KTH Electrical Engineering

Modelling, Analysis and Experimentation of a Simple Feedback Scheme for Error Correction Control

OSCAR FLÄRDH

Licentiate Thesis
Stockholm, Sweden 2007

Automatic Control Lab
School of Electrical Engineering
Royal Institute of Technology (KTH)
SE-100 44 Stockholm
SWEDEN

TRITA-EE 2007:009
ISSN 1653-5146
ISBN 978-91-7178-586-2

© Oscar Flärdh, 2007

Tryck: Universitetsservice US AB

Abstract

Data networks are an important part in an increasing number of applications with real-time and reliability requirements. To meet these demands a variety of approaches have been proposed. Forward error correction, which adds redundancy to the communicated data, is one of them. However, the redundancy occupies communication bandwidth, so it is desirable to control the amount of redundancy in order to achieve high reliability without adding excessive communication delay. The main contribution of the thesis is to formulate the problem of adjusting the redundancy in a control framework, which enables the dynamic properties of error correction control to be analyzed using control theory. The trade-off between application quality and resource usage is captured by introducing an optimal control problem. Its dependence on the knowledge of the network state at the transmission side is discussed. An error correction controller that optimizes the amount of redundancy without relying on network state information is presented. This is achieved by utilizing an extremum seeking control algorithm to optimize the cost function. Models with varying complexity of the resulting feedback system are presented and analyzed. Conditions for convergence are given. Multiple-input describing function analysis is used to examine periodic solutions. The results are illustrated through computer simulations and experiments on a wireless sensor network.

Acknowledgement

The work in this thesis was partially financed by the Swedish Research Council, by the Swedish Foundation for Strategic Research through an Individual Grant for the Advancement of Research Leaders, and the European projects RUNES and HYCON. Their support is gratefully acknowledged.

While performing the research presented in this thesis, I've had lots of people that contributed in various ways. Some of them deserves an extra mentioning.

First of all, I would like to thank my advisor Karl H Johansson for giving me the opportunity to become his PhD student. I am very grateful for all your guidance and help that made me accomplish this thesis, and also for letting me learn by trying on my own. Your infectious enthusiasm and dedication to academic research is a true inspiration. My co-advisor Mikael Johansson has provided many ideas and suggestions for my research, as well as co-authoring my papers. I am very thankful for that.

Carlo Fischione deserves many thanks for sharing his extensive knowledge in wireless communication, and also co-authoring two of the papers this thesis is based on. Thanks also to master's thesis student Per-Erik Björkestad and research engineer Pan Gun Park for implementing and running the error control mechanism on the sensor motes test bed.

Thanks to Carlo Fishione, Björn Johansson, Karl H Johansson and Mikael Johansson for proofreading different parts of this thesis. The administrator at the control group, Karin Karlsson-Eklund, is always very helpful with whatever kind of administrative issue I have. Thanks for all the help.

I would also like to thank all my colleagues, former and present, for all interesting discussions and suggestions for improvement. You have also contributed with a very nice and positive atmosphere at the department.

Last but not least, I would like to thank my parents, Sylvia and Peter, and my brother, Jacob, for their incessant love and support.

Stockholm, February 2007
Oscar Flärdh

Contents

Chapter 1

Introduction

During the last decades the rapid development of computers and networks has been revolutionary for our approach to communication. The Internet and the cellular networks are probably the ones most known, but networks are also ubiquitous in cars, manufacturing industries etc. This rapid increase in communication possibilities has enabled new economic opportunities for many companies, and efficient and reliable communication is today a crucial success factor in many businesses. Emerging applications include those that need guarantees for limited delays or losses, *e.g.*, wireless automation and media streaming. Following this development, huge research efforts has been made in order to understand these systems and controlling their behavior. A trend where networks go from best-effort to networks that supports quality of service, QoS, mechanisms can bee seen. Best-effort networks, like UDP, give no guarantee for delivery and there is no mechanism to prioritize between different data flows. The term QoS refers to mechanisms that may prioritize certain data flows which may for example guarantee a minimum bandwidth or maximum delay. An example of the trend towards QoS is the third generation model network, UMTS, which supports four different classes of quality of service.

In a communication network, there is some probability that the data communicated are lost. Depending on the causes of the losses, and the application using the communication link, there are different ways to handle the situation. If the losses are due to congestion, rate control is the most suitable way to decrease the losses. This may be combined with retransmissions, as in TCP, which is beneficial if delays can be tolerated. For wireless links, losses are often caused by interference from other systems and bad channel conditions. In interference limited situations, power control can be used to reduce the errors and possibly also increase the capacity. If the channel conditions are bad, then channel coding is a suitable way to handle the losses. Channel coding refers to the concept of adding redundant information to the transmitted data. From this redundancy, the receiver can recover the original message even in presence of losses.

This thesis focuses on the use of redundancy for dealing with lossy links. The key advantage is that messages can be decoded reliably even if some parts of the message are lost during transmission. The drawback, on the other hand, is that load of the network is increased due to the added redundant information. Since network resources are often constrained, it is important not to use excessive redundancy. This raises the need to control the amount of redundancy to the current network conditions, which is sometimes referred to as adaptive forward error correction. In forward error correction the redundancy is decided before the message is sent and appended to the application data. This is in contrast to backward error correction, where extra redundancy is sent on request from the receiver. The advantage of forward error correction is the ability to recover from error immediately when the message is received. Thus it is more suitable for applications with real-time constraints, while backward error correction is better from a link utilization view. This thesis only deals with forward error correction, and in the following "forward" will be omitted to facilitate the reading.

The goal of error correction schemes is to find the best compromise between adding enough redundancy to achieve high probability of recovering sent messages and limiting the amount of redundant traffic load on the network. Mostly, this has been done through utilization maximization based on network traffic models, but it can also be formalized as a feedback control problem where a cost function is minimized. This cost function should penalize both high block error rates and high amount of redundancy. The best compromise is then the amount of redundancy that achieves the lowest cost. With varying network conditions also the cost function, and thereby the optimal amount of redundancy, will vary with time.

From a control theoretic perspective, two main approaches to find the optimal redundancy can be found. The first one, which is totally dominating in the current research, is based on models of how packets are lost on the network. By developing such models, it is possible to calculate the cost function and, from that, find the optimal redundancy. This approach also demands an on-line identification of the model parameters. There are two major drawbacks with this approach; the dependence on an accurate loss model and the need for efficient parameter estimation. If the model or the estimated parameters, or possibly both, are incorrect, the cost function will not reflect the actual network situation, resulting in suboptimal performance. This thesis, therefore, proposes a second approach, which uses feedback information from the success of the ongoing error correction process. This approach does not rely on knowing the shape of the cost function, but uses extremum seeking control to find the optimum of an unknown function. Since the function does not need to be known, neither a correct loss model nor accurate parameter estimates are needed.

The rest of the introduction contains two motivating examples to illustrate and show the relevance of the studied problem. They are followed by the problem formulation and a summary of the contributions, with references to previous publications of the results. The introduction is concluded with a presentation of the outline of the thesis.

1.1 Motivating Examples

To illustrate the concepts introduced, two examples are given. First, a possible application of control over a wireless sensor network is presented. In a manufacturing building the air is to be controlled in such way that the workers, and possibly also the manufacturing process, do not suffer.

Example 1.1
In Figure 1.1 a sketch of a wireless air quality control system for the manufacturing building is shown. It could be a smelting plant, foundry or some other factory with machines that causes pollution and extreme temperatures as well as a chemical process with high demands on the ambient conditions. There are sensors, s, spread out over the building collecting information on conditions such as temperature, oxygen content or humidity. This information is communicated to the controller, which uses heaters, $\mathcal{H}$, air conditioners, $\mathcal{A}$, and fresh-air intakes, $\mathcal{F}$, to achieve desired conditions on the air quality. The controller may also use the sensor data for monitoring the manufacturing process and detect leaks, machine malfunctions or other possibly hazardous events. In such cases it is important to quickly take proper counteractions, for example shutting down the production.
There are several reasons for having a wireless instead of wired system. First, it may be cheaper to have wireless communication than buying and installing possibly several kilometers of cable in a large building. Another reason could be that it is impossible to install sensors or actuators in the desired way if they need to be connected to a cable. Also, a wireless system does not suffer from problems with worn and possibly broken cables.
For this system it is very important to have reliable communications both from sensors to the controller and from the controller to the actuators. Not only lost but also delayed information may cause the control loop and the monitoring systems to perform badly. Hence there is a need for an application that allows the control system to have reliable communication despite the possibly harsh conditions for the wireless links. In this scenario, it is often difficult or impossible to obtain accurate information on the conditions of the wireless channel. Indeed, several moving metal objects, and electromagnetic interference, may prevent the use reliable analytical models packet losses (Willig et al., 2005).

The next example illustrates error correction. The tradeoff between applied redundancy and resource utilization is demonstrated.

Example 1.2
Consider the setup as in Figure 1.2, where a message is sent over a network. The message is divided into blocks, which in turn consists of several packets. Before transmitting the packets with application data they are encoded by an encoder, $\mathcal{E}$. This adds redundancy packets into the block. All packets are then sent one by one over the network, and during the transmission a packet may be lost or delayed.

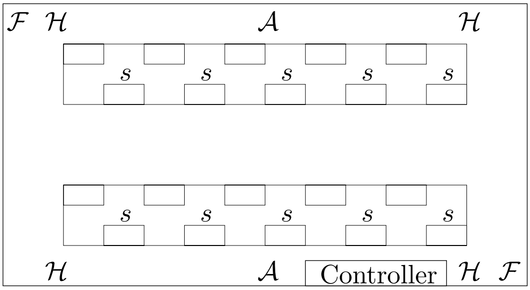

Figure 1.1: An illustration of wireless sensing and control of the air in a manufacturing building. The sensors s transmits information on, *e.g.*, temperature and oxygen content to the controller. With help of heaters, air-conditioners, and fresh-air intakes the air in the building can be controlled.

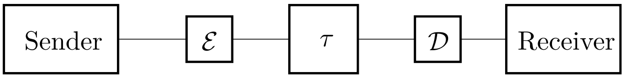

Figure 1.2: This scheme illustrates the setup where a sender transmits a stream of packets over a network. The packets are encoded into blocks, and sent over an unreliable link, *i.e.*, the packets may be lost or delayed during transmission. When all packets in a block are received, it is decoded.

The received packets are decoded by the decoder, $\mathcal{D}$, and if enough packets in the block are successfully received the block will be decoded without error.
A scheme like this is run over a link between two wireless sensor nodes, see Figure 1.3. One node act as the sender, and the other as the receiver.[1] The effect of using redundancy to handle the losses can be found through a simple experiment. By sending a message over a lossy network where in the first part of the transmission no redundancy is added. In the second part redundancy is added, which increases the reliability of the transmission. In Figure 1.4, the number of lost packets after decoding on the receiver side are shown. At block 500, redundancy is introduced causing the amount of lost packets to decrease significantly. However, the increased reliability that the redundancy accomplishes has the drawback that it reduces the transmission rate of the application data packets. This phenomena is illustrated in Figure 1.5, where the number of recovered data packets after decoding decrease slightly when the redundancy is introduced. The reason for this is that some of the bandwidth now is used for redundancy packets. The figure also illustrates the

[1]Instead of implementing the encoding and decoding algorithms, every packet is marked as either redundancy or application data. With knowledge on the characteristics of the code it is then possible to conclude on the outcome of the error correction.

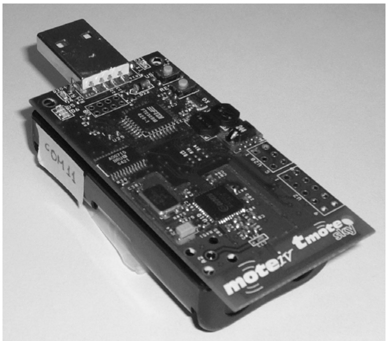

Figure 1.3: A Telos sensor mote. Error correction is illustrated with sending packets between two such motes. The first part of the experiment, the error correction is turned off. During the second half of the experiment redundancy is introduced, unveiling the characteristics of error correction.

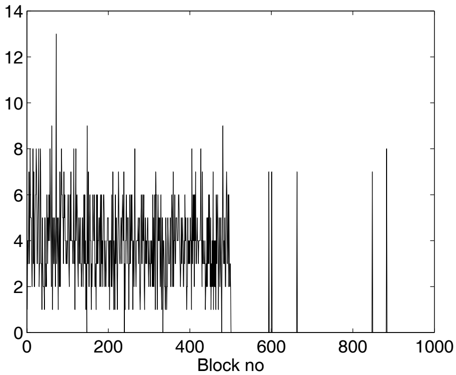

Figure 1.4: The number of distorted application data packets per block after decoding. At 500 blocks redundancy is introduced, decreasing the number of distorted packets significantly. The data is from an experiment on a network with two nodes.

benefits in a nice way; the error correction enables a stable and predictable data rate. Using too much redundancy would indeed increase the predictability, but in the same time decrease the throughput. Reducing the redundancy, however, will reduce the reliability and predictability.

These two examples motivate the need for reliable wireless communication and also demonstrate the ability of error correction to achieve this. They have also

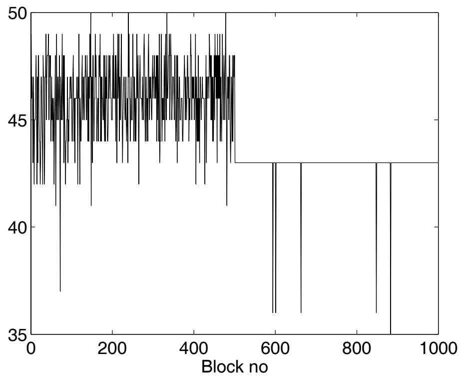

Figure 1.5: The number of recovered application data packets per block after decoding. The introduction of the redundancy enables a stable and predictable data flow to the price of lower transmission rate.

stressed the importance of using the correct amount of redundancy.

1.2 Problem Formulation

The problem studied in this thesis is how to chose the right amount of redundancy in order to compensate for varying network conditions. The focus is thus on handling losses that are due to bad channel conditions, *e.g.*, attenuation or shadowing. Several adaptive error correction schemes[2] has been proposed in the literature, but many of them rely on a model of the network loss process. Based on this model the optimal amount of redundancy can be found given an objective function. The drawback with this model reliance is that the network loss process can never be exactly known, possibly causing suboptimal performance. The purpose of this thesis is, therefore, to present a control theory approach to this problem that aims to provide robust error control without relying on any loss model.

1.3 Contributions

The main contribution of the thesis is to formulate the problem of adjusting the redundancy in a control framework, which enables the dynamic properties of feedback error correction control to be analyzed using control theory. The idea of using

[2]The word “adaptive”, in network context, usually means that a signal is not constant but changes according to some rules. In control theory, however, “adaptive” means that the parameters in the controller and/or the model are updated online. To avoid confusion, the term “adaptive error correction” will from now on be avoided in favor for “error correction control”.

feedback from the outcome of the error correction process was initially presented in

> **Flärdh, O., Johansson, K. H. and Johansson, M.**: A Comparison of Control Structures for Error Correction in Packet-Switched Networks, in *IEEE INFOCOM* Student Workshop, Miami, USA, March 14, 2005.

and extended with the design and investigation of an extremum seeking feedback controller in

> **Flärdh, O., Johansson, K. H. and Johansson, M.**: A New Feedback Control Mechanism for Error Correction in Packet-Switched Networks, in *44th IEEE Conference on Decision and Control and European Control Conference*, Seville, Spain, December 12-15, 2005.

A thorough analysis of the control structure, introduction of the cost function and error control for more advanced coding schemes are found in

> **Flärdh, O., Fischione, C., Johansson, K. H. and Johansson, M.**: A Control Framework for Online Error Control Adaptation in Networked Applications, in *IEEE Second International Symposium on Communications, Control and Signal Processing*, Marrakech, Morocco, March 13-15, 2006.

The theoretical analysis presented in Chapter 4 is based on

> **Flärdh, O., Fischione, C., Johansson, K. H. and Johansson, M.**: Analysis of a Simple Feedback Scheme for Error Correction over a Lossy Network, in *IEEE Second International Conference on Networking, Sensing and Control*, London, UK, April 15-17, 2007.

In these joint papers, the author of this thesis has been the main contributor. Both in writing the papers and performing the research presented in them.

1.4 Outline

The rest of the thesis is organized as follows. In Chapter 2 background theory on estimation and control over networks, extremum seeking control and error correction coding is given. Chapter 3 contains the model of the error correction process as well as a discussion on control architecture and presentation of the extremum seeking feedback controller. A theoretical analysis of the presented controller is found in Chapter 4 together with some examples to illustrate the results. The control setup is then investigated through packet-based simulations and experiments on a teal sensor network in Chapter 5. Finally, Chapter 6 summarizes the thesis and presents directions for future research.

Chapter 2

Background

This chapter gives the background to the concepts covered in this thesis. First, we will discuss the the general problem of performing estimation and control over unreliable networks. This issue has received substantial attention in the last years, and several novel schemes for handling lost information on the feedback channel has been presented. Most of them, however, concentrate on modifying the control and estimator algorithms to handle them. Another approach is to develop interfaces between the network and the application to sustain the performance in presence of losses. One way to do this is by using error correction codes, which is presented in the next section. This also includes an introduction to the error correction control problem. Finally, an overview of extremum seeking control is given.

2.1 Estimation and Control over Lossy Networks

If the plant and the controller are connected via a network, see Figure 2.1, it is important to consider the effects of the delays and drops that the network may introduce. For a robust controller, occasionally delays or drops can probably be handled. If there is a significant amount of drops though, it must be coped with in some way. Hence there is a need for mechanisms handling packet drops in control applications running over wireless networks. Estimation of a system with limited bandwidth is studied in (Wong and Brockett, 1997) and extended to feedback control in (Wong and Brockett, 1999). A filter design for compensating for drops on the feedback channel was developed by Ling and Lemmon (2004). Sinopoli et al. (2004) consider Kalman filter in the presence of independent losses of the measurement and give bound on minimum arrival probability of measurements for the filter to be stable. For networks with feedback channel the separation principle is shown to hold, giving the optimal controller as linear feedback from observed states (Schenato et al., 2006). Joint design of controller and quantize encoding are presented in (Tatikonda and Mitter, 2004). Relations between control objectives, like observability and stability, and minimum allowable rate on the feedback chan-

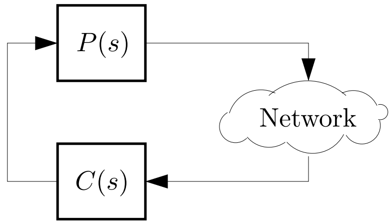

Figure 2.1: The feedback from the plant $P(s)$ is communicated to the controller $C(s)$ over a network. This causes the feedback information to be delayed and possibly also dropped, which imposes difficulties for the estimation and control.

nel are also given. Gupta et al. (2006) also uses the LQG framework for analyzing feedback control over networks. Analysis of stability and performance of the closed loop system in the special case with independent losses is also performed, giving, *e.g.*, bounds on loss probability for guaranteed stability.

2.2 Error Correction Coding

The concept of error correction coding is described in this section. In error correction coding, the sender adds redundant information to the original data before transmission. This allows the receiver to recover the original data through decoding of the received packets, if a sufficient amount of the total packets arrives. The previous presented ways of handling network losses all have in common that they handle the uncertainties and variations from the network by modifying the existing applications. By changing the estimation or control algorithms, the performance of the application is preserved. Another approach is to develop interfaces between the application and the network, and in that way preserve the application performance. Error correction coding is one way to support control and other real time applications running over networks.

2.2.1 Coding

The concept of coding originates in 1948 with the famous paper by Shannon (1948). It stated that all communication rates below capacity is achievable with zero probability of error, and that channel codes could be used for that. A (n, k) channel code takes a batch of length k bits and encodes it into a new batch of length $n > k$ bits. This new batch is called a codeword, and all codewords in a code is denoted the alphabet of the code. The batch with length n is then sent over the network channel and then decoded back to a batch of k bits by the decoder. This is shown in Figure 2.2. Only end-to-end error correction is considered, *i.e.*, the path from sender to receiver is seen as one channel with certain characteristics. With help of

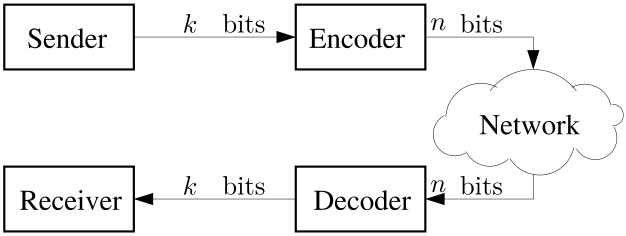

Figure 2.2: An information batch of k bits are encoded into a new batch with n bits and sent over the network. At the receiving side, it is decoded to a batch of length k bits.

Source	Codeword	Received	Decoded as	Outcome
0	00	00	0	DC
	00	01	x	DE
	00	10	x	DE
	00	11	1	DF
1	11	11	1	DC
	11	01	x	DE
	11	10	x	DE
	11	00	0	DF

Table 2.1: The source transmits 0 or 1, which is encoded as 00 or 11. The received message is then decoded. Depending on the errors during transmission the decoder may decode correctly (DC), detect an error (DE) or decode false (DF).

this coding, errors can be detected and/or corrected. A simple example illustrates this.

Example 2.1
Consider the transmission of a single bit over a channel, *i.e.*, the sender transmits either 1 or 0. A 1 is decoded as 11, and 0 as 00. Hence the alphabet of the code is $\{11, 00\}$. The receiver decodes 00 as 0 and 11 as 1, but when 01 or 10 is received it concludes that there has been an error in the transmission. Table 2.1 shows the possibilities during the encoding-decoding process. From the table it can be concluded that this code can detect one error, but not correct any errors. If there are two errors during the transmission, it will decode with error. To correct errors, longer codewords would be needed.

The example shows that the difference between the codewords are important for the code's performance. Hamming (1950) formalized this and concluded that the number of positions that the codewords differ is the important property. This number is called the Hamming distance, and the smallest Hamming distance between any two codewords in the alphabet of a code is called the minimum distance, δ, of the code. The minimum distance of the code in Example 2.1 is thus 2, since 00 and 11 differ at two positions.

A code can detect $\delta - 1$ errors and correct $\lfloor (\delta - 1)/2 \rfloor$ errors. It can be shown that for a (n, k) code the minimum distance is bounded by

$$\delta \leq n - k + 1 \tag{2.1}$$

A code that achieves this bound is called a maximum distance separable code. An example of such a code is the Reed-Solomon code (MacWilliams and Sloane, 2003).

An error correction code can also be used for erasure correction. In erasure correction the receiver may determine a position in the received block as erased, meaning that it doesn't assign any value for that position (before decoding). An error correction code can correct $\delta - 1$ erasures. Another important property of the code is that it can be systematic, which means that the original data is not changed. Hence, only u bits are added to form the codeword.

The theory described here holds for linear binary codes, but is also valid for most linear codes. For more results on error correction codes and generalizations to other fields than the binary see, *e.g.*, (Blahut, 1983) or (MacWilliams and Sloane, 2003) as well as (Cover and Thomas, 1991) for an information theoretic background.

2.2.2 Error Correction on Packet Level

When coding is performed on packet level the information in several packets are encoded, creating new packets containing redundancy. An example of systematic error correction codes on packet level are shown in Figure 2.3. The message that is to be transmitted is divided into packets and sent over the network. But each packet is also stored in a frame at the sender, corresponding to one row in the figure. When k application data packets have been sent, each column is filled with redundancy bits to form a codeword. This will generate u redundancy packets which also will be sent over the network. Since the application data packets can be sent directly without needing to be decoded there will be no delay of the application data. The collection of the n packets encoded together will be referred to as a block.

When the packets in a block are sent over the network, some of them might be lost. If the packets are marked with, *e.g.*, sequence numbers it is possible for the receiver to know which packets are missing and thus perform the more powerful erasure correction. By using a maximum distance separable code, it follows from (2.1) that up to u packets losses can be corrected. Figure 2.4 shows an example of this. If at least any k packets of the block is successfully received, it is possible to decode the original information without error.

Hence coding is not a separate protocol, but rather a method to improve the performance of the communication without involving the protocol. Instead it can be viewed as a protocol booster, which are modules for improve the performance

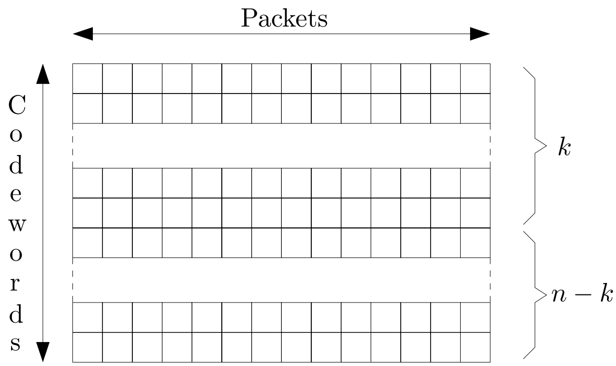

Figure 2.3: Systematic error correction coding. In the block with n packets, each row corresponds to a packet and each column to a codeword. The k first packets contain application data, whereas the last u are redundancy packets.

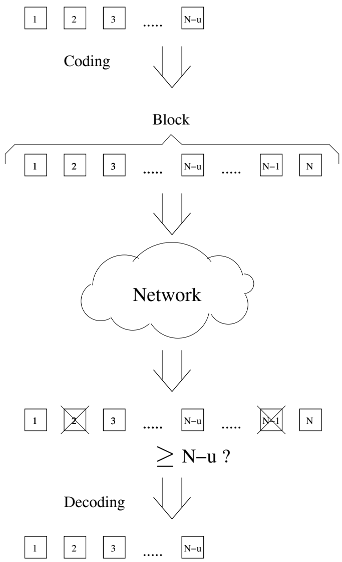

Figure 2.4: Decoding of the packets. If at most u out of n sent packets in a block are lost, then it is possible to recover the original k data packets of the block.

of a protocol but still being transparent to it (Feldmeier et al., 1998). The main advantage is that they enable fast evolution and customization of existing protocols, compared to changing or replacing the protocol itself.

2.2.3 Error Correction Control

The problem of choosing a suitable amount of redundancy for error correction in packet-switched networks has extensive interest in the literature in the last decade. One driving force for this is the rapidly increasing number of applications utilizing media streaming over the internet. An early presentation of packet-level error correction is (Shacham and McKenny, 1990), which however did not consider the problem of deciding the amount of redundancy. Many studies have then focused on investigating the effect of error correction for different network conditions as well as various redundancy levels. Most studies show encouraging results, *e.g.*, (Lundqvist and Karlsson, 2005), (Lundqvist and Karlsson, 2004), (Barakat and Al Fawal, 2004) and (Busse et al., 2006), but some studies point out that highly correlated losses impose difficulties for the error correction algorithms, *e.g.*, (Bolot et al., 1995) and (Yu et al., 2005).

There are several control schemes presented where the redundancy is controlled. Most of them rely on a model of or knowledge on the network loss process. Models for the packet losses include independent as well as correlated losses, *e.g.*, (Baldantoni et al., 2004), (Waldby, 2001) and (Bolot et al., 1999), and also algorithms for predicting packet losses, *e.g.*, (Roychoudhuri and Al-Shaer, 2004). Other proposed controllers estimate certain properties of the packet loss probability, *e.g.*, (Chakareski and Chou, 2004) and (Du and Zhang, 2006).

An error control mechanism that does not rely on network state information was presented by Park and Wang (1998). They consider the problem of maximizing the recovery rate γ, defined as the number of received and recovered packets in a block within a certain time. Moreover, the relation between the amount of redundancy and the recovery rate is assumed to be a unimodal function. A controller for maximizing the recovery rate is also proposed, which though gives the maximum recovery rate γ^* as an unstable equilibria of the feedback system. Thus, target recovery rate γ_* is introduced and if $\gamma_* < \gamma^*$ the system will have γ_* as a stable equilibria. A gradient estimator is added to make the equilibria globally stable. A drawback with this approach is that γ^*, or a lower bound of it, must be known.

Inspired by the algorithm of Park and Wang (1998), Flärdh et al. (2005) proposed an extremum seeking controller for error correction that does not rely on network information. The controller is described in Chapter 3 and analyzed in Chapter 4.

2.3 Extremum Seeking Control

Extremum seeking control, or sometimes referred to as just extremum control, is a special type of control problems in which an output $y(t)$ is to be minimized or max-

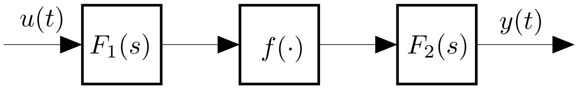

Figure 2.5: A general extremum seeking control system with dynamics both on the input and the output. The nonlinearity $f(\cdot)$ should have at least one extremum point.

imized. This is in contrast to standard control problems, where there is a reference that is to be followed. The block diagram in Figure 2.5 shows a system for which extremum seeking control is applicable. The nonlinear function $f(\cdot)$ may be unknown, but has one or several extremum points. Extremum control is a classical approach to control, which only recently has been studied with some theoretical depth. The first publication is from 1922 (Leblanc, 1922) but it took another 30 years until the extremum control problem was addressed in a larger extent. Important publications include (Draper and Li, 1951), (Morosanov, 1957), (Ostrivskii, 1957) and (Blackman, 1962), but the difficulties in performing theoretical analysis reduced the interest. Sternby (1979) presents a nice survey of the theoretical developments and describes common applications of extremum control such as combustion processes, and blade adjustments in turbines and wind mills. Due to increased demand of online optimization as well as theoretical advances within adaptive control, the interest in extremum control has gained significantly the last decade. An extremum controller for a class of nonlinear systems was presented and shown to be locally stable in (Krstić and Wang, 2000). Similar schemes has been used for PID tuning (Killingsworth and Krstić, 2006) and global extremum seeking (Tan et al., 2006). Applications still include combustion processes (Larsson, 2005) but also, *e.g.*, anti braking systems (Drakunov et al., 1995), biochemical reactors and formation flights (Ariyur and Krstić, 2003). The feedback scheme from (Krstić and Wang, 2000) has been shown to allow tracking a reference slope of the nonlinearity (Ariyur and Krstić, 2004), *i.e.*, slope seeking. Most extremum control systems face the problem with balancing control performance and system excitation to find the optimal operating point. In this sense there are similarities to adaptive dual control, *e.g.*, (Filatov and Unbehauen, 2004).

The survey in (Sternby, 1979) identifies four different classes of extremum seeking controllers. These are perturbation methods, switching methods, self-driving systems and model-oriented methods. Perturbation methods seem to be the most common used lately, *e.g.*, (Killingsworth and Krstić, 2006) and (Tan et al., 2006), and are based on adding an excitation signal to the control signal $u(t)$. The effect of this perturbation on the output $y(t)$ can be used to estimate the gradient and then use that information to drive the system towards the optimum. For extremum controllers utilizing the switching method, the system is driven in the same direction until the output does not improve anymore. The direction is then reversed, driving the system in the opposite direction and the output will thus switch around the optimum. One problem with this approach is choosing the initial direction. Switching controllers have been widely used, even though the interest in it seems to have diminished lately. In self-driving systems, the controller drives the system in the same direction until $\dot{y}(t) = 0$ is achieved, and then stay at that stationary point. Also for these systems it can be hard to design a control law to make the system move in the right direction initially, as well as not fulfilling $\dot{y}(t) = 0$ for any other conditions than the optima of $f(\cdot)$. This kind of systems are quite rare in the literature. Finally, the model-oriented methods all have in common that they identify parameters from some model structure that is assumed. They differ at, *e.g.*, what kind of model that is assumed and whether the control signal is adjusted to facilitate identification or not. Recent publications include (Wittenmark and Urquhart, 1995), (Wittenmark and Evans, 2001) and (Egart and Larsson, 2005), which use Hammerstein models, Wiener models and second-order polynomial approximations.

Chapter 3

Communication Model and Control Structure

A network can be modelled in several ways. Since the focus of this thesis is to study error correction from a control perspective, we derive a simple model of the communication process suitable for analysis. Starting from a general network description, we try to identify the important characteristics for the design of a feedback controller. Following a discussion on cost functions, the control structure and the components of the proposed controller are presented.

3.1 Communication Model

In a packet-switched network, the sender communicates with a receiver through numerous nodes connected by links. The sender divides the message into packets and transmits them over the communication link, following a path via the nodes in the network. This setup is depicted in Figure 3.1.

3.1.1 Error Correction

The packets are collected in a block and encoded by the sender before transmission, and the receiver decodes the block to retrieve the application data. We assume that there is a lower level protocol that uses sequence numbers on the packets, hence delivering them in the right order and indicates which packets are lost. Hence only end-to-end properties of communication is considered and the path between sender and receiver is seen as a single link that delays and drops packets. This influence is seen as a loss process p_t acting on the link causing the error correction to be delayed and possibly fail, *i.e.*, not successfully recover all application data in a block. This approach is depicted in Figure 3.2.

A block can be decoded only after sufficient number of packets have arrived, or the last packet has been delayed more than its timeout interval if not enough packets arrived. This means that the error correction process is sampled every

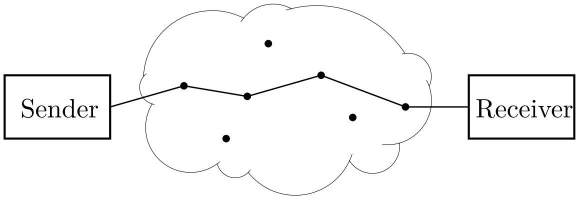

Figure 3.1: The sender communicates with the receiver through a network path, possibly consisting of several nodes. Each packet may take their own path through the network.

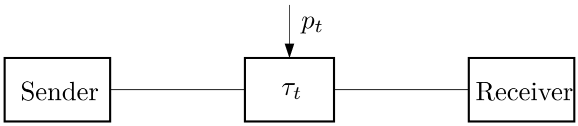

Figure 3.2: The network is modelled as a system that delays and drops packets. This influence is described via the packet loss process p_t.

block only, and the delay of this sample is given by the quotient of the block size and the rate plus the delay of the last received useful packet. Delays and jitter on packet level will thus have little direct influence on the delay of each block.

The coding setup considered is when each block has a fixed number of packets, N. In block t, there are u_t redundancy packets and thus $N - u_t$ packets with application data. The minimum distance is then $u_t + 1$, since it is known not only how many packets are lost but also which ones. A maximum distance separable code can hence correct up to u_t packet losses. In other words, if at least any $N - u_t$ are received, all the original information can be recovered. Since systematic coding is used, the original data are not encoded but sent as is in the $N - u_t$ data packets and information will be transmitted even if more than u_t packets are lost. The amount of information depends on how many of the original application packets that are actually received. Let $X_t^1 \in [0, N - u_t]$ and $X_t^2 \in [0, u_t]$ be stochastic variables denoting the number of received application and redundancy packets, respectively, in block t. Then the number of recovered data packets in block t is

$$y_t = \begin{cases} N - u_t & \text{if } X_t^1 + X_t^2 \geq N - u_t \\ X_t^1 & \text{if } X_t^1 + X_t^2 < N - u_t \end{cases} \tag{3.1}$$

If an insufficient number of application packets are received, the decoding process will not be able to recover all packets. This data loss introduces a distortion on the original message, and that distortion can be measured in several ways. Here, we consider the distortion signal d_t to be simply the difference between the number of

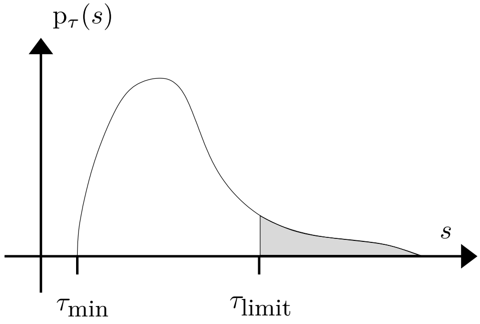

Figure 3.3: A probability density function, $p_\tau(s)$, for the delay τ. If the delay is more than a specified limit, τ_{limit}, the packet is considered lost. The gray area corresponds to the probability for a packet loss.

sent application data packets and the number of recovered data packets, *i.e.*,

$$d_t = N - u_t - y_t$$

The distortion is affected by the packet loss process, p_t, and the amount of redundancy. Note that this is always a positive number since its not possible to recover more data packets than has been sent. This distortion measure describes the amount of lost data after the error correction, and thereby the rate of distortion the application on the receiving side experiences.

3.1.2 Loss Models

For packets that travels through a network, there are several causes for delays and losses. Buffers in forwarding nodes, for example, may delay the packet significantly if the load is high. The packet might even be dropped, either by the buffer being full or by some active queue management scheme (Koo et al., 2004). There are also transmission losses due to, *e.g.*, noisy channels and fading. These kind of losses are mostly present in wireless channels.

For real-time systems, delayed packets can be as bad as lost packets. Thus, lost can be seen as delayed more then a given limit, τ_{limit}. To implement this view of packet losses the sender and the receiver need to be synchronized and the packets marked with timestamps. An illustration of such a model is shown in Figure 3.3. There is a minimum delay, τ_{min}, due to the propagation. From this model, the probability of a packet loss at t is $\Pr(\tau_t > \tau_{limit})$.

The stochastic process describing the probability for such an event is thus very important for the system behavior. To illustrate the differences between loss processes we discuss some basic models for packet loss. The simplest one assumes a fixed probability p for losing a packet, and that the packet losses are independent.

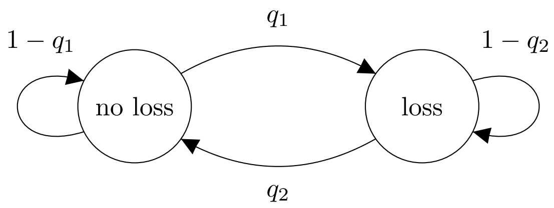

Figure 3.4: The Gilbert model for correlated packet losses.

A bit more realistic model is to have a probability that varies between the blocks but is constant within each block. Depending on the length of the block and the characteristics of the network, it might be reasonable to assume the probability to be independent of the previous block.

Usually, however, network channels are considered to have memory why the independence assumption does no longer hold. A model for correlated losses is the two-state Markov model also known as the Gilbert model (Jiang and Schulzrinne, 2000), shown in Figure 3.4. In this model, q_1 is the probability that the next packet is lost, given that the previous one has arrived, and q_2 is the probability that next packet is received, given that the previous one was lost. The average packet loss probability (the probability for being in state "loss") is $q_1/(q_1 + q_2)$. Even though the Gilbert model is accepted as more realistic than uncorrelated losses there has been some criticism against it, *e.g.*, (Yu et al., 2005). Mainly, it is the probability for longer bursts and the autocorrelation for large time differences that has been lacking. This indicates that Markov models with more states could be useful. In this thesis, however, we use the Gilbert model to illustrate the differences between uncorrelated and correlated packet losses why high accuracy in predicting real loss behavior is not crucial.

3.2 Control Objective

The redundancy can be identified as the control signal, as it has major influence on the error correction. For the output, on the other hand, there are several possibilities, *e.g.*, the number of successfully recovered application packets or the distortion d_t. It is also important to consider the control objectives, what is the desired behavior of this system? The coding is introduced to increase the reliability in the transmission, hence the distortion should be low. This can be achieved by using a large amount of redundancy, but this a major drawback. In our setup, the blocks are of fixed size N which means that increasing the redundancy implies that the number of application data packets must be decreased. A similar argument would

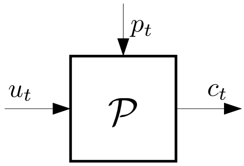

Figure 3.5: The error correction plant. The observed cost c_t can be controlled by the amount of redundancy u_t but is also disturbed by the packet loss process p_t.

apply even if the block size is not fixed, since a too high amount of redundancy would be a waste of network resources and also increase the delay due to the larger block sizes.

To capture this tradeoff we introduce a cost function $J : \mathbb{R}^2 \mapsto \mathbb{R}$ that takes the redundancy u and the distortion d as argument. For small u_t, the distortion d_t is likely to be high and hence the cost is high. For high u_t, the distortion will be low but the cost will be high since u_t is high. As discussed above, the cost function should penalize both high redundancy and high distortion. We make the following standing assumptions:

- J is continuous.
- $J(d, \cdot)$ and $J(\cdot, u)$ are monotonically increasing functions.
- J has a unique global minimum.

If the packet loss process p_t is known, the relation between u_t and d_t will also be known, according to Equations (3.1) and (3.1.1). In such a case, the cost function can be evaluated as a function of the redundancy u_t and packet loss process p_t. The cost can then be minimized analytically or numerically. Thus, it is not only the shape of the cost function J but also the loss process that determines the difficulties on finding the minimizing u_t.

We conclude by presenting the model of the error correction process, which is to be controlled. For each block, an observation of the cost is available as $c_t = J(d_t, u_t)$. The distortion depends on u_t, p_t and the coding scheme. While u_t has a direct influence on the cost c_t, it has little or no influence on the loss process. On the contrary, the loss process p_t influences the output c_t and is hence considered as an external disturbance acting on the system. Thus, the error correction can be seen as a plant, denoted $\mathcal{P}$, with redundancy as the control signal, the packet loss process acting as a disturbance and the observed cost as the output, see Figure 3.5.

The optimal amount of redundancy will be the one that minimizes the expected cost. The objective is then to minimize the expected value of the cost

$$\min_{u_t} \mathbb{E}\{c_t)\} \tag{3.2}$$

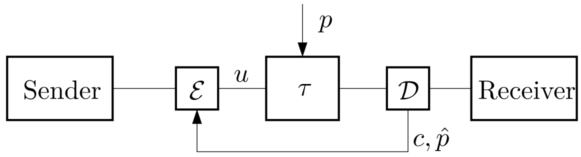

Figure 3.6: The sender encodes the message and transmits it using redundancy u. On the receiver side, the message is decoded, and information on the cost and the estimated packet loss process is sent back to the sender.

with respect to the plant dynamics.

By assuming that the packet loss process is stationary during a sufficiently short time interval, the plant model and the control problem simplifies significantly. When p_t is a stationary process, the expected value of c_t can be seen as a static function of u_t, *i.e.*, $c_t = h(u_t)$. The properties of h will depend on the loss process, but some general observations can still be made. For large enough values of u_t the distortion will be small, but the cost still high due to high value of u_t. When decreasing u_t, the cost will decrease as long as the distortion is not affected. At some point, decreasing u_t will increase the distortion and possibly increasing the cost again. Using $u_t = 0$ will cause a high cost due to extensive distortion.

3.3 Control Structure

The control structure for error correction is discussed next, including presentation of the feedforward and feedback controller.

The cost experienced by the receiver can be communicated back to the sender, in addition the receiver can estimate the loss process p_t and provide the sender also with that information. The sender may then use this information to adapt the redundancy, according to the setup described in Figure 3.6.

Note that the loss process contains feedforward information while the cost is feedback information. The block diagram for this control system is given by Figure 3.7. Hence basing the control signal on the information given by p_t results in a feedforward controller, while basing the control signal on c_t results in a feedback controller. Obviously, the feedforward approach should respond faster but be more sensitive to modelling errors, while the feedback approach is expected to respond slower but be able to adapt to unmodelled changes. Combining this two components, an error correction controller should use the feedforward information for fast transient behavior and the feedback information to get good performance during steady state.

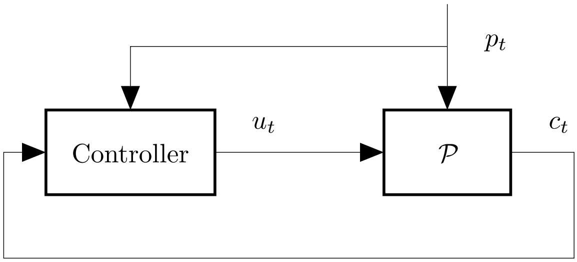

Figure 3.7: A general control structure for error correction with indication of feedforward and feedback information.

3.3.1 Feedforward Control

Feedforward control uses information on the estimated packet loss process to decide on the applied redundancy amount. To decide on the optimal redundancy amount given the packet loss probability, the relation between p_t, u_t and c_t needs to be known. This relation is usually hard to find, why a simplified model of it usually is used. The discrepancies between this model and the current network state might lead to an incorrect amount of applied redundancy, resulting in suboptimal performance. This will not be compensated for, unless there is feedback information on the cost available.

3.3.2 Feedback Control

To address the drawbacks with the feedforward controller, a feedback controller can be utilized. The feedback signal c_t gives information about the cost for the current redundancy level and can be used for estimating $\mathbb{E}\{J(d_t, u_t)\}$. This estimate can then be used to find the amount of redundancy that gives the minimum cost. Since the location of the minimum and also the shape of the cost function are uncertain, a simple strategy is to estimate on which side of the minimum the state is and then adjust u_t so that the state moves closer to the peak. The estimation is based on the difference estimations of c_t and u_t. However, since the underlying signals are very noisy, we need to filter them carefully to obtain a reliable estimate. To this end, we compute c_t^d and u_t^d filtering them through a low-pass filter with a difference term. By comparing the signs of c_t^d and u_t^d it can be concluded on which side of the optimum the current control value corresponds to. The control action will then be to drive the system in the direction of the minimum. Such a controller is given by

$$u_{t+1} = u_t - \beta \operatorname{sgn}(u_t^d c_t^d) \tag{3.3}$$

where $\beta > 0$ is the control parameter. This gives an extremum seeking controller, and it is inspired by the switching controllers in (Sternby, 1979). The control signal

u_t is hence linearly increasing or decreasing with slope β, and it will move towards the minimum until the filtered signals indicate that the it has been passed. The direction is then reversed and the system is driven back towards the minimum.

Chapter 4

Analysis of the Feedback System

The model of the error correction presented in the previous chapter will now be used for analyzing the feedback controller. The focus is on the stationary behavior, with only the feedback component being active. Analysis is performed on block level, and also on a fluid flow model. These results are then illustrated with simulations.

4.1 Convergence Analysis

In an idealized situation, all signals can be known exact. Then, the filtered difference estimations in (3.3) can be replaced with the first order differences, giving the control law

$$u_{t+1} = u_t - \beta \operatorname{sgn}(\Delta u_t \Delta c_t) \tag{4.1}$$

with $\Delta c_t = c_t - c_{t-1}$. Moreover, implementation issues like the quantization is not considered here. Thus the static relation $c = h(u)$ may be viewed as a continuous function.

For this feedback system, we will show convergence to a neighborhood of the optimum and also estimate the size of this region. It will be shown that it is sufficient for h to be a unimodal function to guarantee global convergence of the system, *i.e.*, no convexity is needed. This analysis is done both for a system without delay, as well as for a system with fixed and known delay.

Proposition 4.1.1. *Suppose the relation between the redundancy u_t and the cost c_t is given by $c_t = h(u_t)$, where $h \in \mathcal{C}^1$ is static and unimodal with minimum at u^*. Then u_t converges to a neighborhood of u^* under the control law* (4.1). *In particular, $|u_t - u^*| < 2\beta$, $\forall t > T$ where T is bounded by*

$$T \leq \left\lceil \frac{\min\left(\left|u_0 - (u^* - 2\beta)\right|, \left|u_0 - (u^* + 2\beta)\right|\right)}{\beta} \right\rceil + 2$$

with u_0 being the initial value.

Proof. We first perform a translation such that $u^* = 0$. Moreover, since $h(\cdot)$ is assumed to be unimodal we have that

$$\begin{aligned} h'(u) < 0 \text{ if } u < 0 \\ h'(u) > 0 \text{ if } u > 0 \end{aligned} \tag{4.2}$$

The Δc_t can be expressed as

$$\begin{aligned} \Delta c_t = c_t - c_{t-1} = h(u_t) - h(u_{t-1}) = \\ = (u_t - u_{t-1})h'(\xi) = \Delta u_t h'(\xi), \quad \xi \in [u_{t-1}, u_t] \end{aligned}$$

with help from the mean value theorem. Inserting this in Equation (4.1) gives

$$\begin{aligned} u_{t+1} &= u_t - \beta \operatorname{sgn}(\Delta u_t \Delta c_t) = \\ &= u_t - \beta \operatorname{sgn}(\Delta u_t h'(\xi) \Delta u_t) = \\ &= u_t - \beta \operatorname{sgn}\left((\Delta u_t)^2 h'(\xi)\right) = \\ &= u_t - \beta \operatorname{sgn}(h'(\xi)), \quad \xi \in [u_{t-1}, u_t] \end{aligned}$$

For this system, at a given block $t = t_0$, we consider two different cases. One when u^* is in the interval $[u_{t_0}, u_{t_0-1}]$ and the other when it's not.

We first consider the case where both u_{t_0} and u_{t_0-1} are less than $u^* = 0$. In that case, $h'(\xi) < 0$ and the control update will be $u_{t_0+1} = u_{t_0} + \beta$. Hence the control signal will increase towards the optimum. The case with both u_{t_0} and u_{t_0-1} being greater than u^* is similar.

To see what happens when the control signal reaches the optimum, we consider the other case when u_{t_0} and u_{t_0-1} are on different sides of u^*. If

$$u_{t_0-1} < u^* < u_{t_0} \tag{4.3}$$

then $h'(\xi)$ may have either sign and u_{t_0+1} will move either towards ($u_{t_0+1} = u_{t_0} - \beta$) or from ($u_{t_0+1} = u_{t_0} + \beta$) the optimum. In the first case, $|u_{t_0+1} - u^*| < \beta$ since

$$u^* < u_{t_0} < u^* + \beta \Rightarrow u^* - \beta < u_{t_0+1} < u^*$$

where the first inequality comes from (4.3) together with the control law (4.1) and the implication follows from $u_{t_0+1} = u_{t_0} - \beta$. If $u_{t_0+1} = u_{t_0} + \beta$, on the other hand, we have that

$$u^* < u_{t_0} < u^* + \beta \Rightarrow u^* + \beta < u_{t_0+1} < u^* + 2\beta$$

and $|u_{t_0+1} - u^*| < 2\beta$.

To conclude, we see that we will always move towards the origin if both u_t and u_{t-1} are on the same side of the optimum. This will always be the case if $|u_t - u^*| \geq \beta$. If $|u_t - u^*| < \beta$, on the other hand, u_t and u_{t-1} may be on each side of the optimum. In that case the state may move away to at maximum 2β from the optimum. If it does, we are back in the case when $|u_t - u^*| \geq \beta$ and the state will thus move towards the optimum again.

If the system is not started within this invariant region, the number of blocks it takes to converge to it can be found by considering the system at some initial condition u_0. Since the initial differences may be incorrect, the system may initially move away from the optimum. On the next block, however, the differences will be correct and the system will move towards the optimum. Hence, after at most two blocks the system will be at u_0 and moving towards u^*.

The distance to the invariant region from u_0 is $|u_0 - (u^* - 2\beta)|$ if $u_0 < u^*$ and $|u_0 - (u^* + 2\beta)|$ if $u_0 > u^*$. With a step of β each block, this gives the convergence bound. □

If there is a fixed and known delay τ on the feedback channel, it is still possible to prove convergence. By this we mean that at block t, information on the cost at block $t - \tau$, *i.e.* $c_{t-\tau}$, is available. Note that this can be achieved by using timestamps on the feedback packets. Hence, the control law in this case is

$$u_{t+1} = u_t - \beta \operatorname{sgn}(\Delta u_{t-\tau} \Delta c_{t-\tau}) \tag{4.4}$$

Note that the difference in the control signal is delayed as well. This is needed to get a correct estimation of slope of the cost function, since the cost at $t-\tau$ affected by the redundancy at block $t - \tau$. Because of this delay, however, the region to which the system converges to will be larger and in fact also depend on the delay.

Proposition 4.1.2. *Suppose there is a static relation between the redundancy u_t and cost function c_t is given by $c_t = h(u_t)$, where $h \in \mathcal{C}^1$ is static and unimodal with minimum at u^*. Then u_t converges to a neighborhood of u^* under the control law* (4.4)*. In particular, $|u_t - u^*| < (\tau + 2)\beta$, $\forall t > T$ where T is bounded by*

$$T \leq \left\lceil \frac{\min\left(|u_0 - (u^* + (2+\tau)\beta)|, |u_0 - (u^* - (2+\tau)\beta)|\right)}{\beta} \right\rceil + 2(\tau + 1)$$

with u_0 being the initial value.

Proof. We first preform a translation such that $u^* = 0$. The difference $\Delta c_{t-\tau}$ will then be, using the unimodal property (4.2) as in the proof of Proposition 4.1.1,

$$\Delta c_{t-\tau} = \Delta u_{t-\tau} h'(\xi), \quad \xi \in [u_{t-\tau-1}, u_{t-\tau}]$$

Inserting this in Equation (4.4) gives

$$u_{t+1} = u_t - \beta \operatorname{sgn}(h'(\xi)), \quad \xi \in [u_{t-\tau-1}, u_{t-\tau}]$$

Again, we consider the different cases when u^* is in the interval $[u_{t_0-\tau-1}, u_{t_0-1}]$ and the other when it's not.

First, consider the case where $u_t < u^*, \forall t \in [t_0 - \tau - 1, t_0]$. Then, $h'(\xi) < 0$ and the control update will be $u_{t+1} = u_t + \beta$. This will continue at most until $u_{t-\tau-1} > u^*$. Then, $u_t = u_{t-\tau} + \tau\beta$ since there has been an increase of β for τ steps. u_t is then bound by $u^* + (\tau + 2)\beta$ since

$$u^* < u_{t-\tau-1} < u^* + \beta \Rightarrow u^* + (\tau + 1)\beta < u_t < u^* + (\tau + 2)\beta$$

The case with $u_t, t \in [t_0 - \tau - 1, t_0]$ are all being greater than u^* is similar.

The number of blocks until convergence is found in the same way as the proof of Proposition 4.1.1, with the invariant region now being $(\tau + 2)\beta$. In addition, the initial conditions may drive the system in the wrong direction for $\tau + 1$ samples and it will take another τ to get back to the initial condition. □

4.2 Fluid Flow Analysis

The results in the propositions in Section 4.1 shows that the system converges to a neighborhood of the minimum, hence it is interesting to further study the properties of the solution. This is done using the multiple-input describing function method and indicates that the solution is an oscillation, where the frequency and the amplitude are given by the control parameters.

4.2.1 Closed-Loop Model of the Feedback System

A continuous-time model for the extremum-seeking controller and the plant is derived next. The discussion in Section 3.2 motivated that, during a sufficiently short time interval, the cost can be approximated as a static, nonlinear function of the redundancy:

$$c(t) = h\left(u(t)\right)$$

The continuous version of the control law (3.3) is

$$\dot{u}(t) = -\beta \operatorname{sgn}(u^d(t)c^d(t)) \tag{4.5}$$

where the estimated derivatives are obtained by a linear filter $F(s)$. With these assumptions, we have the model, denoted *Double loop model*, shown in Figure 4.1. In the figure, the notation $x(t) = u^d(t)$ and $w(t) = c^d(t)$ is used.

Figure 4.2 represents a simplified model of Figure 4.1, and is called the *Single loop model*. It is obtained by opening the loop to the right in Figure 4.1, interpreting $w(t)$ as an external input, and introducing $G(s) = -\beta F(s)/s$. In the next section we will show a proposition on the solution of the Simple loop model and relate it to the Double loop model. This is done by multiple-input describing function analysis.

4.2.2 Multiple-Input Describing Function Analysis

In the case when $h(\cdot)$ is quadratic and $x(t) = A\sin(\omega t)$ in the Double loop model of Figure 4.1, then $w(t) \approx B\sin(2\omega t + \varphi)$. The parameters B and φ will depend on ω in addition to h and the filter $F(s)$. Opening the outer loop and letting $w(t) = B\sin(2\omega t + \varphi)$ gives the single loop model with a specified external input. Based on these assumptions, it is possible to analyze the system using the multiple-input describing function method. The results are formalized in the following proposition.

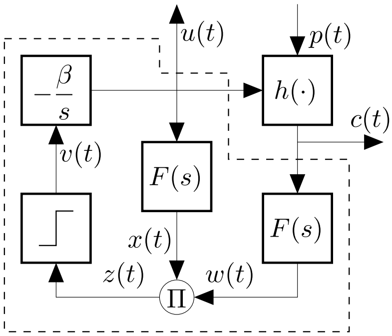

Figure 4.1: Block diagram representing a simplified model of the systems in Figure 3.7. The controller in Figure 3.7 is given by the dashed line and the network by the nonlinear block $h(\cdot)$. This model is referred to as the *Double loop model*.

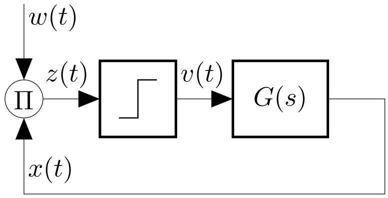

Figure 4.2: Block diagram representing the simplified network model denoted the *Single loop model*. One of the loops of the Double loop model in Figure 4.1 has been cut and is represented by the external input w.

Proposition 4.2.1. *Consider the Single loop model in Figure 4.2 with* $x(t) = A\sin(\omega t)$ *and* $w(t) = B\sin(2\omega t + \pi)$*. If* $G(s) = -\beta F(s)/s$ *is low pass, then multiple-input describing function analysis gives*

$$\frac{4}{A\pi} e^{-i\pi/2} G(i\omega) = 1 \tag{4.6}$$

Proof. The modulated signal $z(t) = x(t)w(t)$ will have two frequency components with the same amplitude, why the input to the relay is

$$z(t) = \bar{A}\sin(\overbrace{\omega t + \varphi + \pi/2}^{\Psi_1}) + \bar{A}\sin(\overbrace{3\omega t + \varphi - \pi/2}^{\Psi_2}) \tag{4.7}$$

with $\bar{A} = AB$. In general a two-sinusoid-input describing function will have two arguments, the amplitude of the two sinusoids (Gelb and Vander Welde, 1968).

In this case, however, the amplitudes are the same why there will only be one argument.

A Fourier series expansion then gives

$$v(t) = \sum_{m=0}^{\infty} \sum_{\substack{n=-\infty (m\neq 0) \\ n=0 (m=0)}}^{\infty} \left[P_{mn} \sin(m\Psi_1 + n\Psi_2) + Q_{mn} \cos(m\Psi_1 + n\Psi_2)\right] \tag{4.8}$$

The coefficients P_{mn} and Q_{mn} are found by the following integrals

$$P_{mn} = \frac{1}{2\pi^2} \iint_{-\pi}^{\pi} \operatorname{sgn}\left(B\sin(\Psi_1) + B\sin(\Psi_2)\right) \sin(m\Psi_1 + n\Psi_2)\ d\Psi_1\, d\Psi_2$$
$$Q_{mn} = \frac{1}{2\pi^2} \iint_{-\pi}^{\pi} \operatorname{sgn}\left(B\sin(\Psi_1) + B\sin(\Psi_2)\right) \cos(m\Psi_1 + n\Psi_2)\ d\Psi_1\, d\Psi_2$$

First we note that, since sgn is odd, $Q_{mn} \equiv 0$. Evaluating P_{mn} gives

$$P_{mn} = \begin{cases} \frac{8}{\pi^2(m^2-n^2)} \sin\left(\frac{\pi}{2}(m-n)\right) & \text{for } m-n \text{ odd.} \\ 0 & \text{for } m-n \text{ even.} \end{cases} \tag{4.9}$$

Replacing Ψ_1 and Ψ_2 from (4.7) the output is

$$\begin{aligned} v(t) &= \sum_{m=0}^{\infty} \sum_{\substack{n=-\infty (m\neq 0) \\ n=0 (m=0)}}^{\infty} P_{mn} \sin(m(\omega t + \varphi + \pi/2) + n(3\omega t + \varphi - \pi/2)) \\ &= \sum_{m=0}^{\infty} \sum_{\substack{n=-\infty (m\neq 0) \\ n=0 (m=0)}}^{\infty} P_{mn} \sin((m+3n)\omega t + (m+n)\varphi + (m-n)\pi/2) \end{aligned} \tag{4.10}$$

Since $G(s)$ is low-pass, $v(t)$ will mainly consist of

Therefore, the output components of frequency ω is the only one of interest. This corresponds to the terms of the sum (4.10) where $m + 3n = \pm 1$, for which $m + n$ is odd and thereby $P_{mn} \neq 0$ according to (4.9). The output with frequency ω is then

$$\begin{aligned} v(t) = &\sum_{n=-\infty}^{0} \frac{8}{\pi^2(8n^2 - 6n + 1)} \sin(\omega t + \varphi + \pi/2 - 2n\varphi) \\ &+ \sum_{n=-\infty}^{-1} \frac{8}{\pi^2(8n^2 + 6n + 1)} \sin(\omega t + \varphi + \pi/2 + 2n\varphi) \end{aligned}$$

For $\varphi = \pi$ the sums evaluates to

$$\begin{aligned} v(t) &= \frac{8}{\pi^2}\left(\sum_{k=0}^{\infty}\frac{1}{8k^2+6k+1}+\sum_{k=1}^{\infty}\frac{1}{8k^2-6k+1}\right)\sin(\omega t+3\pi/2) \\ &= \left(\frac{2\pi+4\ln 2}{\pi^2}+\frac{2\pi-4\ln 2}{\pi^2}\right)\sin(\omega t-\pi/2) \\ &= \frac{4}{\pi}\sin(\omega t-\pi/2) \end{aligned}$$

The signal $x(t) = A\sin(\omega t)$ in Figure 4.2 becomes $v(t) = \frac{4}{\pi}\sin(\omega t - \pi/2)$, and thus the describing function in this case is

$$N_\pi(A) = \frac{4}{A\pi}e^{-i\frac{\pi}{2}}$$

This gives the equation in the proposition. □

4.3 Numerical Examples

The results in Propositions 4.1.1, 4.1.2 and 4.2.1 will now be illustrated with a few examples.

4.3.1 Illustration of Convergence

The results showing convergence to a symmetric invariant set around the minimum will first be illustrated. This is done by simulating the systems in Equations (4.1) and (4.4) for different values of the controller gain β and the delay τ. Throughout these simulations, the cost function $c_t = u_t^2$ were used giving the minimum as $u^* = 0$.

The first simulation is shown in Figure 4.3, where the controller gain β was set to 0.1. The delay τ was set to zero, and from Proposition 4.1.1 the invariant region for such a system is given by $u^* \pm 0.2$, which is marked by the dotted black lines in the figure. The control signal u_t is the solid black line, which stays within the invariant region ones it has been entered. This region is reached within at $t = 7.5$, which is below the bound.

The grey lines in Figure 4.3 shows a simulation for a similar case, but now there is a delay of $\tau = 2$. This gives the invariant region as $u^* \pm 0.4$ shown as the dotted grey lines in the figure. Also here, the control signal stays within this region after reaching it. The boundary at 0.4 is reached at $t = 6.5$, which is less than the bound 7 for $u_0 = 0.65$.

A similar experiment with $\beta = 0.25$ was also performed. The results from this experiment is shown in Figure 4.4. With delay $\tau = 0$, black lines, the interval $[-0.5, 0.5]$ is an invariant set for the control signal. Starting at the initial value $u_0 = 1.9$, this set is reached at $t = 7.6$ The bound in this case is 8. Using the

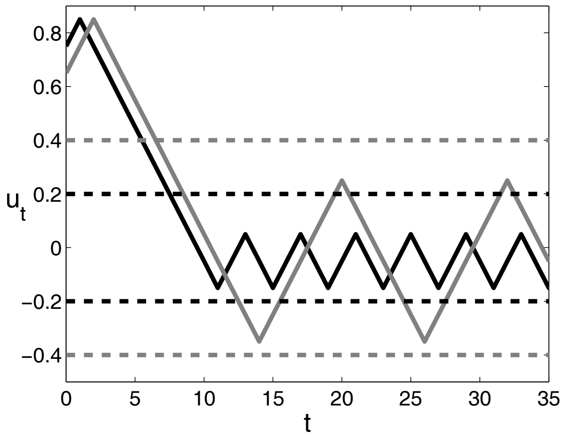

Figure 4.3: Simulation of the systems (4.1) and (4.4). The solid lines are u_t, while dashed lines are the bounds $\pm(2+\tau)\beta$. Black lines show the case when $\tau = 0$, grey lines $\tau = 2$. The controller gain, β, is 0.1 in both cases.

same β but adding a delay of $\tau = 3$, the results still hold. With the control signal starting at $u_0 = 2.1$ and in the right direction, the boundary at -1.25 is reached at $t = 3.4$, clearly less than the bound 12.

It is also interesting to note the sawtooth shaped trajectory of the control signal; this limit cycle behavior was a motivation of the describing function analysis. The oscillation is in general not symmetric around the optimum, which is natural since the control signal only can change in steps of β. Thus, this offset depends only on the initial conditions in this case. A nonsymmetric nonlinearity h may also cause the oscillation to not be symmetric around u^*.

4.3.2 Illustration of Fluid Flow Models

The describing function conditions from Proposition 4.2.1 will be illustrated with simulations of the Single loop model. The results are also compared with simulations of the Double loop model.

The filter is implemented in discrete form with the structure

$$G_d(z) = -\beta \frac{(1-a)(z+1)}{(z-a)^3} \tag{4.11}$$

For parameter values $\beta = 0.1$ and $a = e^{-0.2}$ the continuous approximation of it is

$$G(s) = \frac{-0.01205s^2 + 0.04821s - 0.04821}{s^3 + 0.598s^2 + 0.1192s + 0.007921} \tag{4.12}$$

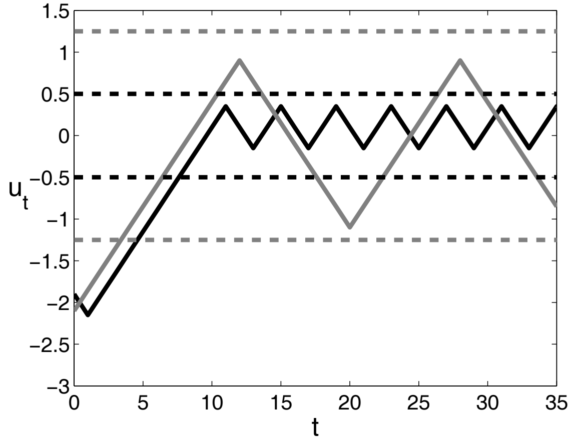

Figure 4.4: Simulation of the systems (4.1) and (4.4). The solid lines are u_t, while dashed lines are the bounds $\pm(2+\tau)\beta$. Black lines show the case when $\tau = 0$, grey lines $\tau = 3$. The controller gain, β, is 0.25 in both cases.

Proposition 4.2.1 proposes an oscillation of the form $x(t) = A\sin(\omega t)$ with the amplitude $A = 5.4$ and frequency 0.017 Hz. To investigate whether this oscillation is expected to be stable or not, the Nyquist plot of $G_1(i\omega)$ in Figure 4.5 is studied. The oscillation occurs at the intersection between $G(i\omega)$ and $\frac{1}{N_\pi(A)} = -\frac{A\pi}{4}i$ in the complex plane. An oscillation with amplitude $A_1 < 5.4$ will correspond to a point on the positive imaginary axis below the intersection, encircled by the Nyquist curve. Thus, replacing the nonlinearity with a constant gain A_1 will result in an unstable system according to the Nyquist criterion, causing the amplitude to increase. Oscillation with an amplitude $A_2 > 5.4$, on the other hand, corresponds to a point on the positive imaginary axis above the intersection, outside the Nyquist curve. A system with gain A_2 will thus be stable, and the amplitude is expected to decrease. This result suggests that the oscillation is stable and that the solution will converge to it if the system is initialized close enough.

Simulation of the Single loop model with the parameters as in (4.12) gives the result shown in Figure 4.6. The signal $x(t)$ oscillates with amplitude slightly higher than 5, and from the frequency spectrum the oscillation is found to have a frequency of 0.017 Hz. This agrees very well with the predicted values, why the describing function approach seems to hold. Moreover, the solution converges to this oscillation as suggested by the analysis of the Nyquist curve.

Now, it is also interesting to see how well this simplified model describes the more detailed Double loop model. By simulating that system, splitting $G_1(s)$ into $-\frac{\beta}{s}F(s)$ with $\beta = 0.1$, the results in Figure 4.7 are obtained. The square function

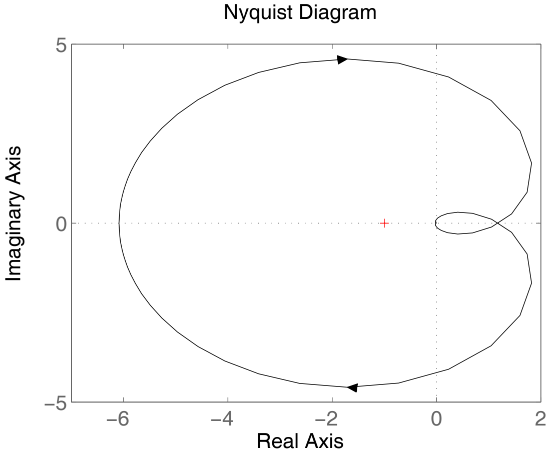

Figure 4.5: The Nyquist plot of $G_1(i\omega)$ indicates that the oscillation is stable. $1/N_\pi(A)$ is an increasing function along the positive imaginary axis, that intersects the Nyquist curve at $A = 5.4$ and $\omega = 0.11$.

was used as the nonlinearity $h(\cdot)$ in the model (see Figure 4.1). Comparing with the Single loop model, the oscillation in $x(t)$ has slightly higher frequency, 0.022 Hz, and an amplitude somewhat above 3, which is lower than for the Single loop model. The oscillation is also stable, just as predicted by analysis and the Single loop model.

The assumption that $w(t)$ has one main frequency component with twice the frequency of $x(t)$ holds, which is clearly seen in the right plot in Figure 4.7. There are small components of other frequencies, *i.e.*, the third harmonic of $x(t)$ at 0.066 Hz and the second harmonic of $w(t)$ at 0.088 Hz. The energy in these components, however, is about 1/50 of the energy in the first harmonic of the respective signal.

This experiment was repeated for various values of the parameters β and a. The results from these experiments are summarized in Table 4.3.2. The single loop model, which the describing function calculations were based on, clearly agrees better with the values predicted by the describing function method. The frequency is following the calculated values very well, while the amplitude is slightly lower than the calculated value. The Double loop model, on the other hand, produces an oscillation with slightly higher frequency and lower amplitude. It is worth noting, though, that when changing the filter G to obtain other characteristics of the oscillation the effect on both models is similar. When using twice as high gain, *i.e.*, filter G_2, the amplitude is predicted to be doubled, which also is the case for both models. The same relation holds when changing the filter dynamics, *i.e.*, filter G_3, where the frequency is increased by a factor just below 2.2 in all cases. These

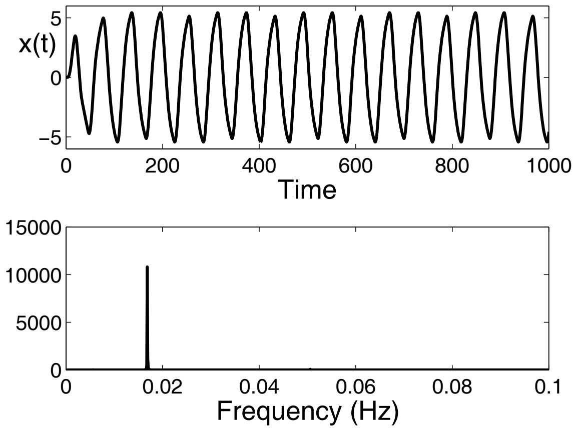

Figure 4.6: A simulation of the Single loop model (Figure 4.2), where the oscillation fortifies our describing function assumption. The upper plot shows $x(t)$, while its frequency content is plotted below.

results indicate that even though the results in Proposition 4.2.1 may not predict exact properties of the Double loop system, it gives values that agrees reasonably well. More important, it seems to be able to predict the effects of changing the parameters in the model.

This analysis has showed the connection between the Single loop model and the Double loop model. The motivation for this is the investigation presented in Section 5.3.2, comparing the Double loop model with data from an experiment on wireless network link.

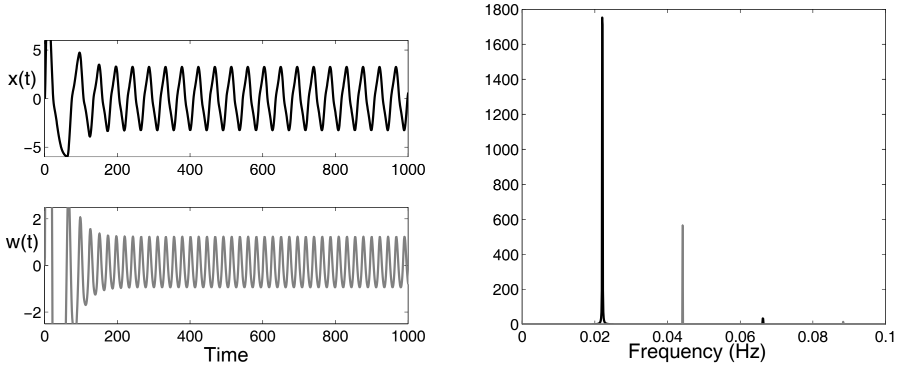

Figure 4.7: A simulation of the Double loop model (Figure 4.1) the signals $x(t)$, black, and $w(t)$, gray, are shown to the left. To the right, their frequency content. The oscillations in $x(t)$ and $w(t)$ indicate that the Single loop model captures the main dynamic behavior. The frequency is slightly higher than in the simplified model, but the assumptions with one frequency component in $x(t)$ and one in $w(t)$ with twice the frequency seem to hold.

$\begin{bmatrix}\beta & a\end{bmatrix}$		$\begin{bmatrix}0.1 & e^{-0.2}\end{bmatrix}$	$\begin{bmatrix}0.2 & e^{-0.2}\end{bmatrix}$	$\begin{bmatrix}0.1 & e^{-0.5}\end{bmatrix}$
Proposition 4.2.1	Amp.	5.4	10.7	1.2
	Freq.	0.017	0.017	0.037
Single loop model	Amp.	5.1	10.1	1.2
	Freq.	0.017	0.017	0.037
Double loop model	Amp.	3.3	6.5	0.86
	Freq.	0.022	0.022	0.048

Table 4.1: The Single loop model shows very good agreement with the values predicted by Proposition 4.2.1, while the Double loop model seems to overestimate the frequency and underestimate the amplitude.

Chapter 5

Experiments

In this chapter, the error correction scheme is evaluated with simulations of a packet-switched network and experiments on a sensor network. The previous chapter presented encouraging results on the properties of the feedback algorithm on a simple model of the error correction. Thus it is interesting to see how well these results translates to conditions that are more realistic. First, the choice of feedforward algorithm and cost function is presented.

5.1 Implementation

The implemented controller uses the feedback algorithm during steady state, but also utilizes a change detection filter based on feedforward information on the packet loss probability. This combination is chosen since the feedback is assumed to handle slower variations but lacks in the transients, as discussed in Section 3.3.2. The feedforward part is thus added only to improve the transient behavior.

For this implementation, the change detection filter is chosen as a sequential probability ratio test that works in the following way (Gustafsson, 2001). At each time step (*i.e.*, for each packet) a residual is calculated as $\varepsilon_t = \bar{p}_t - \hat{p}_{t-1}$, where $\bar{p}_t$ is the observed packet loss probability over a short window w_1, and $\hat{p}_{t-1}$ is the packet loss probability over the longer window w_2. The filter state g_t is then updated according to

$$g_t = g_{t-1} + \varepsilon_t - \nu$$

After each update, there is a check for alarm.

$$\begin{cases} \text{Alarm, change time } T_{change},\ \text{and } g_t, w_1, w_2 = 0 \\ \qquad\qquad \bar{p}, \hat{p} \text{ estimated over } [t - T_{change}, t] & \text{if } g_t > h \\ g_t = 0, T_{change} = t & \text{if } g_t < \alpha \end{cases}$$

The parameter ν is used to compensate for slow drifts. If g_t exceeds the threshold h, there will be an alarm and a new packet loss probability is estimated. The internal state of the estimator is also reset, including the windows w_1 and w_2. To avoid negative drift, which could increase the time to detect a change, the state g_t is also reset when it reaches a negative threshold, α. Since this corresponds to a one-sided test, two tests are run in parallel to detect both sudden increases and decreases in packet loss probability.

If a change in packet loss probability is detected, the redundancy is reset to what is optimal given the new estimate. The optimality depends on the loss model, and here uncorrelated losses are assumed when calculating the new redundancy value. The extremum seeking algorithm (3.3) then continues seeking for the optimal steady state value. By combining these ideas it is possible to achieve both fast responses during transients and converging to the optimal value during steady state. The control algorithm will then be

$$u_{t+1} = \begin{cases} u^*(\hat{p}_t) & \text{if change detected} \\ u_t - \beta \operatorname{sgn}(u_t^d c_t^d) & \text{otherwise} \end{cases} \tag{5.1}$$

where $u^*(p_t)$ is the optimal amount of redundancy for packet loss probability $\hat{p}_t$.

Evaluating this controller on a real or simulated network, a cost function is introduced following the discussion in Section 3.2. Based on that increasing u decreases the throughput linearly, and that a large distortion d is highly undesirable, the following cost function will be studied

$$J(d,u) = \left(\frac{d}{N}\right)^2 + \rho\frac{u}{N} \tag{5.2}$$

where ρ is a positive constant. By assuming a fixed packet loss probability it is possible to calculate the cost function, which is shown in Figure 5.1 for some values of p_t.

For this particular cost function, $u^*(\hat{p}_t)$ can be found in the following way. First, the expected value of the cost function (5.2) at block t is evaluated.

$$\begin{aligned} \mathbb{E}\{J\} &= \mathbb{E}\left\{\left(\frac{d_t}{N}\right)^2 + \rho\frac{u_t}{N}\right\} = \mathbb{E}\left\{\left(\frac{d_t}{N}\right)^2\right\} + \rho\frac{u_t}{N} \\ &= \mathbb{E}\left\{\left(\frac{(N-u_t)-y_t}{N}\right)^2\right\} + \rho\frac{u_t}{N} \\ &= \left(\frac{N-u_t}{N}\right)^2 - 2\left(\frac{N-u_t}{N^2}\right)\mathbb{E}\{y_t\} + \frac{\mathbb{E}\left\{y_t^2\right\}}{N^2} + \rho\frac{u_t}{N} \end{aligned} \tag{5.3}$$

By assuming a suitable loss process, $\mathbb{E}\{y\}$ and $\mathbb{E}\{y^2\}$ can be found using (3.1). If the losses are independent, X_t^1 and X_t^2 will both be binomial random variables and (5.3) is easily minimized with respect to u_t. This optimization problem can be solved off-line, and then the control is a simple table look-up based on the estimated packet loss probability $\hat{p}_t$.

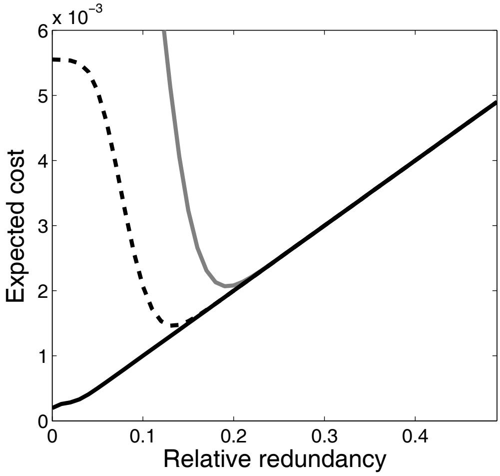

Figure 5.1: The cost function (5.2) in the given setup for three different values of the packet loss probability. The grey line corresponds to $p = 11\%$, dotted black is for 7% and solid black for $p = 1\%$.

5.2 Simulation

This section will present a simulation to illustrate the ability of the feedback algorithm to find the optimum without the need for a model. This will be done by comparing a feedforward controller and one that utilizes feedback in combination with the change detection filter described in Section 5.1. The feedforward controller assumes the losses are independent, and may thus find the minimizing u_t from the estimated packet loss probability $\hat{p}_t$. The strategy for finding $\hat{p}_t$ is described in Section 5.1. By performing simulations of these two controller running over a network, the packet loss process can be controlled to study the effects of model errors. In this experiment the Gilbert model described in Section 3.1.2 is used for generating the loss process.

The results from this experiment are presented in Figure 5.2. To the left, the applied redundancy for the two controller are shown together with the optimal redundancy. After half the experiment, the parameters in the Gilbert model are modified to reflect a change in the network conditions for the worse. With a higher probability of packet loss it is beneficial to increase the redundancy, as both controllers do.

There are several interesting observations that can be made from this figure. First, the feedforward controller (dark grey) applies too low amount of redundancy during the whole simulation, and thus causes the cost to be higher. The reason that it applies to low redundancy is of course the faulty assumption of independent

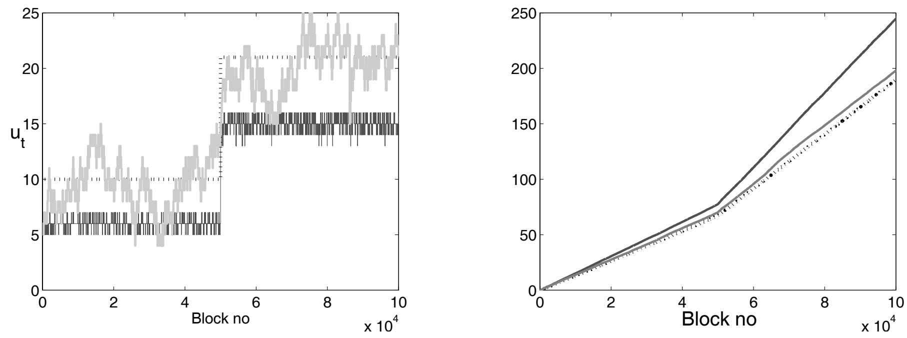

Figure 5.2: The control signal u_t, the number of redundancy packets per block, is shown to the left, and the cumulative cost is shown to the right, for a simulation with packet losses generated by the Gilbert model. In both figures, the dotted black line is the optimal control, the dark grey is the feedforward and the light gray is the feedback controller with the change detection filter.

losses. If there is a correlation on the loss process, it is more likely that losses comes in bursts. This increases the probability of loosing many packets in the same block, causing the probability of successful decoding of a block to decrease. Consequently, the distortion will increase and the cost will then be higher. It is then beneficial to increase the redundancy amount to compensate for this. The feedback controller, instead, uses information on the cost and can thus compensate for this by using more redundancy. In the plot to the right this effect can be seen as the feedforward controller has the highest increase of the cumulative cost.

5.3 Sensor Mote Experiments

The feedback controller was also implemented on sensor motes and run over a wireless link. This test bed will be described in more detail, and then the results from the experiment is presented.

5.3.1 Test Bed

The test bed includes applications implemented in TinyOS (TinyOS Alliance) running on Telos motes (Polastre et al., 2005) and a server (or gateway) software for bridging between the individual sensor network and the IP network. The Telos motes read data from their onboard sensors and send the readings to the server using the standard TinyOS multi-hop communication protocol. The server is in charge of the implementation of the control algorithm, since the packet loss rate from the remote nodes up to the node base station can be very high (even more

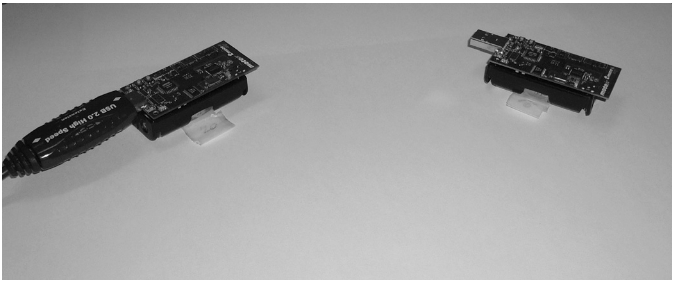

Figure 5.3: The error correction mechanism is illustrated using two wireless sensor motes.

then 10%), while the packet error rate from the server to the remote application may be small. Details on the implementation are found in (Björkestad, 2007). The setup is pictured in Figure 5.3.

5.3.2 Results

In this section, the ability of the Double loop model to describe the error correction control over a wireless link is investigated through an experiment. The feedback controller is run on the test bed for several control parameter settings, and the Double loop model is simulated with the same values. The cost function used in the simulation was obtained by taking the average value of the experienced cost for redundancy level.

The results from this experiment is presented in Table 5.3.2, where the control parameters β and a follows the notation in Section 4.3.2. The signal $x(t)$ from the experiment is shown in Figure 5.4 for two different values of control parameters. The signals are oscillatory though not sinusoids with one single frequency. From the frequency plot (to the right) it is seen that most of the energy in the signal is concentrated in a rather small frequency band. One reason for this is the "noise" in the measurement of the cost. Even though the average value of the cost is fairly constant over a large number of blocks, the observed cost at a specific block may differ significantly from that.

The values obtained in the experiment deviates by nearly a factor two from the ones achieved when simulating the Double loop model. On the other hand, the effects on the amplitude and frequency of the oscillation when changing a control parameter is very similar. When the gain β is doubled, the amplitude is also approximately doubled and the frequency only changes marginally. Changing the poles and thereby the cut off frequency of the filter affects mainly the frequency, which both for the Double loop model and for the experiment is approximately halved.

$[\beta\ a]$	Double loop		Test bed	
	Amp.	Freq.	Amp.	Freq. $[\mu\ \sigma]$
$[0.1\ e^{-0.2}]$	3.1	0.023	≈ 5	[0.012 0.0089]
$[0.2\ e^{-0.2}]$	6.6	0.022	≈ 10	[0.013 0.0072]
$[0.1\ e^{-0.1}]$	11	0.012	≈ 15	[0.0069 0.0038]
$[0.2\ e^{-0.1}]$	22	0.011	≈ 30	[0.0084 0.0039]

Table 5.1: The table shows the amplitude and frequency for the signal $x(t)$, *i.e.*, the estimated derivative of the control signal (see Figure 4.1). The experiment is performed on a real network, and compared with values from simulation of the Double loop model. Even though the values differ by up to almost a factor of two, the order of magnitude as well as the effects of changing the control parameters agrees well between the model and the true system. The frequency spectrum of the signal from the experiment was fitted to a normal distribution with mean μ and standard deviation σ.

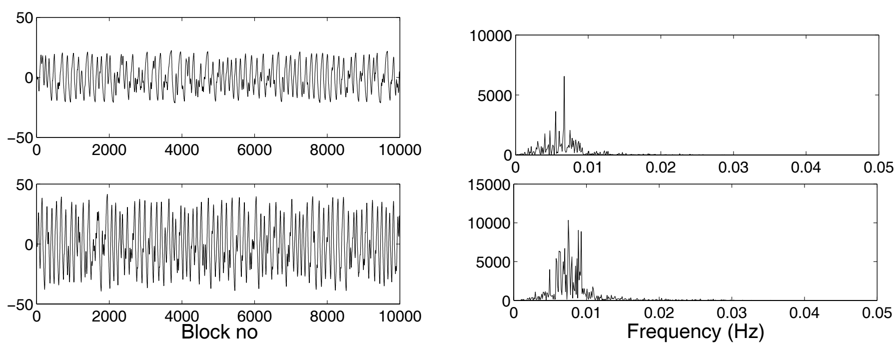

Figure 5.4: The signal $x(t)$, left, and its frequency component, right, during the experiment on the test bed. The control parameter were $a = e^{-0.1}$ in both cases and $\beta = 0.1$ (top) and $\beta = 0.2$ (bottom).

Chapter 6

Conclusions

The thesis is concluded with a brief description of the main results. It is followed by a presentation of a few interesting problems that warrants future research efforts within this area.

6.1 Summary

The main contribution of this thesis is the analysis of the dynamical properties of a simple feedback error correction controller. It has been obtained by formulation error correction as an extremum seeking control problem. The controller has been proven to converge to a neighborhood of the optimum, and describe the size of this region in terms of control and model parameters. A fluid flow model has been developed and multiple-input describing function analysis predicts oscillatory behavior.

The results obtained has also been illustrated and evaluated using simulations as well as experiments on a wireless sensor network. This comparison indicates that the results are relevant also for real networks, with presence of network characteristics such as loss bursts in addition to implementation aspects such as discretization and quantization.

6.2 Future Work

It would be interesting to further investigate the multiple-input describing function approach and find the solution for a general phase difference φ.

Another issue to study is the relation between the two presented closed loop models and the true error correction system run over a network. This would also suggest possible improvements of the models.

A great challenge would be to find an error correction algorithm that is implemented and used today, and analyze in it this framework. This will bring several new interesting issues to this problem, *e.g.*, such distortion measures are often considerably more complicated than the one used here.

Bibliography

K. B. Ariyur and M. Krstić. *Real-Time Optimization by Extremum-Seeking Control*. John Wiley & Sons, 2003.

K. B. Ariyur and M. Krstić. Slope seeking: a generalization of extremum seeking. *International Journal of Adaptive Control and Signal Processing*, 18(1):1–22, feb 2004.

L. Baldantoni, H. Lundqvist, and G. Karlsson. Adaptive end-to-end FEC for improving TCP performance over wireless links. In *Proceedings IEEE International Conference on Communications*, June 2004.

C. Barakat and A. Al Fawal. Analysis of link-level hybrid fec/arq-sr for wireless links and long-lived tcp traffic. *Performance Evaluation Journal*, 57(4):43–50, 2004.

P.-E. Björkestad. Reliable data delivery in wireless sensor networks. Master's thesis, Royal Institute of Technology, February 2007.

P. F. Blackman. Extremum-seeking regulators. In J. H. Westcott, editor, *An Exposition of Adaptive Control*. The Macmillan Company, New York, NY, 1962.

R. E. Blahut. *Theory and Practice of Error Control Codes*. Addison-Wesley, 1983.

J.-C. Bolot, H. Crepin, and A. V. Garcia. Analysis of audio packet loss in the internet. In *Network and Operating System Support for Digital Audio and Video*, pages 154–165, 1995.

J.-C. Bolot, S. Fosse-Parisis, and D. F. Towsley. Adaptive FEC-based error control for internet telephony. In *INFOCOM (3)*, pages 1453–1460, 1999.

M. Busse, T. Haenselmann, and W. Effelsberg. The impact oc forward error correction on wireless sensor network performance. Technical report, Technical Report TR-2006-005, Praktische Informatik IV, Universität Mannheim, 2006.

J. Chakareski and P. A. Chou. Application layer error-correction coding for rate-distortion optimized streaming to wireless clients. *IEEE Transactions on Communications*, 52(10):1675–1687, 2004.

T. J. Cover and J. A. Thomas. *Elements of Information Theory.* John Wiley & Sons, 1991.

S. Drakunov, U. Ozguner, P. Dix, and B. Ashrafi. Abs control using optimum search via sliding modes. *IEEE Transactions on Control Systems Technology*, 3 (1):79–85, 1995.

C. S. Draper and Y. J. Li. Principles of optimalizing control systems and an application to an internal combustion engine. *ASME Publications*, September 1951.

Q. Du and X. Zhang. Adaptive low-complexity erasure-correcting code-based protocols for qos-driven mobile multicast services over wireless networks. *IEEE Transactions on Vehicular Technology*, 55(5):1633–1647, September 2006.

B. Egart and S. Larsson. On a parameter adaptive extremum controller. In *44th IEEE Conference on Decision and Control and European Control Conference*, pages 4809–4814, December 2005.

D.C. Feldmeier, A.J. McAuley, J.M. Smith, D. Bakin, W.S. Marcus, and T. Raleigh. Protocol boosters. *IEEE JSAC, Special Issue on Protocol Architectures for 21st Century*, 16(3):437–44, 1998.

N. Filatov and H. Unbehauen. *Adaptive Dual Control.* Lecture Notes in Control and Information Sciences. Springer Verlag, 2004.

O. Flärdh, K. H. Johansson, and M. Johansson. A new feedback control mechanism for error correction in packet-switched networks. In *44th IEEE Conference on Decision and Control and European Control Conference*, December 2005.

A. Gelb and W. E. Vander Welde. *Multiple-Input Describing Functions and Nonlinear System Design.* McGraw-Hill Electronic Sciences Series, 1968.

V. Gupta, A. F. Dana, J. P. Hespanha, R. M. Murray, and B. Hassibi. Data transmission over networks for estimation and control. Submitted to publication, December 2006.

F. Gustafsson. *Adaptive Filtering and Change Detection.* John Wiley & Sons, 2001.

R.W. Hamming. Error detecting and error correcting codes. *Bell System Technical Journal*, 29:147–160, 1950.

W. Jiang and H. Schulzrinne. Modeling of packet loss and delay and their effect on real-time multimedia service quality. In *Proc. NOSSDAV*, 2000.

N. J. Killingsworth and M. Krstić. Pid tuning using extremum seeking: Online, model-free performance optimization. *IEEE Control Systems Magazine*, 26(1): 70–79, 2006.

J. Koo, S.g-J. Ahn, and J.-W. Chung. Performance analysis of active queue management schemes for ip network. In *International Conference on Computational Science*, volume 3036 of *Lecture Notes in Computer Science*, pages 349–356. Springer, 2004.

M. Krstić and H. H. Wang. Stability for extremum seeking feedback for general nonlinear dynamic systems. *Automatica*, 36(4):595–601, April 2000.

S. Larsson. *Torque Based Combustion Phasing Control.* PhD thesis, Department of Signals and Systems, Automatic Control, Chalmers University of Technology, Dec 2005.

M. Leblanc. Sur l'electrification des chemins de fer au moyen de courants alternatifs de frequence elevee. Revue Generale de l'Electricite, (Frances), 1922.

Q. Ling and M.D. Lemmon. Power spectral analysis of networked control systems with data dropouts. *IEEE Transactions on Automatic Control*, 49(6):955–959, 2004.

H. Lundqvist and G. Karlsson. TCP with end-to-end forward error correction. In *Proceedings of International Zurich Seminar on Communications*, February 2004.

H. Lundqvist and G. Karlsson. On the optimization of local and end-to-end forward error correction. In *In Proc. of European Wireless 2005*, April 2005.

F. J. MacWilliams and N. J. A. Sloane. *The Theory of Error-Correcting Codes.* North-Holland, 2003.

I. S. Morosanov. Method of extremum control. *Automatic & Remote Control*, 18: 1077–1092, 1957.

I. I. Ostrivskii. Extremum regulation. *Automatic & Remote Control*, 18:900–907, 1957.

K. Park and W. Wang. AFEC: An adaptive forward error correction protocol for end-to-end transport of real-time traffic. In *International Conference on Computer, Communications and Networks*, pages 196–207, 1998.

J. Polastre, R. Szewczyk, and D. Culler. Telos: Enabling ultra-low power wireless research. *IPSN/SPOTS*, April 2005.

L. Roychoudhuri and E. Al-Shaer. Real time analysis of delay variation for packet loss prediction. In *7th International Conference on Management of Multimedia Networks and Services*, October 2004.

L. Schenato, B. Sinopoli, M. Franceschetti, K. Poolla, and S. Sastry. Foundations of control and estimation over lossy networks. *Proceeding of the IEEE, Special Issue on Networked Control Systems*, 2006. To appear.

N. Shacham and P. McKenny. Packet recovery in high-speed networks using coding and buffer management. In *IEEE INFOCOM*, 1990.

C. E. Shannon. A mathematical theory of communication. *Bell System Technical Journal*, 27:379–423 and 623–656, Jul and Oct 1948.

B. Sinopoli, L. Schenato, M. Franceschetti, K. Poolla, M. Jordan, and S. Sastry. Kalman filtering with intermittent observations. *IEEE Transactions on Automatic Control*, 49(9):1453–1464, September 2004.

J. Sternby. A review of extremum control. Technical report, Department of Automatic Control, Lund Institute of Technology, April 1979.

Y. Tan, D. Nesic, I.M.Y. Mareels, and A. Astolfi. Global extremum seeking in the presence of local extrema. *Automatica*, 46:889–903, 2006.

S. Tatikonda and S Mitter. Control under communication constraints. *IEEE Trans. Automat. Control*, 49:1056–1068, 2004.

TinyOS Alliance, 2006. URL `http://www.tinyos.net`.

J. F. Waldby. *Adaptive Protocols for Reliable Internet Data Transport*. PhD thesis, Department of Electrical Engineering, University of Illinois Urbana-Campaign, 2001.

A. Willig, K. Matheus, and A. Wolisz. Wireless Technology in Industrial Networks. *Proceedings of the IEEE*, 93(6):1130–1151, June 2005.

B. Wittenmark and R. J. Evans. Extremal control of wiener model processes. Technical report, Department of Automatic Control, Lund Institute of Technology, Dec 2001.

B. Wittenmark and A. Urquhart. Adaptive extremal control. In *IEEE Conference on Decision and Control*, pages 1639–1644, 1995.

W. S. Wong and R. W. Brockett. Systems with finite communication bandwidth constraints. I. State estimation problems. *IEEE Trans. Automat. Control*, 42(9): 1294–1299, 1997.

W. S. Wong and R. W. Brockett. Systems with finite communication bandwidth constraints. II. Stabilization with limited information feedback. *IEEE Trans. Automat. Control*, 44(5):1049–1053, 1999.

X. Yu, J. W. Modestino, and X. Tian. The accuracy of gilbert models in predicting packet-loss statistics for a single-multiplexer network model. In *IEEE INFOCOM*, pages 2602–2612, 2005.