 \quad 2],$$

where K is the input sampling schedule and L is the output sampling schedule. With this sampling schedule, we compute the lifted system [23] in the standard setup $\underline{P}$:

$$P = \left[\begin{array}{c|cc} \underline{A} & [\underline{0} \quad \underline{B}] & \underline{B} \\ \hline \underline{C} & [\underline{0} \quad \underline{D}] & \underline{D} \\ \underline{C} & [\underline{I} \quad \underline{D}] & \underline{D} \end{array}\right]. \tag{4.29}$$

A closer look at the feedthrough term, $\underline{D}_{22}$:

$$\underline{D}_{22} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 6.03 & 28.67 & 0 & 0 & 0 \\ 65.10 & 23.28 & 33.06 & 0 & 0 \\ 44.96 & 96.28 & 6.03 & 7.41 & 0 \end{bmatrix}, \tag{4.30}$$

shows that it is not equal to zero. Therefore we need to use the alternative method, which was earlier proposed in Corollary 4.3.1 to design the controller.

We design the controller in two steps:

1. Using the procedure we explained in section 3; we design an $\mathcal{H}_2$-optimal controller for the subsystem G_1 which is subject to causality constraint.

2. Using Corollary 4.3.1, we compute the lifted controller; To proceed, we calculate K_{opt} as it was shown in (4.10). We decompose R into $R_1'R_1$ and S into S_1S_1', where matrices R_1 and S_1 are appropriate block lower triangular matrices, however, one may use a diagonal decomposition, e.g. Cholesky decomposition, see [31]. Next, we calculate $R_1L_0S_1$ and decompose the resulting matrix into $W + W_\perp$, where W and $W_\perp$ are lower and upper block-triangular matrices (4.31) and (4.32), respectively. Then, we compute the controller for the subsystem G_1. Consequently, we calculate the optimal controller $\underline{K}_{opt}$ (4.33) for the lifted system $\underline{\mathrm{G}}$ using Lemma 4.3.1.

$$W = \begin{bmatrix} 0.030 & -0.003 & 0 & 0 & 0 \\ 0.007 & 0.012 & 0 & 0 & 0 \\ -0.007 & 0.007 & 0.003 & 0 & 0 \\ -0.001 & 0.007 & 0.005 & 0.0038 & 0 \\ 0.001 & 0.002 & -0.001 & -0.0008 & -0.001 \end{bmatrix}, \tag{4.31}$$

$$W_\perp = \begin{bmatrix} 0 & 0 & -0.0020 & 0.016 & -0.001 \\ 0 & 0 & 0.0057 & 0.002 & 0.002 \\ 0 & 0 & 0 & 0.006 & 0.001 \\ 0 & 0 & 0 & 0 & 0.003 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}. \tag{4.32}$$

$$\underline{K}_{opt} = \left[\begin{array}{c|c} \underline{A}_{K_{opt}} & \underline{B}_{K_{opt}} \\ \hline \underline{C}_{K_{opt}} & \underline{D}_{K_{opt}} \end{array}\right] =$$

$$\left[\begin{array}{cccc|ccccc} 4.386 & -0.857 & 2.391 & -7.071 & -0.040 & -0.081 & 0.059 & 0.0180 & -0.0060 \\ 1.336 & 0.201 & 4.616 & -2.197 & -0.018 & -0.070 & 0.042 & 0.003 & -0.008 \\ 0.208 & -0.406 & 3.179 & -1.769 & -0.004 & -0.035 & 0.003 & 0.003 & 0.019 \\ 0.232 & -0.460 & 3.533 & -1.753 & -0.004 & -0.037 & 0.005 & 0.003 & 0.007 \\ \hline -13.259 & -8.911 & 6.736 & 5.229 & 0.089 & -0.056 & 0 & 0 & 0 \\ 5.196 & 4.288 & -47.174 & -22.678 & -0.036 & 0.329 & 0 & 0 & 0 \\ 21.687 & -1.907 & -6.394 & -35.568 & -0.108 & -0.189 & 0.171 & 0 & 0 \\ 3.586 & -11.234 & 88.123 & -38.120 & -0.091 & -0.892 & 0.277 & 0.061 & 0 \\ -3.657 & 5.045 & 37.286 & 6.014 & -0.032 & -0.389 & 0.199 & 0.026 & -0.080 \end{array}\right]. \quad (4.33)$$

Joint Multirate Design

To see the effect of the multirate design scheme on the closed-loop system performance improvement, we solve the benchmark problem in the single-rate and multirate modes. For the multi-rate case, we consider the following sampling scheme

$$K = [2 \quad 3] \text{ and } L = [3 \quad 2],$$

for the sensors and the actuators, respectively. Figure 4.10 shows the output variance versus the sampling time[5]: Observe that compared to the fast-rate setup, for the small sampling periods, the associated output variance is less than that of its fast-rate counterpart. With increasing the sampling time, the single-rate setup attains its minimum, at the point m_1, earlier than its multirate counterpart. After passing point m_1, unlike the single-rate setup, the output variance of the multirate setup continues decreasing until point m_2, where it reaches its minimum. However, after it attains its minimum, the output variance increases quickly. Observe that after the two curve cross each other at the mid point, C, fast-rate setup gives a better performance.

This behavior is justified because in both the single-rate and multirate setups, prior to achieving their minima, the network-induced noise has the dominant effect.

[5] For the multirate system, horizontal axe shows the base period, h.

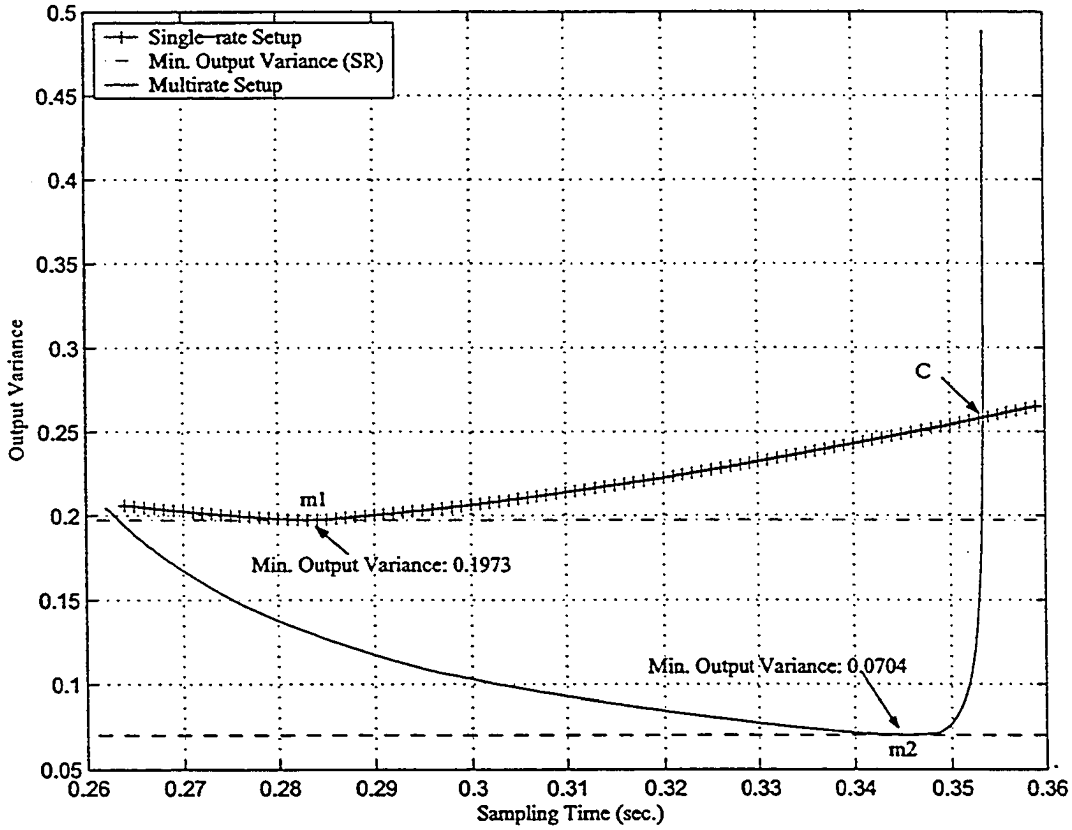

Figure 4.10: The Performance Curves of the Multirate Setup and Single-rate Setup vs. the Sampling Time.

Therefore, increasing the sampling time, introduces less noise by the network to the system and this decrease the system output variance.

4.4 Summary

Reducing the network traffic is the most effective way to ameliorate the NCS performance. In this chapter, an effective way to reduce the network communications was proposed. First a combination of optimal dual-rate controller along with the the optimal resource allocation was examined. The numerical result showed that the dual-rate output variance was lower than the fast-rate output variance. An optimal multirate controller was proposed to replace the dual-rate controller. To handle the causality constraint associated with the multirate system, a design procedure was proposed to design the optimal multirate controller. The optimal multirate controller combined with the optimal resource allocation were proposed to improve the NCS performance. Finally, performance of the proposed method was compared with its single-rate counterpart. Simulations show, for the small sampling time, the proposed method achieves a better performance.

Chapter 5

Conclusion

Replacing the traditional control systems by their network-based counterparts takes place as a consequence of the demands for the higher-performance systems in the industry, while enormous advances in the microelectronics and the telecommunications areas facilitate this adjustment. Being low-cost, easily maintained and highly flexible features NCS as a desirable choice for system designers. However, some network-inherited properties such as time-delay, communication capacity constraint and data-loss challenge the NCS superiority over the traditional digital control systems; they degrade the system performance and can destabilize the system.

Network-induced delays degrade closed-loop system performance and can destabilize the system. Depending on the choice of the protocol, the delay can be deterministic or stochastic, upper-bounded or unbounded. Communication capacity in the network environment, where the resources such as time and bandwidth are shared among several users, becomes a constraint. It results in allocating less amount of resource to each individual user. Based on the design parameters, NCS design and analysis methods are placed into three categories: network design, controller design and joint design. The joint design method is more challenging while gives higher flexibility in design, which we are also interested in.

Amongst several proposed methods to ameliorate the NCS performance, the most effective method has been shown to be decreasing the communication network traffic. We investigated the effect of the communication capacity constraint on the performance. Next, we presented a new method to reduce the network traffic and, resultantly, to attain a higher performance for the NCS.

5.1 Achieved Results

The main contributions of this thesis are highlighted below.

1. A new method, based on the dual-rate control scheme, was suggested to decrease the communication network traffic. To decrease the traffic, we suggested a dual-rate control scheme combined with the proposed optimal network resource allocation in [39]. The simulation results verify the effectiveness of the method. We show that unlike the traditional digital control system, the dual-rate control system achieves a higher performance compared to the fast-rate control system.

2. To further investigate this issue, we investigated the problem in a more general framework. We implemented a multirate control strategy and simultaneously allocated the time slots optimally. The proposed method was simulated. The results show that for the small enough sampling times, the multirate control method achieves a higher performance than the fast-rate control method. However, for the relatively large sampling times, the fast-rate control scheme shows to have a better performance.

3. The $\mathcal{H}_2$-optimal control problem for the multirate sampled-data control systems has been considered. We derived the optimal controller in the general form. The achieved controller has a state-space formulation, which has higher computational efficiency compared to the frequency domain methods. A clear procedure to derive the controller has been given. The given example crystalizes

this procedure.

4. To investigate the effect of the network capacity on the closed-loop control system performance, we simulated a wireless networked control system. The system was simulated with different network capacity. The result verifies the limited capacity as the bottleneck for the NCS design.

Bibliography

[1] Can specification version 2.0. Technical report, Robert Bosch GmbH, Postfach 30 02 40, D-70442 Stuttgart, 1991.

[2] P. Albertos and A. Crespo. Real-time control of non-uniformly sampled systems. *Control Engineering Practice*, (7):445–458, 1999.

[3] S. Boyd and C. H. Barrat. *Linear Control Design Limits of Performance.* Information and System Science Series. Prentice Hall, 1st edition, 1991.

[4] S. Boyd and L. Vandenberghe. *Convex Optimization.* Cambridge University Press, The Edinburgh Building, Cambridge, CB2 2RU, UK, 1st edition, 2004.

[5] H. Chan and Ü. Özgüner. Closed-loop control system over a communication network with queues. *International Journal of Control*, 62(3):493–510, 1995.

[6] H. Chan and Ü. Özgüner. Optimal control of systems over a communication network with queues via a jump system approach. *Proceedings of 4th IEEE Conference on Control Applications*, pages 1148–1152, 1995.

[7] T. Chen and B. Francis. *Optimal Sampled-Data Control Systems.* Springer, 1995.

[8] T. M. Cover and J. A. Thomas. *Elements of Information Theory.* Series in Telecommunications. Wiley, 1st edition, 1991.

[9] R. Gallager. D. Bertsekas. *Data Networks.* Prentice Hall, Englewood Cliffs, N.J., 2nd edition, 1992.

[10] G. G. Franklin, J. D. Powell, and A. Emami-Naeini. *Feedback Control of Dynamical Systems.* Addision-Wesley, Reading, NA, 2nd edition, 1994.

[11] R. M. Gray and D. L. Neuhoff. Quantization. *IEEE Transactions on Information Theory*, 44(6):1–30, Oct. 1998.

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

[13] A. P. B. Isle. Stability of systems with nonlinear feedback through randomly time-varying delays. *IEEE Transactions on Automatic Control*, 20:67–75, Feb. 1975.

[14] Y. Ji, H. J. Chizeck, X. Feng, and K. A. Loparo. Stability and control of discrete-time jump linear systems. *Control Theory and Advanced Technology*, 2:247–270, Jun. 1991.

[15] P. P. Khargonekar, K. Poolla, and A. Tannenbaum. Robust control of linear time-invariant plant using periodic compensation. *IEEE Transactions on Automatic Control*, 30:1088–1096, 1985.

[16] L. Kleinrock and F. A. Tobagi. Packet switching in radio channels: Part i-carrier sense multiple-access modes and their throughput-delay charactericss. *IEEE Transactions on Communications*, 23(12):1400–1416, Dec. 1975.

[17] K. C. Lee and S. Lee. Performance evaluation of switched ethernet for real-time industrial communications. *Computer Standards and Interfaces*, 24:411–423, 2002.

[18] F. L. Lian, J. R. Moyne, and D. M. Tilbury. Network design consideration for distributed control systems. *IEEE Transactions on Control Systems Technology*, 10(2):297–307, Mar. 2002.

[19] F.L. Lian, J. Moyne, and D. Tilbury. Network design consideration for distributed control systems. *IEEE Transactions on Control System Technology*, 10(2), Mar. 2002.

[20] L. Liou and A. Ray. A stochastic regulator for integrated communication and control systems: Part I- formulation of control law. *Journal of Dynamic Systems, Measurement and Control*, 113:604–611, Dec. 1991.

[21] L. Liou and A. Ray. A stochastic regulator for integrated communication and control systems: Part II- numerical analysis and simulation. *Journal of Dynamic Systems, Measurement and Control*, 113:612–619, Dec. 1991.

[22] R. Luck and A. Ray. An observer-based compensator for distributed delays. *Automatica*, 26(5):903–908, 1990.

[23] D. G. Meyer. A new class of shift-varying operators, their shift-invariant equivalents, and multirate digital systems. *IEEE Transactions on Automatic Control*, 35(4):429–433, Apr. 1990.

[24] P. G. Otanez, J. R. Moyne, and D. M. Tilbury. Using deadbands to reduce the communication in networked control systems. *Proceedings of the American Control Conference*, pages 3015–3020, 2002.

[25] P. R. Pérez-Alcázar and A. Santos. Relationship between sampling rate and quantization noise. *Proceedings of 14th International Conference on Digital Signal Processing*, 2:807 –810, Jul. 2002.

[26] Péton and J.-Ph Vial. A brief tutorial on ACCPM,. *European Journal of Operational Research*, Dec. 2001.

[27] Péton and J.-Ph Vial. A tutorial on ACCPM: user's guide for version 2.01. Technical report, Logilab,University of Geneva, 2001.

[28] A. Ray. Output feedback control under randomly varying distributed delays. *Journal Of Guidance, Control, And Dynamics*, 17(4), Jul.-Aug. 1994.

[29] K. G. Shin. Real-time communications in a computer-controlled workcell. *IEEE Transactions on Robotics and Automation*, 7(1):105–113, Feb. 1991.

[30] H. Shu and T. Chen. State-space approach to discrete-time $\mathcal{H}_2$-optimal control with a causality constraint. *System and Control Letters*, 26:69–77, 1995.

[31] S. Skogestad. *Multivariable feedback control : analysis and design / Sigurd Skogestad, Ian Postlethwaite.* Chichester ; New York : Wiley, 1996.

[32] H. T. Toivonen and P. M. Mäkilä. Computer-aided design procedure for multiobjective LQG control problems. *International Journal of Control*, 49(2):655–666, 1989.

[33] G. C. Walsh, O. Beldiman, and L. Bushnell. Asymptotic behavior of the networked control systems. *IEEE Transactions on Automatic Control*, 46(7):1093–1097, Jul. 2001.

[34] G. C. Walsh and H. Ye. Scheduling of networked control system. *IEEE Control System Magazine*, 21(1), Feb. 2001.

[35] G. C. Walsh, H. Ye, and L. G. Bushnell. Stability analysis of networked control systems. *IEEE Transactions on Control Systems Technology*, 10(3), May 2002.

[36] W. S. Wong and R. W. Brockett. System with finite communication bandwidth constraints-part I: State estimation problems. *IEEE Transactions on Automatic Control*, 42(9):1294–1299, Sept. 1997.

[37] W. S. Wong and R. W. Brockett. System with finite communication bandwidth constraints-ii: Stabilization with limited information feedback. *IEEE Transactions on Automatic Control*, 44(5):1049–1053, May 1999.

[38] L. Xiao, M. Johansson, H. Hindi, S. Boyd, and A. Goldsmith. Joint optimization of communication rates and linear systems. *Proceedings of 40th IEEE Conference on Decision and Control*, 3:2321–2326, Dec. 2001.

[39] L. Xiao, M. Johansson, H. Hindi, S. Boyd, and A. Goldsmith. Joint optimization of communication rates and linear systems. *IEEE Transactions on Automatic Control*, 48(1):148–153, Jan. 2003.

[40] P. V. Zhinvoglyadov and R. H. Middleton. Networked control design for linear systems. *Automatica*, 39:743–750, 2003.

[41] K. M. Zuberi and K. G. Shin. Scheduling message on control area network for real-time cim applications. *IEEE Transactions on Robotics and Automation*, 13(2):310–314, Apr. 1997.

Appendix

Codes

This appendix includes the written code to the numerical examples that have been provided throughout the thesis. These programs have been written in Matlab.

Mass-Spring Model

Following code belongs to the model that has been widely used throughout the thesis. This part has been written as a Matlab function.

```
function [Ac,Bc,Cc,Dc,Rw,Rv]=mass_spring_cont(n,mass,spring,damper)
%function [Ac,Bc,Cc,Dc,Rw,Rv]=mass_spring_cont(n,mass,spring,damper)
% Discrete-time LTI system: n>=2 mass-spring-dampers in series
%     x(t+1) = A*x(t) + Bu*u(t) + Bw*w(t)
%     y(t)    = C*x(t) + D*u(t)  + v(t)
% Rw and Rv are covariance matrices of w(t) and v(t).
  m = mass;
  k = spring;
  b = damper;
% continuous time dynamics
  Ac = zeros(2*n,2*n);
  for i=1:n;
      Ac(2*i-1,2*i) = 1;
  end;
  Ac(2,1:4) = [ -k(1)-k(2) -b(1)-b(2) k(2) b(2) ]/m(1);
  Ac(2*n,2*n-3:2*n) = [  k(n)  b(n) -k(n) -b(n) ]/m(n);
  for i=2:n-1;
      Ac(2*i,(2*i-3):(2*i+2))= ...
        [ k(i) b(i) -k(i)-k(i+1) -b(i)-b(i+1) k(i+1) b(i+1) ]/m(i);
  end;

  Bc = zeros(2*n,n);
  for i=1:n
      Bc(2*i,i) = 1/m(i);
  end;

  Cc = zeros(n,2*n);
  for i=1:n
%       Cc(i,2*i-1) = 1;
      Cc(i,2*i-1) = 100;
  end;
  Dc = zeros(n,n);
% Noise matrices
  Rw = (1e-6)*eye(n);
  Rv = (1e-6)*eye(n);
```

Lifted Model

Following Code has been written to compute the lifted model associated with the provided single-rate model and sampling schedules.

```
function [Am,Bm,Cm,Dm] = multirate(Af,Bf,Cf,Df,N,K,L)
%
%  [Am,Bm,Cm,Dm] = multirate(Af,Bf,Cf,Df,N,K,L)
%  [Af, Bf, Cf, Df] = c2d(Ac,Bc,Cc,Dc,'h')   (fast-rate model)
%  Given the fast rate, sampling rate and the general
%  sampling rate it calculates the multirate model.
%
%  N = LCM(L1,L2,...,Lp,K1,K2,...,Km)
%  L1,L2,...,Lp -->  sampling rates
%  K1,K2,...,Km -->  holding rates
%
%  Example
%  Sampling rates h, 2h, 4h --> L = [1 2 4]
%  Hold rates h, 3h, h      --> K = [1 3 1]
%  N = LCM {1, 2 , 3, 4}   --> N = 12
%
P = N./L;
M = N./K;
nI = size(Bf,2);  % number of Inputs (fastrate model)
nO = size(Cf,1);  % number of Outputs (fastrate model)
% computing Am
    Am = Af^N;
% computing Bm
Bm=[ ];
for j=1:nI
Mj=M(j);
Bj=[ ];
for alpha=1:Mj
sum=0;
for q=1:(N/Mj)
sum = sum+ (Af^(N-q-(alpha-1)*N/Mj))*Bf(:,j);
end
Bj=[Bj sum];
end
Bm= [Bm Bj];
end
%computing Cm
```

```
Cm=[ ];
for i=1:nO
Pi=P(i);
Ci=[ ];
sum=0;
for qq=1:Pi
sum = Cf(i,:)*Af^((qq-1)*N/Pi);
Ci=[Ci;sum];
end
Cm=[Cm;Ci];
end
% computing Dm
Dm=zeros(size(Cm,1),size(Bm,2));
pCount = 0;
for i=1:nO
Pi=P(i);
Ci=Cf(i,:);
mCount = 0;
for j=1:nI
Mj=M(j);
Bj=Bf(:,j);
Dij =Df(i,j);
Dmij = [ ];
for alpha=1:Pi
for beta = 1:Mj
sum = [ ];
sum1 =0;  %% modified
r = (alpha-1)*N/Pi-(beta-1)*N/Mj;
sum1 = Dij*mtheta(N/Mj-1-r)*mtheta(r);
sum2 = 0;
for q=0:(N/Mj)-1
sum2 = sum2 + mtheta(r-q-1)*Ci*(Af^q)*mssay(r-N/Mj,Af)*Bj;
end
sum = sum1 + sum2;
Dm(pCount + alpha, mCount + beta) = sum;
end
end
mCount =mCount + Mj;
end
pCount = pCount + Pi;
end
[Dm] = holdcorrection(Dm,K,N);
```

```
[Bm] = holdcorrection(Bm,K,N);
[Dm]=samplecorrection(Dm,L,N);
[Cm]=samplecorrection(Cm,L,N);
```

Multirate Controller

Following Code has been used to compute the Multirate controller in Chapter 4.

```
%% Discrete Multirate H2-optimal Control Design
clc;
clear all;
CCost = [ ];
Cost=[ ];
RowCost = [ ];
% We take the model
n = 2;  % number of masses adn springs
mass   = [10; 50];
spring = [1; 1];
damper = [0;0];
[Ac,Bc,Cc,Dc,Rw,Rv]=mass_spring_cont(n,mass,spring,damper);
 % Continuous-time model
Ts = (1/max( abs( eig(Ac) )) )*2*pi/10; % Sampling-Time Ts =0.71
sysd=c2d(ss(Ac,Bc,Cc,Dc),Ts,'zoh'); % Discrete-Model
clear mass spring damper;
Add=sysd.a;
Bdd=sysd.b;
Cdd=sysd.c;
Ddd=sysd.d;
[Ad,Bd,Cd,Dd] = multirate(Add,Bdd,Cdd,Ddd,6,[2 3 ],[3 2]);
%---------   standard formulation
b0 = Ts*.22*(10)*log2(1+10/(10)/0.1);
Wnoise = 2^(-2*b0)*1e-0*eye(5);  %Network-induced Noise
Vnoise = 2^(-2*b0)*1e-0*eye(5);  %Network-induced Noise
epsilon = 0.02;
A=Ad;
B1=[zeros(size(Bd)),Bd*Wnoise];
B2=Bd;
C1=Cd;
C2=Cd;
D11 = [zeros(size(C1,1)) Dd*Wnoise];
D12 = Dd
D12(1,[1 2 5])=10;
D12(2,1:2)=8;
D21 = [eye(size(C2,1))*Vnoise,Dd*Wnoise];
D22 = Dd;
%---------------- H2 Optimal Control Design Output Feedback
M = D12'*D12;
```

```
invM = inv(M);
Axric = A-B2 * invM * D12' * C1;
Qxric = C1'*((1+0.02)*eye(size(C1,1))-D12*invM*D12')*C1;
Qxric = floor(Qxric*1e7)*1e-7; % To force it to be symmetrical
Rxric = inv(B2*invM*B2' + epsilon * eye(size(B2,1)));
Rxric = floor(Rxric*1e7)*1e-7;
Bxric = eye(size(Rxric,1));
[X,L,G,RR] = DARE(Axric,Bxric,Qxric,Rxric);
clear L G RR;
F  = -inv(M+B2'*X*B2)*(B2'*X*A+D12'*C1);
F0 = -inv(M+B2'*X*B2)*(B2'*X*B1+D12'*D11);
%------%-----%
mEpsilon = 0;
N = (D21+mEpsilon*ones(size(D21)))*(D21+mEpsilon*ones(size(D21)))';
invN = (floor(inv(N)*1e7)/1e7);
Ayric = (A-B1*D21'*invN*C2)';
Qyric = B1*(eye(length(D21))-D21'*invN*D21)*B1';
Ryric = inv(C2'*invN*C2+epsilon* eye(size(C2,2)));
Byric = eye(size(Ryric,1));
[Y,L,G,RR] = DARE(Ayric,Byric,Qyric,Ryric);
clear L G RR;
L = -(A*Y*C2'+B1*D21')* inv(N+C2*Y*C2');
L0 = -(F*Y*C2'+F0*D21')* inv(N+C2*Y*C2');
%-- Controller --
Ak = A+B2*F+L*C2-B2*L0*C2;
Bk = L-B2*L0;
Ck = L0*C2 - F;
Dk = L0;
%%% Closed-Loop System
Acl = [A+B2*Dk*C2 B2*Ck;
       Bk*C2      Ak ];
Bcl = [B1+B2*Dk*D21;
       Bk*D21];
Ccl = [C1+D12*Dk*C2  D12*Ck];
Dcl = [D11+D12*Dk*D21];
%% Computing the cost function
Qlyap = Bcl*Bcl';
Llyap = dlyap(Acl,Qlyap);
sqrt(trace (Dcl*Dcl'+Ccl*Llyap*Ccl'));
%% With causality constraint
Rcaus = M+B2'*X*B2;
RcausChol = chol(Rcaus);
RcausChol = RcausChol';
Scaus = N+C2*Y*C2';
ScausChol = chol(Scaus);
ScausChol = ScausChol';
W = RcausChol*L0*ScausChol;
% decomposing
W_ = zeros(size(W));
W_ (1,3:5)=W(1,3:5);
W_ (2,3:5)=W(2,3:5);
W_ (3,4:5)=W(3,4:5);
W_ (4,5)=W(4,5);
V = L0 - inv(RcausChol)*W_*inv(ScausChol);
Akm = A+B2*F+L*C2-B2*V*C2;
Bkm = -L+B2*V;
Ckm = F-V*C2;
Dkm = V;
%% closed-loop multirate
Aclm = [A+B2*Dkm*C2 B2*Ckm;
       Bkm*C2      Akm ];
Bclm = [B1+B2*Dkm*D21;
       Bkm*D21];
Cclm = [C1+D12*Dkm*C2  D12*Ckm];
Dclm = [D11+D12*Dkm*D21];
%% Computing the cost function
Qlyap = Bclm*Bclm';
Llyap = dlyap(Aclm,Qlyap);
sqrt(trace(Dclm*Dclm'+Cclm*Llyap*Cclm'));
Cost=[Cost,sqrt(trace(Dclm*Dclm'+Cclm*Llyap*Cclm'))];
% Akm
invTemp = inv(eye(size(Dkm*D22))+Dkm*D22);
Akmfinal = Akm-Bkm*D22*invTemp*Ckm;
Bkmfinal = Bkm-Bkm*D22*invTemp*Dkm;
Ckmfinal = invTemp * Ckm;
Dkmfinal = invTemp * Dkm;
Dkmfinal = floor(Dkmfinal*1e7)*1e-7;
%% closed-loop multirate
Aclmf = [A+B2*Dkmfinal*C2 B2*Ckmfinal;
       Bkmfinal*C2      Akmfinal ];
Bclmf = [B1+B2*Dkmfinal*D21;
       Bkmfinal*D21];
Cclmf = [C1+D12*Dkmfinal*C2  D12*Ckmfinal];
Dclmf = [D11+D12*Dkmfinal*D21];
```

```
%% Computing the cost function
LenInp = size(Bclmf,2);
RowCostTemp = [ ];
for InpCnt= 1:LenInp
Qlyap = Bclmf(:,InpCnt)*Bclmf(:,InpCnt)';
Llyap = dlyap(Aclmf,Qlyap);
RowCostTemp =[RowCostTemp,abs(sqrt(trace (Cclmf*Llyap*Cclmf')))];
end
RowCost = [RowCost;RowCostTemp];
 CCost = [CCost,sqrt(sum(RowCostTemp.^2))];
%%%%%%%%%%%
function [ssay] = mssay(q,Af)
%
% [ssay] = mssay(q,Af)
ssay = 1;
if q >=0
    ssay = Af^q;
end
%%%%%%%%
function [theta] = mtheta(q)
%
% [theta] = mtheta(q)
theta = 0;
if q>=0
    theta = 1;
end;
%%%%%%%%%%%%%%%%%
function [Dmt] = samplecorrection(Dm,L,N)
%
% [Dmt] = samplecorrection(Dm,K,N)
% K  sample rate
% N = lcm (K, L)
Dmt = Dm;
InputLength = sum(N./L);
InputOrder = [ ];
for i = 1:length(L)
   InputOrder=[InputOrder,1:L(i):N]
end
InputOrder = [InputOrder;1:InputLength]
for i = 2:InputLength-1
    for j = i+1:InputLength
    if InputOrder(1,i) > InputOrder(1,j)
    InputOrder = InputOrder(:,[1:i-1,j,i+1:j-1,i,j+1:InputLength])
    end
    end
end
InputOrder = InputOrder(2,:)
Dmt = Dm(InputOrder,:)
%%%%%%%%%%%
```