



HELMUT SCHMIDT  
UNIVERSITÄT

---

Universität der Bundeswehr Hamburg

# Implementation and Evaluation of Audio Based Methods for Robust Inter-Robot Communication

*Project Thesis*

by

**Finn Poppinga**

Start date:	April 15, 2016
End date:	July 29, 2016
Supervisor:	Lars Urbansky, M.Sc.
Supervising Professor:	Prof. Dr.-Ing. habil. Udo Zölzer



## Abstract

*RoboCup* is a competition that aims to implement robot soccer under realistic conditions. Currently robots are allowed to communicate via WiFi. To test the capability of cost-effective robots to communicate via an aerial acoustic channel, we implement chirp-signal based acoustic communication on the NAO robotic platform in a way similar to [1]. It is demonstrated that it is possible to transmit data over application typical distances of more than 6.5 m and with a data-rate of more than 20 Bytes<sup>-1</sup>. The transmission scheme is then extended to use 2 bit per symbol and enhance the data-rate even more. The communication scheme developed for this thesis was used in the *RoboCup 2016 No-WiFi Challenge* and ranked first, transmitting more than 50 time the amount of data as the runner-up implementation.



*To the HULKS, greatest RoboCup team in the world.*



# Statement

Hereby I do state that this work has been prepared by myself and with the help which is referred within this thesis.

Hamburg, July 27, 2016





# Foreword

Hamburg, July 27, 2016



# Contents

<b>List of Figures</b>	<b>xiii</b>
<b>List of Tables</b>	<b>xv</b>
<b>List of Abbreviations</b>	<b>xvii</b>
<b>1 Introduction</b>	<b>1</b>
1.1 Motivation . . . . .	1
1.2 Requirements . . . . .	2
1.3 The NAO Robotics Platform . . . . .	2
1.3.1 Audio Hardware . . . . .	3
<b>2 Chirp-Based Communication</b>	<b>5</b>
2.1 Working Principle . . . . .	5
2.1.1 Properties of Chirp Signals . . . . .	5
2.2 Transmitter Design . . . . .	6
2.2.1 Assumptions . . . . .	6
2.2.2 Transmitter Stages . . . . .	7
2.3 Receiver Design . . . . .	8
2.3.1 Raw Signal Buffering . . . . .	8
2.3.2 Symbol Detection . . . . .	8
2.3.3 Peak Detection . . . . .	9
2.3.4 Envelope Detection . . . . .	10
2.4 N-Ary Modulation . . . . .	11
<b>3 Implementation</b>	<b>13</b>
3.1 Software Architecture . . . . .	13
3.1.1 Inversion of Control . . . . .	14
3.2 Software Libraries . . . . .	14
3.2.1 PortAudio . . . . .	14
3.2.2 FFTW . . . . .	14
3.3 Audio Communication Modules . . . . .	15
<b>4 Verification</b>	<b>17</b>
4.1 Relative Data Rate . . . . .	17
4.2 Bandwidth and Frequency Range Selection . . . . .	19
4.2.1 Raw data . . . . .	19
4.2.2 Analysis . . . . .	19
4.3 Achievable Data Rates (CBOK) . . . . .	21
4.3.1 Raw Data . . . . .	22

4.3.2	Analysis . . . . .	22
4.4	Bandwidth and Frequency Range Selectivity (MCOK) . . . . .	24
4.4.1	Raw Data . . . . .	24
4.4.2	Analysis . . . . .	27
4.5	Noise Sensitivity (CBOK) . . . . .	27
4.5.1	Raw Data . . . . .	27
4.5.2	Analysis . . . . .	31
<b>5</b>	<b>Summary</b>	<b>35</b>
5.1	Summary . . . . .	35
5.2	Outlook . . . . .	35
	<b>Bibliography</b>	<b>37</b>

# List of Figures

1.1	A NAO robot kicking a ball at <i>RoboCup</i> 2016 in Leipzig. . . . .	3
1.2	Position of the NAO's microphones. . . . .	4
1.3	Position of the NAO's speakers. . . . .	4
2.1	Examples of up- and down-chirp signals. $T_s = \frac{1}{44\,100}$ s, $T_{\text{sym}} = 0.005$ s. . . . .	6
2.2	Auto- and cross-correlation of the chirp signals as seen in figure 2.1. . . . .	6
2.3	Layout of the packet in time domain. Each chirp is followed by a guard interval. The packet is announced by a preamble. . . . .	8
2.4	Buffers are aligned with an offset of one packet size. . . . .	8
2.5	The correlation of buffer and preamble shows significant peaks at the position of the preamble. Note, that the buffer contains one full packet between both peaks. . . . .	9
2.6	Autocorrelation of an up-chirp signal with $f_1 = 6$ kHz, $f_2 = 8$ kHz. The hysteresis peak detection detects multiple peaks in the vicinity of the correlation maximum. . . . .	11
2.7	Peak detection on the envelope of the auto correlation signal from figure 2.6. The envelope of the correlation is plotted in red. . . . .	11
2.8	Correlation of two up-chirps, $f_1 = 3$ kHz, $f_2 = 5$ kHz and $f'_1 = 3.5$ kHz, $f'_2 = 5.5$ kHz. The orthogonality properties are even better, compared to figure 2.1. . . . .	12
4.1	The byte error rate (4.1) improves with increasing bandwidth. Higher bandwidths also show less scattering of the error rate. . . . .	21
4.2	The byte error rate is nearly constant over the mean frequencies $f_{12} = \frac{f_1 + f_2}{2}$ of the chirps. Note that the bandwidth is chosen to be higher than 2 kHz and the all frequencies are contained in the speaker's bandwidth. . . . .	21
4.3	This plot shows the effective data rate, i.e. the fraction of the theoretically possible data rate that was actually achieved. The best results had a configuration of $T_{\text{Sym}} = 5$ ms and $T_{\text{G}} = 1$ ms. . . . .	22
4.4	The byte error rate decreases for lower packet frequencies. If the packet duration is lower than 30 ms, the byte error rate is significant. The outlier at $T_{\text{packet}} = 100$ ms might be caused by a synchronization error. . . . .	24

---

4.5	Spectrogram of the received noise. The noise generated by the robot's fans is clearly visible in the first subplot. The externally generated white noise is reasonably flat in the frequency range from 0.2 kHz to 10 kHz. . . . .	29
4.6	Spectrogram of a CBOK transmission with $f_1 = 5$ kHz and $f_2 = 7.5$ kHz. The transmission starts after 4 s of measurement. This transmission uses no pulse-shaping, the harmonics are observable quite clearly outside the transmission band. No external noise was generated in this measurement. . . . .	31
4.7	Spectrogram of a CBOK transmission with reduced volume. In comparison to the noise spectrogram without any signal, no clear transmission is detectable in the range 2.5 kHz to 5 kHz. This leads to very high error rates. . . . .	32
4.8	Spectrograms of MCOK transmission with different external noise levels. The two bands for the different chirp symbols are distinguishable in the plot. Note that due to a roll-off factor of $\beta = 0.1$ , the harmonics outside the transmission band vanished completely. . . . .	33

# List of Tables

4.2.1 Raw data of experiment 1. Frequencies $f_1$ , $f_2$ were varied, while $T_G$ and $T_{\text{Sym}}$ were kept constant. . . . .	20
4.3.1 Raw data for experiment 2. The chirp bandwidth and location are kept constant, while the symbol duration and guard interval is reduced to the minimum. The last data row shows that it is possible to transmit more data when the chirp bandwidth is higher. . . . .	23
4.4.1 Raw data for Experiment 3. Multi-Chirp Orthogonal Keying (MCOK) transmission is possible and yields high data rates, when the total bandwidth for the transmission is at least 5 kHz. . . . .	25
4.4.2 Measurement results for experiment 3b. The rolloff factor does not influence the data transmission capability when it is selected reasonably. For $\beta = 0.5$ the byte error rate is higher than 99 %. . . . .	26
4.5.1 Measurement results for experiment 4. The noise type references the signals described in figure 4.5. This data shows, that Chirp Binary Orthogonal Keying (CBOK) is not affected much by external noise. . . . .	28
4.5.2 Measurement results for experiment 4b. In this experiment, the volume of the transmitter was reduced to 30 % of its original volume, while keeping the noise levels from the previous experiment. . . . .	28
4.5.3 Measurement results of experiment 5. This data shows that the drops in error free data rate are higher for MCOK than for CBOK. . . . .	30





# List of Abbreviations

**FFT** Fast Fourier Transform

**FFTW** Fastest Fourier Transform in the West

**IFFT** Inverse Fast Fourier Transform

**CBOK** Chirp Binary Orthogonal Keying

**MCOK** Multi-Chirp Orthogonal Keying

**SPL** Standard Platform League

**SNR** Signal to Noise Ratio

**LOS** Line of Sight

**TCP** Transmission Control Protocol

**WGN** White Gaussian Noise

**HSU** Helmut Schmidt University of the Federal Armed Forces Hamburg

**IOC** Inversion of Control

**FDMA** Frequency Division Multiple Access

**TDMA** Time Division Multiple Access



# Chapter 1

## Introduction

Digital communication via acoustic channel has been around since the late 1950s and the sound of one of its most famous devices, the 50k modem, is still in the head of many people. However acoustic couplers are infamous for having slow data rates and producing intrusive sounds, hence they are usually only used for special applications, mostly in underwater sensor networks, where the alternative radio signal transmission would require low frequencies and therefore large antenna sizes [2, 3]. In normal environments aerial acoustic communication isn't used often, possible applications are smart devices, where the implementation of radio transmission would be too expensive or situations where inter-device communication must be observable by humans [1, 4].

This work shows that audible acoustic communication via aerial channel can be implemented on a cost-effective robotics platform. The communication is tested for robustness in an application-typical environment in the context of *RoboCup* technical challenges, and it is shown that acoustic communication is a feasible way to broadcast information relevant for robot soccer to multiple receivers.

After explaining the motivation of this thesis in section 1.1, the working principle of aerial acoustic communication using chirp-signals is be explained in chapter 2. Different experimental results are presented in chapter 4 and it is shown that acoustic communication between robots can indeed work in a dynamic environment. Finally future research topics are discussed in chapter 5.1.

### 1.1 Motivation

*RoboCup* is an international research contest that was founded in 1997 [5]. One department of *RoboCup* is robot soccer, a challenge that aims to develop robots that can win a soccer game against the human *FIFA* world champion until 2050. The Standard Platform League (SPL) is a subdivision of *RoboCup* soccer in which all teams use the same low-cost robotics platform. The current model is provided by the company *SoftBank Robotics*<sup>1</sup>.

The rule set of this league is currently quite different from the official *FIFA* rules [6], however

---

<sup>1</sup>formerly known as *Aldebaran Robotics*

every year the rules are updated to bring them closer to the official soccer rules. To investigate possible future rule changes, *Technical Challenges* are being held [7]. In 2016's *RoboCup* one of the challenges was to demonstrate ways of wireless inter-robot communication without the use of WiFi data transmission.

The implementation of a non-WiFi communication between robots developed for this thesis was developed specifically for this challenge. The achieved error free data rate and robustness during the competition was by a factor of about fifty times faster than the implementation of the runner-up team [8].

## 1.2 Requirements

The *RoboCup* 2016 No-WiFi-Challenge requires the communication method used to fulfill at least the following criteria.

**Distance between receiver and transmitter** The robots are required to transmit information over a range of at least 4.5 m and at most 6.5 m in direct line of sight.

**Minimum data rate** The first part of the challenge requires the robot to transmit 32 bits of position information in a period of 15 s. This yields a minimum required data rate of  $2.14 \text{ bit s}^{-1}$

**Robustness** The challenges are **be** judged in a convention hall environment, where other *RoboCup* events take place simultaneously. The communication scheme must be able to deal with environments with bad Signal to Noise Ratio (SNR).

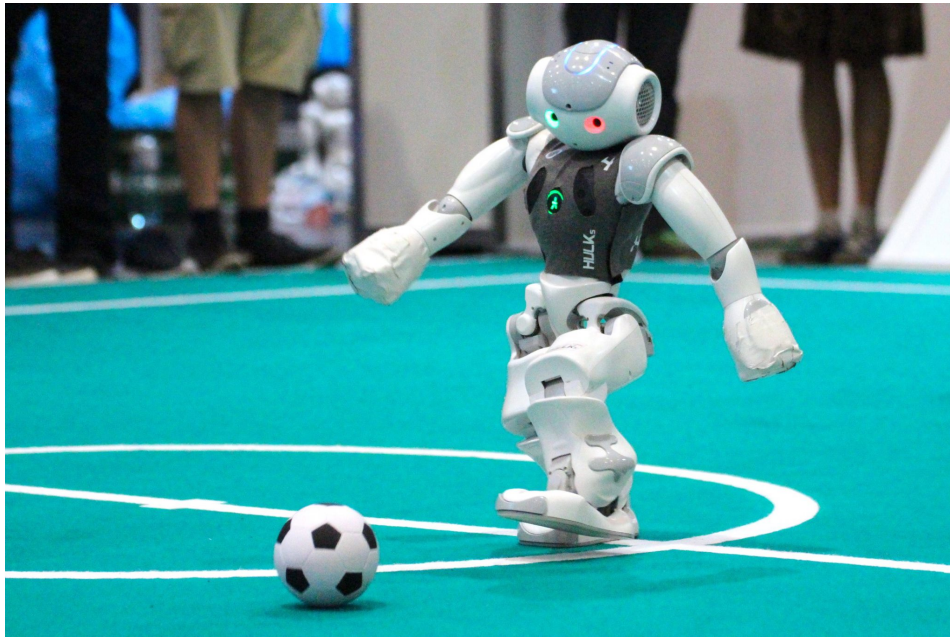
More requirements for the mode of communications arise from hardware restrictions of the *NAO* robotics platform.

**Audio bandwidth** The speakers of the *NAO* robotics platform have a documented frequency range from **600** Hz to 10 kHz [9]. Experiments showed that the upper frequency range of the *NAO*'s speakers is in the area of 7 kHz [10]. Therefore audio signals used for communication between *NAO* robots must be contained in this frequency range. This excludes ultrasonic sound.

## 1.3 The NAO Robotics Platform

The *NAO* robotics platform is a cost-effective robotics platform that is developed by the french company *Aldebaran Robotics* since 2004. The current version of the *NAO* system is V5. This platform is used as the standard platform in *RoboCup* SPL and is therefore used by many researchers all over the world to play soccer with (see figure 1.1).

For this thesis, only the audio hardware of the robot is relevant.



**Figure 1.1:** A NAO robot kicking a ball at RoboCup 2016 in Leipzig.

### 1.3.1 Audio Hardware

The NAO is equipped with speakers and microphones, both located in the head of the robot.

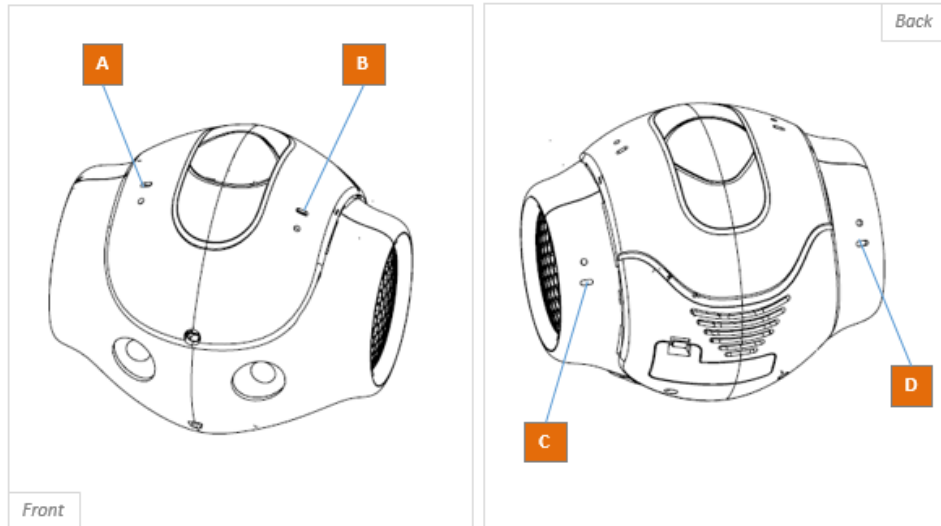
#### 1.3.1.1 Microphones

The H25 version of the V5 NAO robot has four microphones that are positioned at the upper side of its head (see figure 1.2). They allow sampling rates up to 48 kHz when using a mix of all four channels and sampling rates up to 16 kHz when the individual channels are required [9].

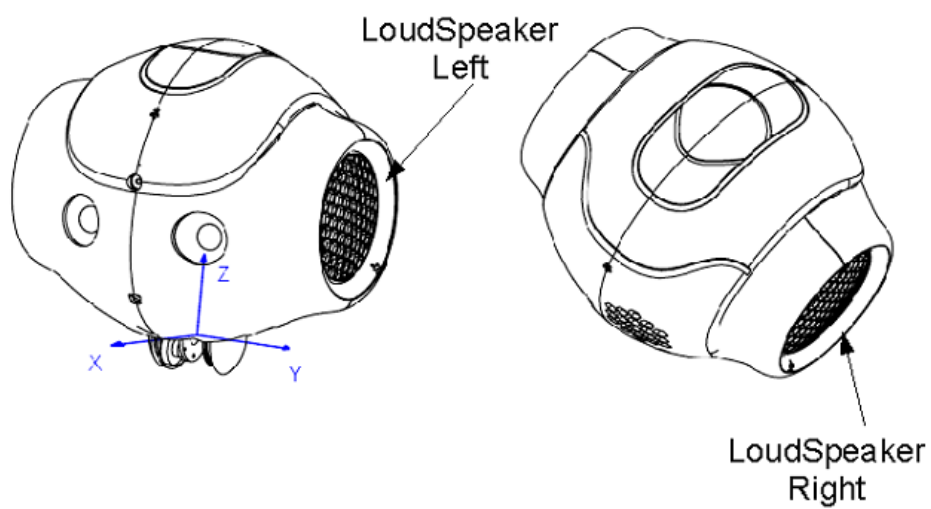
The NAO's microphones are of significantly higher quality than the robot's speakers, therefore it is not expected that they will limit the transmission capabilities for acoustic communication [10].

#### 1.3.1.2 Speakers

Since the first NAO version, the robot has two speakers located at the position where one would expect the ears of a human to be (see figure 1.3). The speakers of the NAO are quite large compared to e.g. smartphone speakers. However their frequency range is specified to be from 0.2 kHz to 10 kHz [9]. The older V3 version of the NAO has been measured to have a reasonably flat speaker response in the frequency range from 0.2 kHz to 7 kHz [10], which fits the observations made in chapter 4.



**Figure 1.2:** *Position of the NAO's microphones.*



**Figure 1.3:** *Position of the NAO's speakers.*

## Chapter 2

# Chirp-Based Communication

### 2.1 Working Principle

#### 2.1.1 Properties of Chirp Signals

The method of communication used in this thesis **relies on** is based on chirp signals. Chirps are a kind of sinusoidal signal that changes in frequency while maintaining a constant amplitude. If the frequency is raising over the duration of the signal, the signal is called *up-chirp*, if the frequency lowers *down-chirp*, respectively (2.1).

The chirp can be defined in continuous time domain

$$x(t) = \cos(2\pi f_1 t + \mu t^2) \quad (2.1)$$

or in discrete time with sampling rate  $T_s$  and  $N$  samples per symbol over a symbol duration of  $T_{\text{sym}}$ :

$$x(k) = \cos(2\pi f_1 k T_s + \mu (k T_s)^2) \quad (2.2)$$

with

$$\mu = 2\pi \frac{(f_2 - f_1)}{N}$$

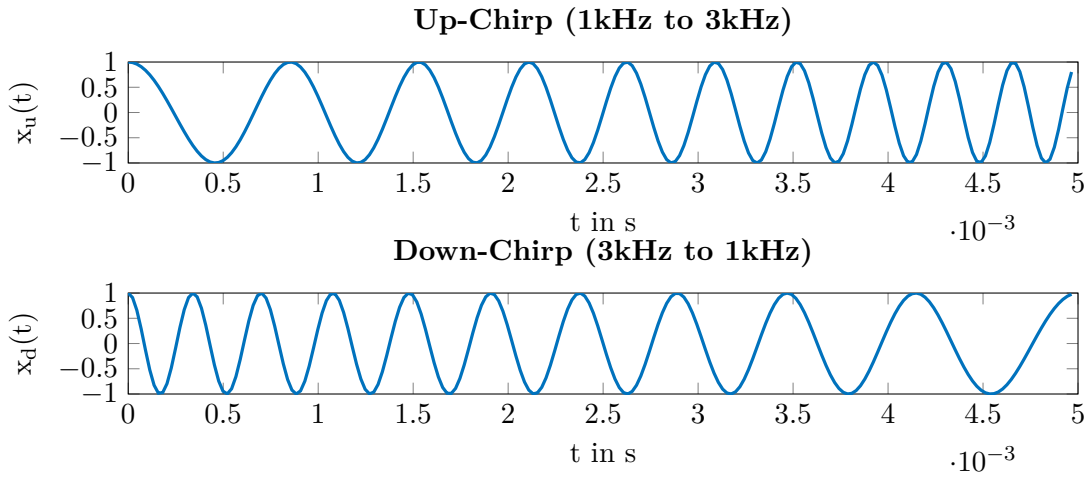
Two examples for chirp signals can be seen in figure 2.1.

Let  $B = f_2 - f_1$  the *bandwidth* of the chirp. If  $B > 0$ , the respective chirp is an up-chirp.

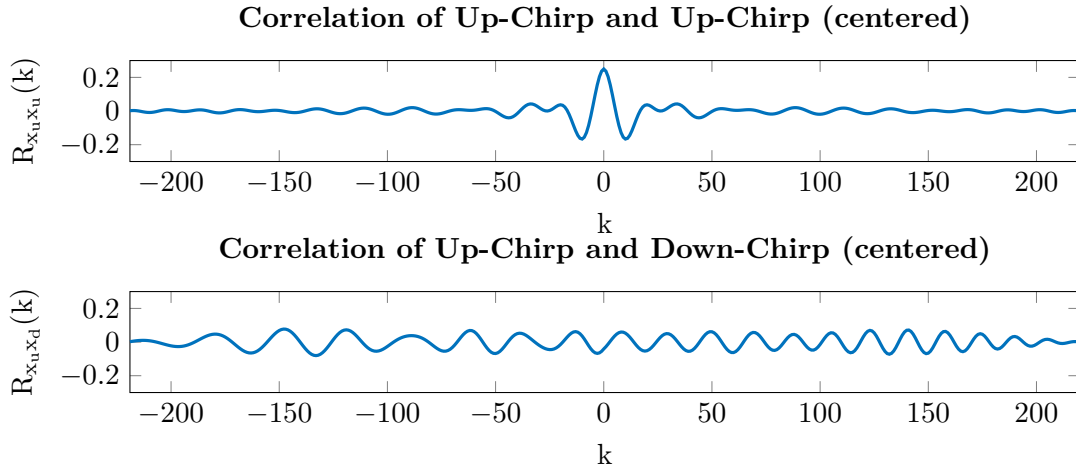
One key feature of up- and down-chirps is their quasi-orthogonality with respect to the standard correlation formula (2.3) [1].

$$R_{xx}(\tau) = \int_{-\infty}^{\infty} x^*(t) x(t + \tau) dt \quad (2.3)$$

The correlation function shows a significant peak at  $\tau = 0$  for the autocorrelation of a chirps (a discrete correlation is plotted in figure 2.2). This has two advantages:



**Figure 2.1:** Examples of up- and down-chirp signals.  $T_s = \frac{1}{44100}$  s,  $T_{sym} = 0.005$  s.



**Figure 2.2:** Auto- and cross-correlation of the chirp signals as seen in figure 2.1.

1. up- and down-chirps can be distinguished using a matched filter receiver, i.e. a correlation receiver
2. the peak can be used to synchronize the symbol clock for each symbol

## 2.2 Transmitter Design

### 2.2.1 Assumptions

It is assumed that a stream of information bits is provided to the transmitter implementation at a data rate much higher than the actual transmission rate of the acoustic communication. This is reasonable, as the data payload is software generated and the achievable data rate of aerial acoustic communication is orders of magnitude lower than the memory bandwidth of



any modern computer system. Therefore the transmission can start whenever payload data arrives at the transmitter.

### 2.2.2 Transmitter Stages

The design of the chirp-based transmitter is separated into different stages. First the incoming **bits stream** is separated into packages of a fixed size. Then all bits of each package will be mapped onto the transmission symbols. The time domain signals of those symbols are then concatenated to yield the audio signal that is to be transmitted. This audio signal will then be transferred to the audio hardware's internal buffers.

#### 2.2.2.1 Symbol Mapping

The approach proposed in [1] suggests to map 1-bits to up-chirps and 0-bits to down-chirps, respectively. The choice of which bit to map to which chirp signal is arbitrary, as long as the size of the symbol space is equal to two. This mapping scheme is called CBOK [1]. In 2.4 an approach to use a larger symbol space to acquire higher data rates is presented.

#### 2.2.2.2 Packet Assembly

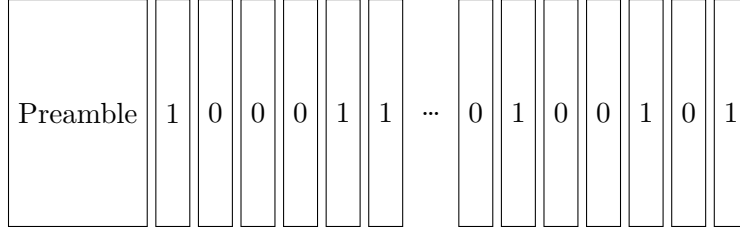
To make the transmission robust against synchronization errors, a guard interval can be inserted between symbols. The guard interval is a fixed length signal of silence and is therefore not influencing the correlation result in the matched filter receiver. Robustness against synchronization error is especially relevant when using chirps of higher frequency, see 2.3.4.

As the receiver can not predict when a transmission will start, a preamble signal is inserted before any packet will be transmitted. The preamble signal needs to meet two requirements:

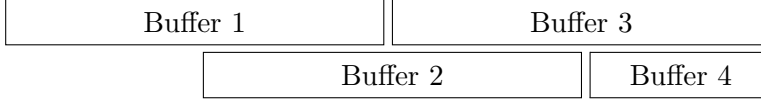
1. The receiver must be able to distinguish it from all symbols
2. To get information about the exact timing of the preamble start, it must be strictly aperiodic.

Those requirements can be fulfilled by using a chirp signal that is not contained in the symbol set. To maximize the chance of receiving the preamble signal, it can be chosen to be significantly longer than the symbols. Lee et al. propose to use an up-chirp of a duration  $T_{\text{preamble}} = 5T_{\text{sym}}$ .

The structure of a complete packet has been visualized in figure 2.3.



**Figure 2.3:** Layout of the packet in time domain. Each chirp is followed by a guard interval. The packet is announced by a preamble.



**Figure 2.4:** Buffers are aligned with an offset of one packet size.

## 2.3 Receiver Design

### 2.3.1 Raw Signal Buffering

Lee et al. use chirp based communication to broadcast a 16-bit payload repeatedly. The use case considered in this work depends on receiving multiple consecutive uncorrelated information packets over an uncertain time period. Hence it is required to identify each individual that might has been sent by the transmitter and pass it through the receiving process.

To allow for a detection of individual packages, each package must be fully contained in exactly one single buffer. Therefore the buffer layout as illustrated in figure 2.4 is used. The buffer length is chosen such that at all possible signal offsets, one full packet including its preamble is contained in every buffer. The buffer size  $N_{\text{buffer}}$  is hence twice the length of a complete packet (2.4).

$$N_{\text{buffer}} = 2T_s (N_{\text{sym}} (T_{\text{sym}} + T_{\text{guard}}) + T_{\text{preamble}} + T_{\text{guard}}) \quad (2.4)$$

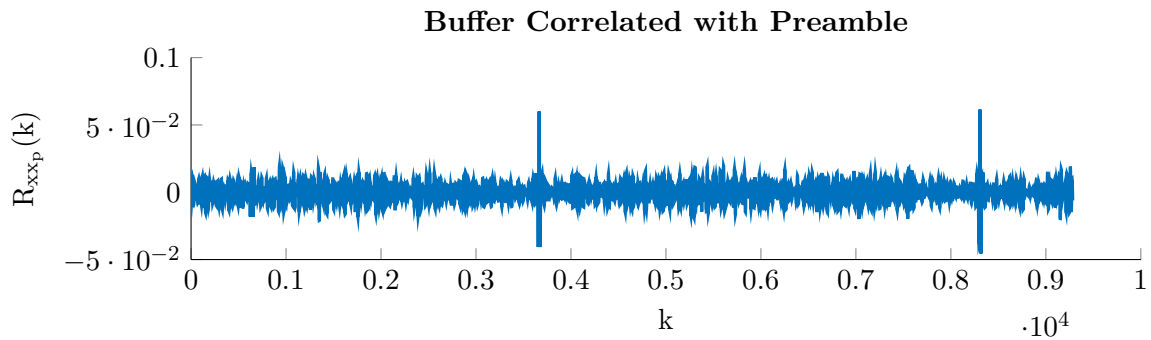
The buffers have an offset of half of the buffer size, as visualized in figure 2.4.

### 2.3.2 Symbol Detection

For each buffer, the receiver tries to find the preamble signals in it. This is done by envelope detection on the correlation of the whole buffer with the preamble signal (see 2.3.4). If the preamble has been found, i.e. the maximum correlation value was higher than a threshold  $t_{\text{preamble}}$ , and the position of the preamble indicates that a full packet is contained in the buffer, the symbol detection will be applied.

Correlation is implemented using fast convolution after applying Fast Fourier Transform (FFT). Let  $x$  denote the received signal and  $\bar{x}_p$  the reversed preamble signal, zero padded to length  $N_{\text{buffer}}$ . Then

$$X(k) \bullet \text{---} \circ x(n) \quad (2.5)$$




**Figure 2.5:** The correlation of buffer and preamble shows significant peaks at the position of the preamble. Note, that the buffer contains one full packet between both peaks.

is its representation in frequency domain. The correlation between preamble and a buffer can be calculated using the following algorithm:

```
function convolute(buffer, preamble)
    let bufferFreq = fft(buffer);
    let preambleFreq = fft(reverse(preamble));

    for i = 0 to size(preambleFreq)
        conv[i] = bufferFreq[i] * preambleFreq[i];
    end

    return ifft(conv);
end
```

Note that this algorithm is a naïve implementation and can be optimized by pre-calculating the FFT of the **preamble**. 

The correlation of a recorded buffer with the preamble is plotted in figure 2.5.

Symbol Detection is done by splicing the payload part of the buffer into individual buffers of the length of one symbol and one guard interval. Then a matched filter receiver determines the corresponding symbol of each individual buffer, by correlating it with all possible symbols and deciding for the one with the maximum correlation value. The detected symbols are mapped onto their respective bit values and concatenated into an output bit stream.

### 2.3.3 Peak Detection

To allow the receiver to synchronize with the transmission signal, peaks in a large buffer need to be detected reliably. As the peaks in a typical correlation between a buffer and the preamble signal are quite distinct (see figure 2.5), they can be detected using a simple hysteresis based approach.

The idea of the hysteresis based peak detection is, to find multiple local maxima in the signal that are larger than a specified fraction of the global maximum. To find those local maxima,

the following algorithm is used.

---

```

function findPeaks(signal, threshold)
    let globalMaximum = max(signal);

    let results = [];

    let i = 0;
    while (i < size(signal))
        let sample = signal[i];
        if (sample > threshold * globalMaximum)
            candidate = sample;
            candidateIndex = i;
            while (i < size(signal) && signal[i] > threshold *
                globalMaximum)
                if (signal[i] > candidate)
                    candidate = signal[i];
                    candidateIndex = i;
                end
                i++;
            end

            results.add(candidateIndex);
        end
        i++;
    end

    return results;
end

```

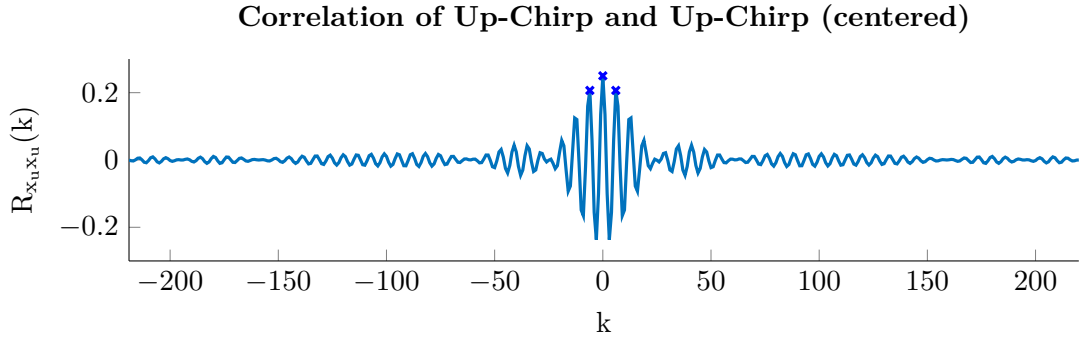
---

The main drawback of this algorithm is its sensitivity to high frequency sidelobes of the correlation signal. This can be prevented by detecting the maxima of the envelope of the correlated signal, as explained in the next section.

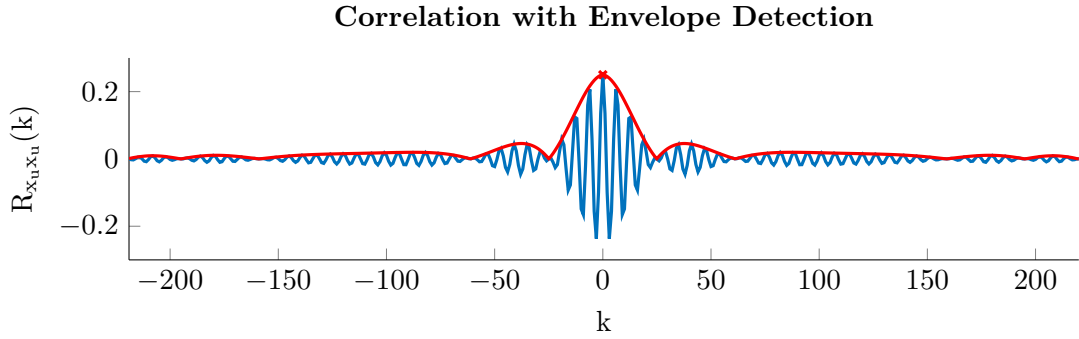
### 2.3.4 Envelope Detection

The correct detection of peaks in correlation data is critical for a successful data transmission using chirps. When the chirp signals are of high frequency compared to the symbol duration, the correlation result is not as distinct as in figure 2.1. The correlation of two symbols of higher frequencies is plotted in figure 2.6.

When using higher frequency chirps, the hysteresis based peak detection (see 2.3.3) will produce erroneous results. This is especially problematic in the synchronization phase, where the exact offset of the preamble in the buffer is to be determined. The resulting offset in symbol alignment consequently increases the bit error probability. To cope with this problem, envelope detection is used [1]. The envelope of a signal can be calculated using the analytic signal, the analytic signal itself can be calculated very efficiently, introducing only one additional multiplication, in the implementation of the correlation algorithm [11].



**Figure 2.6:** Autocorrelation of an up-chirp signal with  $f_1 = 6$  kHz,  $f_2 = 8$  kHz. The hysteresis peak detection detects multiple peaks in the vicinity of the correlation maximum.



**Figure 2.7:** Peak detection on the envelope of the auto correlation signal from figure 2.6. The envelope of the correlation is plotted in red.

The discrete time analytic signal can be determined by omitting negative frequencies from the spectrum.

$$\hat{X}(f) = \begin{cases} 0, & \text{for } f < 0 \\ X(f), & \text{for } f = 0 \\ 2X(f), & \text{for } f > 0 \end{cases} \quad (2.6)$$

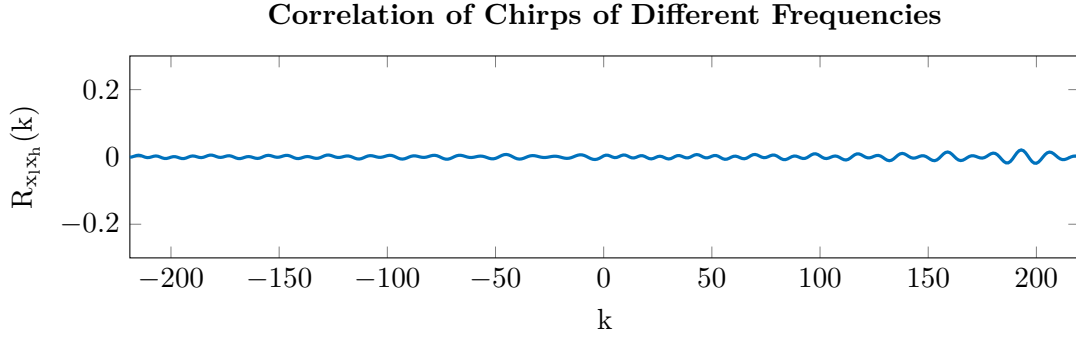
The envelope of the correlation can be determined from (2.6) using Inverse Fast Fourier Transform (IFFT). Figure 2.7 shows that the peak in the correlation can be found exactly using envelope detection<sup>1</sup>.

## 2.4 N-Ary Modulation

When only two different symbols are used, the physical limitation of the transmission data rate is based on the total duration of each symbol and evaluates to

$$R_{\text{CBOK, physical}} = \frac{1}{T_{\text{sym}} + T_{\text{guard}}} \cdot 1 \text{ bit} \quad (2.7)$$

<sup>1</sup> As the position of negative frequencies in the FFT buffers varies over different implementations (esp. between *matlab* and *FFTW*), special care should be taken when implementing envelope detection.



**Figure 2.8:** *Correlation of two up-chirps,  $f_1 = 3$  kHz,  $f_2 = 5$  kHz and  $f'_1 = 3.5$  kHz,  $f'_2 = 5.5$  kHz. The orthogonality properties are even better, compared to figure 2.1.*

By realizing the fact, that chirp signals in different frequency areas have similar orthogonality properties as up- and down-chirps that share a frequency band (see figure 2.8), it is possible to increase the size of the symbol space.

Using a reasonable frequency spacing of 500 Hz and taking into account the bandwidth of the transmitter's speakers, a modulation scheme with four symbols should be considered to double the data rate (2.8). We call this modulation scheme MCOK.

$$R_{\text{MCOK, physical}} = \frac{2 \text{ bits/symbol}}{T_{\text{sym}} + T_{\text{guard}}} = 2R_{\text{CBOK, physical}} \quad (2.8)$$



## Chapter 3

# Implementation

One of the main goals of this thesis is to provide a working implementation of aerial acoustic communication between two cost effective robots. The implementation has to meet the following requirements:

**Efficiency** The NAO robotic system lacks of dedicated audio hardware. Also the internal CPU is not up to date regarding performance. Therefore the implementation must be efficient enough to run in real time on the NAO.

**Reusability** In case of rule changes (see section 1.1), the communication system should be able to transmit arbitrary data in a way that is easy to use from the existing soccer codebase.

**Programming language** As the existing codebase is written in C++, the acoustic communication software should also be written in C++.

Thanks to work done by the SPL-Team *HULKS*, there is an existing C++-framework that allows for high development speeds and is encouraging a modular software architecture. However, as no audio hardware was used by the *HULKS* before, the hardware access had to be implemented during this thesis.

The following sections are a qualitative description of the software developed for this thesis. The detailed interfaces and quirks of the software can be **best understood**, when a copy of the commented **code close** by.

### 3.1 Software Architecture

A detailed description of the *HULKS*' software architecture is beyond the scope of this thesis. To get an idea about the core concepts of the infrastructure, the most important ideas are explained in this section.

### 3.1.1 Inversion of Control

The core concept of the software framework is a form of the Inversion of Control (IOC) idea [12]. This means, that every encapsulated unit in the software does only state its dependencies, but does not instantiate or collect them on its own. Instead an overlying system resolves the dependencies at runtime and provides all necessary information to the modules.

The *HULKs* framework is built in a pipeline-like fashion. Every *Module* has *Dependencies* and *Productions*. All modules are called in a sequence that ensures that all dependencies are existing at the time of the call. After the module has been called, its productions are available to other modules for further processing. This approach has a number of benefits, including the fact that individual modules can be substituted against different implementations without ever touching the previous implementation or any other modules depending on this module<sup>1</sup>.

## 3.2 Software Libraries

The implementation of a chirp-based acoustic communication requires two things to work reliably and with high speed.

1. Algorithms like FFT and IFFT
2. Communication with audio hardware, i.e. microphones and speakers of the NAO.

As the development of audio hardware drivers and FFT-algorithms is not in scope of this thesis, publicly available software libraries were used to ensure the functionality stated above.

### 3.2.1 PortAudio

To communicate with the robot's audio hardware, the free, open source, cross-platform audio library *PortAudio* [13] was used. This C-library allows to access the NAO's microphones and speakers in a straightforward and consistent manner. It allows to control settings like the sampling frequency  $f_s$ , the quantization bitrate and buffer sizes directly from the calling code.

### 3.2.2 FFTW

Fastest Fourier Transform in the West (FFTW) claims to be a very fast implementation of the discrete fourier transform [14]. It provides the user with many different algorithms to transform buffers of arbitrary length and dimensionality between time and frequency domain. FFTW ensures good runtime performance by measuring different algorithms on the actual hardware that the code is executed on. Due to a different layout of the frequency data in *matlab* and FFTW, it is very important to double-check the documentation when translating *matlab* code to C++, though.

---

<sup>1</sup>A more detailed explanation will be published in the 2016's *HULKs* team research report.



### 3.3 Audio Communication Modules

To implement chirp-based acoustic communication on the NAO robotics platform, the following modules were developed.

**AudioRecorder** The **AudioRecorder** module grabs audio samples from the NAO's microphones and converts them into buffers, which can be consumed by other modules. The provided data type is called **MicrophoneData**.

**AudioPlayer** The **AudioPlayer** depends on **PlaybackData**. It converts the provided data to a format that can be played back by the NAO and provides it to the abstraction of the audio hardware. The implementation of **PlaybackData** and **MicrophoneData** is identical, which means it is possible to route the microphone input to the NAO's speakers directly.

**ChirpTransmitter** The **ChirpTransmitter** consumes **NoWifiData** and transforms it into **PlaybackData**. This module contains the complete implementation of chirp-based acoustic data transmission, including symbol mapping, packet assembly, and pulse shaping.

**ChirpReceiver** The **ChirpReceiver** consumes **MicrophoneData** and transforms it into **NoWifiData**. It contains the implementation of all receiver stages described in section 2.3, i.e. buffer layout, synchronization, message splitting, and symbol detection.

The **NoWifiData** is provided by the *RoboCup* 2016 Communication Tester application (see section 4.1), which was integrated into the *HULKS* codebase by Arne Hasselbring.





## Chapter 4

# Verification

In this chapter the results of five experiments are presented. The experiments aim to show the applicability of CBOK and MCOK to inter-robot communication and try to reach the limits regarding speed and band-efficiency of aerial acoustic communication between two robots.

The experiments were held in a controlled environment, using a sound-proof room in the laboratory of Helmut Schmidt University of the Federal Armed Forces Hamburg (HSU). Due to size limitations of the room, the Line of Sight (LOS) distance between receiver and transmitter was chosen to be 2 m. In cases where external noise was generated, two speakers with a reasonably flat frequency response in the range of the transmission signal were used to playback a computer-generated White Gaussian Noise (WGN) signal.

For each configuration of the communication at least three different measurements have been conducted to reduce the risk of outliers or measurement errors<sup>1</sup>.

### 4.1 Relative Data Rate

All experiments were conducted and evaluated using the official *RoboCup* 2016 Communication Tester **Java** application [15]. This application provides functionality to connect to receiver and transmitter via Transmission Control Protocol (TCP). It can then provide randomly generated data to the transmitting robot. Whenever the receiving robot receives a packet of audio transmitted data, it will send this data to the communication tester, where the data is validated. Each experiment takes 15 s, in which up to 10 000 Byte are provided to the transmitter. After the experiment is over, the communication tester displays the statistics of the experiment, calculating the number of bytes transmitted by the robot, the number of correct bytes transmitted by the robot, the correctness ratio of the transmitted bytes (4.1), and the maximum error-free data rate (4.2).

$$C_{\text{byte}} = \frac{\# \text{ Bytes transmitted correctly}}{\# \text{ Bytes transmitted in total}} = 1 - E_{\text{byte}} \quad (4.1)$$

---

<sup>1</sup>To get all measured data in a machine readable form, please contact the author or see <https://github.com/fpoppinga/pa>.

$$R_{\max} = \frac{\# \text{ Bytes transmitted correctly}}{15 \text{ s}} \quad (4.2)$$



The ease of use of the *RoboCup* Communication Tester comes at a **tradeoff**. To allow the receiver to provide a byte stream, even if the transmission was disturbed for a time period, the receiver is allowed to provide

1. an offset of the following byte stream in the total message
2. an arbitrary length *payload* byte stream, beginning at the given offset

The decision, which lengths of byte streams are provided by the receiver and how often the offset is transmitted, is up to the implementation. As the data rate measured by the communication tester depends only on the total number of payload bytes received at the correct offset, this value is only an indication of the absolute amount of correctly transmitted data. Consider the following example:

The transmission is configured to transmit 16 bit per packet. Whenever three packets are transmitted, an additional packet, containing only the offset is transmitted as well. Let's assume this transmission takes 15 s in total. When the receiver receives the packets and the offset, it provides this information to the communication tester. The tester evaluates the information and calculates a total data rate of  $R_{\max} = 3.2 \text{ bit s}^{-1}$  (see (4.2)).

The fact that there is a protocol overhead of 2 Byte for the data offset is neglected at the communication tester. To be able to compare the quality of the data transmission amongst measurements with different settings for the symbol duration or the rate of offset packets, the measured rate has to be normalized. One possible way of normalization is, to relate the measured maximum error free data rate to the duration it takes for one packet to be transmitted, as proposed in (4.3). The resulting value is a measure related to synchronization errors, where whole packets get lost in the transmission, as well as to the byte error rate. However the actual symbol duration does not influence this value. Let  $N_{\text{payload}}$  be the number of *payload* packets transmitted before a offset packet is transmitted and  $P$  be the amount of data contained in each payload packet. Then

$$R_{\text{achievable}} = \frac{N_{\text{payload}}}{N_{\text{payload}} + 1} \cdot \frac{P}{T_{\text{packet}}} \quad (4.3)$$

where  $T_{\text{packet}}$  is calculated using the number of bits per symbol  $N$  and the fact that the preamble has the duration of five symbol durations  $T_{\text{Sym}}$

$$T_{\text{packet}} = \underbrace{(T_{\text{Sym}} + T_{\text{G}})}_{\text{one bit}} \cdot N + \underbrace{5T_{\text{Sym}} + T_{\text{G}}}_{\text{preamble}} \quad (4.4)$$

To get an idea what  $R_{\text{achievable}}$  is about, reconsider the previous example. If the transmission is configured as stated above,  $T_{\text{packet}} = 3.75 \text{ s}$ . The number of packets transmitted per offset packet  $N_{\text{payload}} = 3$ .

$$R_{\text{achievable}} = \frac{3}{4} \cdot \frac{16 \text{ bit}}{3.75 \text{ s}} = 3.2 \text{ bit s}^{-1} \quad (4.5)$$

The readings of the communication tester would now be normalized with  $R_{\text{achievable}}$ , yielding a *Relative Data Rate* of  $R_{\text{relative}} = 1$  (see (4.6)). This means, all packets have been transmitted successfully, i.e. there were no synchronization errors or packet losses, and all received bytes have been correct.

$$R_{\text{relative}} = \frac{R_{\text{max}}}{R_{\text{achievable}}} \quad (4.6)$$

This allows for example to measure the influence of the guard interval length. If a transmission with longer guard intervals was compared to one with a very short guard interval duration,  $R_{\text{max}}$  could be better for the faster configuration, even though it might have dropped more packets due to synchronization errors.

## 4.2 Bandwidth and Frequency Range Selection

In this experiment CBOK is used as a transmission scheme. The robots are placed 2 m apart and with a direct LOS. The symbol duration  $T_{\text{Sym}} = 5 \text{ ms}$  and the guard interval  $T_{\text{G}} = 4 \text{ ms}$  are constant over all variations of this experiment and were chosen based on prior experience with CBOK communication, where this values showed a robust transmission.

The frequencies defining the up- and down-chirp signals used in the communication scheme were then varied to determine

1. the optimal bandwidth  $B = f_1 - f_2$  of the chirp communication
2. the minimum and maximum viable chirp frequencies  $f_{1,\text{min.}}$   $f_{1,\text{max}}$

### 4.2.1 Raw data



The measurement results for this experiment can be reviewed in table 4.2.1.

### 4.2.2 Analysis

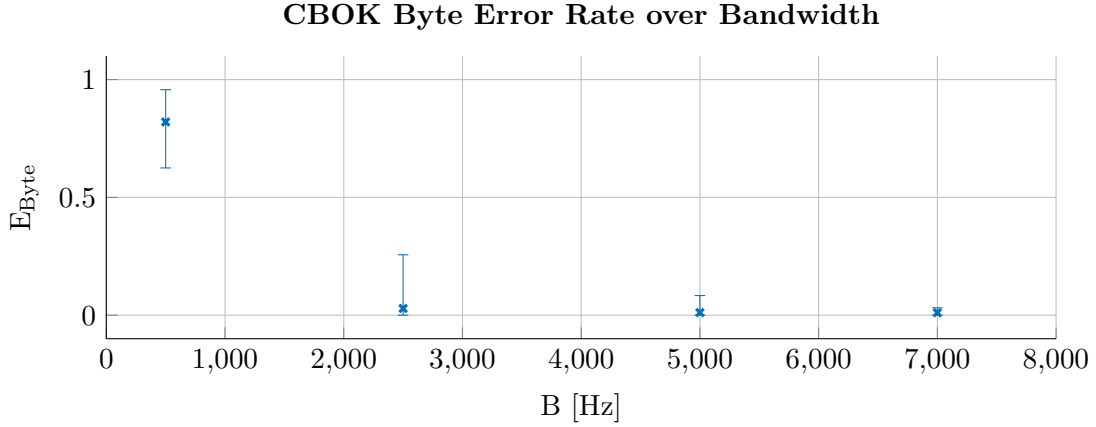
The measurements show the fact that increasing bandwidth of the chirp signals leads to a lower byte error rate (see figure 4.1). This is not too surprising, as the following implication holds:

$$f_1 \rightarrow f_2 \Rightarrow x_{\text{u}} \rightarrow x_{\text{d}} \quad (4.7)$$

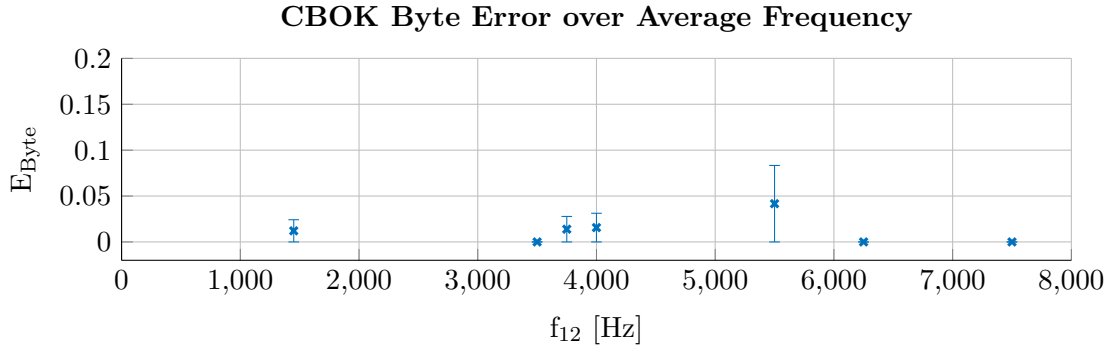
The orthogonality properties of up- and down-chirp signals are no longer given, when the bandwidth of the individual chirps is relatively low. Therefore the correlation receiver will consequently produce a higher detection error rate.

$f_1$ [Hz]	$f_2$ [Hz]	$T_{\text{Sym}}$ [ms]	$T_G$ [ms]	Data Rate [Bytes $^{-1}$ ]	Correct Bytes	Total Bytes
5000	10000	5	4	8.53		
				8.93		
				8.33		
1000	6000	5	4	8.53		
				8.53	128	128
				7.87	118	120
3000	8000	5	4	7.47	112	112
				1.47	22	24
				8.53	128	128
7500	10000	5	4	7.47	112	112
				3.40	51	52
				7.73	116	156
5000	7500	5	4	7.47	112	112
				7.47	112	112
				8.20	123	124
2500	5000	5	4	7.47	112	112
				9.33	140	144
				8.80	132	132
200	2700	5	4	8.53	128	128
				8.07	121	124
				9.27	139	140
2500	3000	5	4	0.87	13	304
				3.20	48	128
				1.07	16	132

**Table 4.2.1:** Raw data of experiment 1. Frequencies  $f_1$ ,  $f_2$  were varied, while  $T_G$  and  $T_{\text{Sym}}$  were kept constant.



**Figure 4.1:** The byte error rate (4.1) improves with increasing bandwidth. Higher bandwidths also show less scattering of the error rate.



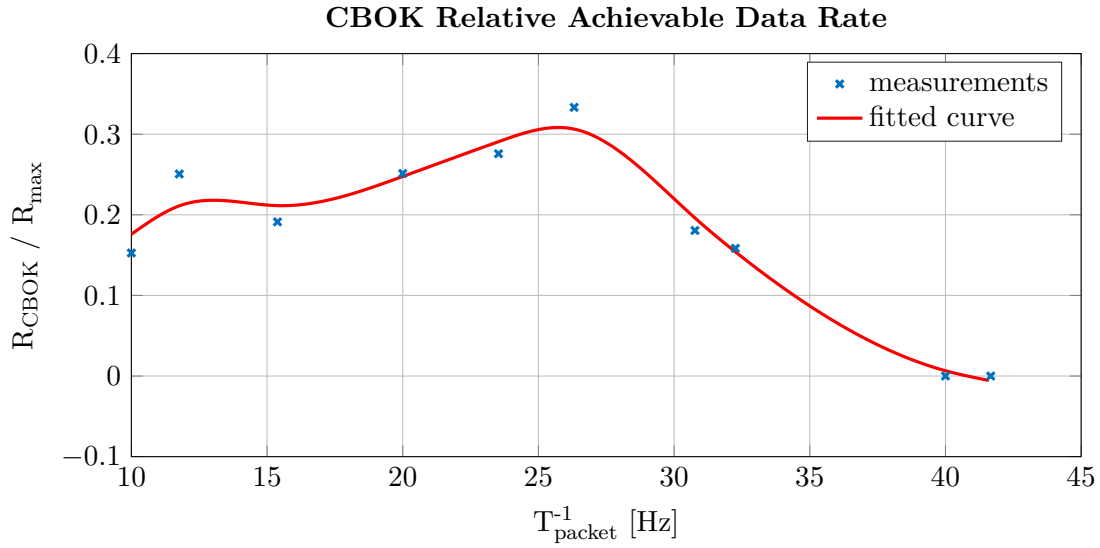
**Figure 4.2:** The byte error rate is nearly constant over the mean frequencies  $f_{12} = \frac{f_1 + f_2}{2}$  of the chirps. Note that the bandwidth is chosen to be higher than 2 kHz and the all frequencies are contained in the speaker's bandwidth.

Another interesting information gathered from this measurements is the following: If frequencies contained in the speaker's frequency band are chosen for the chirp signals, the byte error rate is independent from the center frequency of the transmission. This shows that the transmission channel response is flat enough to allow for error-free CBOK communication in the frequency range from 1 kHz to 8 kHz.

This leads to the conclusion that the choice of a big enough bandwidth is more important than the location of the transmission band within the viable frequency range of the NAO's speakers.

### 4.3 Achievable Data Rates (CBOK)

This experiment evaluates the maximum possible data rate for a CBOK transmission using a fixed frequency range and varying only the durations of the symbol  $T_{\text{Sym}}$  and the guard interval  $T_G$ . The lower frequency is selected such that  $f_1 = 2.5$  kHz and the upper frequency



**Figure 4.3:** This plot shows the effective data rate, i.e. the fraction of the theoretically possible data rate that was actually achieved. The best results had a configuration of  $T_{\text{Sym}} = 5 \text{ ms}$  and  $T_G = 1 \text{ ms}$ .

such that  $f_2 = 5 \text{ kHz}$ .

The symbol duration and the guard interval are initialized with values of  $T_{\text{Sym}} = T_G = 10 \text{ ms}$  and then decreased until no transmission is possible anymore. To compare the effectiveness of the different settings, the relative data rate (4.6) is considered (see section 4.1).

### 4.3.1 Raw Data

The measurement results for this experiment can be reviewed in table 4.3.1.

### 4.3.2 Analysis

The transmitted data rate is expected to be higher for shorter packet durations. Therefore the robustness of the transmission is evaluated using the relative transmission data rate (4.6). From figure 4.3 one can see that there is a sweet spot for a packet rate  $T_{\text{Packet}}^{-1}$  where the relative data rate is maximized.

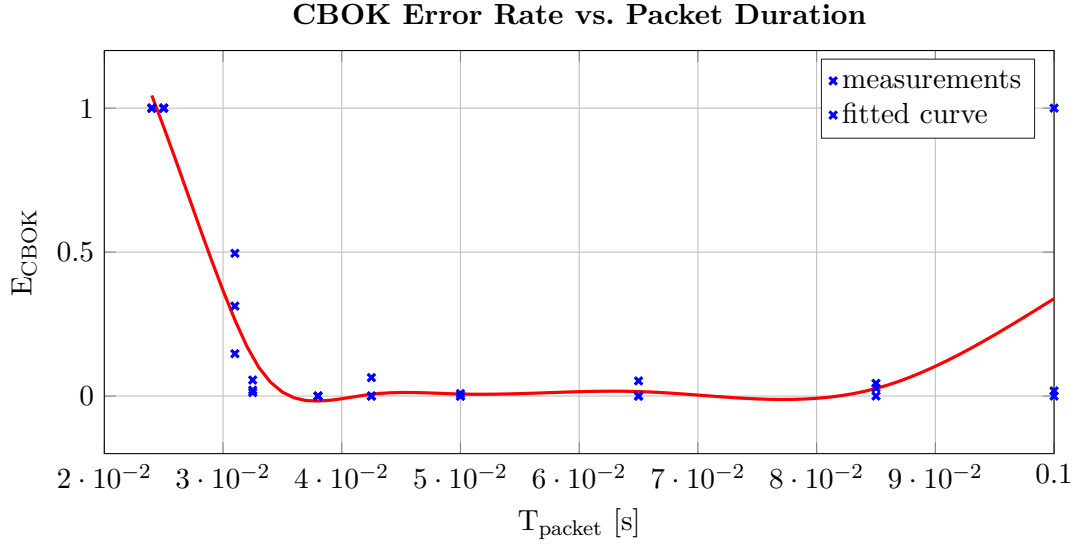
It is not possible to get any information about the optimal relationship between guard interval and symbol duration from figure 4.3. However, as the guard intervals do not carry information and only improve robustness against synchronization errors, they can be chosen to be of shorter duration as the symbols.

If the byte error rate is plotted over the symbol duration it is possible to observe that the byte error rate decays for longer packet durations. The outlier for the 100 ms packet duration can be explained as a synchronization error, as no bytes were received in that case anyway. This behavior is expected, as for very short chirp durations the number of samples per chirp is



$f_1$ [Hz]	$f_2$ [Hz]	$T_{\text{Sym}}$ [ms]	$T_G$ [ms]	Data Rate [Bytes $s^{-1}$ ]	Correct Bytes	Total Bytes
2500	5000	10	10	3.2	48	48
				0	0	0
				3.67	55	56
2500	5000	5	10	4.27	64	64
				4.27	64	64
				4.7	72	76
2500	5000	10	5	4.27	64	64
				4.33	65	68
				4.67	70	72
2500	5000	5	5	7.47	112	112
				7.67	115	116
				7.47	112	112
2500	5000	2.5	5	4.53	68	72
				10.8	162	164
				9.67	145	148
2500	5000	2.5	2.5	0	0	0
				0	0	0
				0	0	0
2500	5000	5	1	12.8	192	192
				12.8	192	192
				13.87	208	208
2500	5000	3	1	0	0	0
				0	0	0
				0	0	0
2500	5000	4	1	8.07	121	240
				7.73	116	136
				7.2	176	256
2500	7500	2.5	0.5	22.4	336	348
				22.6	339	355
				23.53	353	360

**Table 4.3.1:** Raw data for experiment 2. The chirp bandwidth and location are kept constant, while the symbol duration and guard interval is reduced to the minimum. The last data row shows that it is possible to transmit more data when the chirp bandwidth is higher.



**Figure 4.4:** The byte error rate decreases for lower packet frequencies. If the packet duration is lower than 30 ms, the byte error rate is significant. The outlier at  $T_{\text{packet}} = 100$  ms might be caused by a synchronization error.

substantially lower. Given a sampling rate of  $f_s = 44\,100$  Hz, the number of samples in a 2 ms chirp gets as low as  $n = 88$ . Note, that the resolution of FFT depends on sampling frequency as well as the number of samples recorded: the number of frequency bins  $N_{\text{bins}}$  is calculated as in (4.8):

$$N_{\text{bins}} = \frac{n_{\text{samples}}}{2} \quad (4.8)$$

which yields a resolution per bin of

$$d_f = \frac{f_s}{2 \cdot N_{\text{bins}}} \quad (4.9)$$

when plugging in the values from above, the resolution of FFT for very short chirps drops down to  $d_f = 501.3$  Hz. This is in close proximity to the actual chirp's bandwidth of  $B = 2500$  Hz. This coarse frequency resolution can then lead to detection errors in the matched filter receiver. One possible way to overcome this limitation is to increase the chirp bandwidth or, to increase the data rate in the same band, by using MCOK (see section 2.4) as the modulation scheme.

## 4.4 Bandwidth and Frequency Range Selectivity (MCOK)

### 4.4.1 Raw Data

The raw data for this experiment can be reviewed in table 4.4.1.

$f_1$ [Hz]	$f_2$ [Hz]	$f'_1$ [Hz]	$f'_2$ [Hz]	$T_{\text{Sym}}$ [ms]	$T_G$ [ms]	Data Rate [Bytes s <sup>-1</sup> ]	Correct Bytes	Total Bytes
2500	5000	5000	7500	5	1	25.6	384	384
						18.2	273	300
						28.27	424	428
500	4000	4000	7500	5	1	12.47	187	196
						8.6	129	140
						15.13	227	276
2500	3250	3250	5000	5	1	0	0	0
						0	0	0
						0	0	0
2500	5000	5000	7500	4	1	29.87	448	448
						29.27	439	456
						25.6	384	384

**Table 4.4.1:** Raw data for Experiment 3. MCOK transmission is possible and yields high data rates, when the total bandwidth for the transmission is at least 5 kHz.

$f_1$ [Hz]	$f_2$ [Hz]	$f'_1$ [Hz]	$f'_2$ [Hz]	$T_{\text{Sym}}$ [ms]	$T_G$ [ms]	Rolloff	Data Rate [Byte s <sup>-1</sup> ]	Correct Bytes	Total Bytes
2500	5000	5000	7500	5	1	0.1	25.6	384	384
							28.6	429	432
							27.6	414	416
							25.6	384	384
							16.8	252	256
							17.47	262	300
2500	5000	5000	7500	5	1	0.5	0.07	1	384
							0.07	1	396
							0.13	2	388

**Table 4.4.2:** Measurment results for experiment 3b. The rolloff factor does not influence the data transmission capability when it is selected reasonably. For  $\beta = 0.5$  the byte error rate is higher than 99 %.

### 4.4.2 Analysis

As the type of symbol that is used in MCOK is the same as in CBOK, the properties of the transmission regarding frequency selectivity or bandwidth constraints are the same. For the same symbol durations the same bandwidth requirements hold. As MCOK requires chirps in different frequency ranges, the overall bandwidth needs to be higher for MCOK, if no frequency overlap between different chirps is accepted.

In the second part of the experiment the roll-off factor for the pulse-shaping filter was evaluated. Measurements show that a roll-off factor  $\beta = 0.1$  does not negatively influence the transmission error rate. If  $\beta$  is chosen to be too high, e.g.  $\beta = 0.5$  the error rate exceeds to  $E_{\text{byte}} > 0.99$ . The synchronization error rate is not affected by this change, as can be seen by the relatively constant number of total transmitted bytes over the different roll-off factors (data from table 4.4.1).

## 4.5 Noise Sensitivity (CBOK)

To simulate bad transmission conditions, externally generated noise is used. This noise is played back using studio quality speakers. The channel from the noise generator to the *NAO*'s microphones is reasonably flat in the frequency range 0.2 kHz to 10 kHz (see figure 4.5), which includes the frequency range that was used for communication.

This experiment was made to determine the robustness of MCOK and CBOK against noisy channels and bad SNR. One can show in simulation that the matched filter transceiver design used in this work is able to do error-free communication with a SNR as bad as  $-2$  dB. Hence chirp based communication via aerial acoustic channel is expected to work reliably under the noise conditions of this experiment.

The experiment is conducted for CBOK transmission first. The error rate is measured for transmission without externally generated noise and then repeated for two different noise levels. The same experiment is then repeated with the volume of the transmitter set to 30 % of the original volume.

The measurements are also collected for a MCOK transmission.

### 4.5.1 Raw Data

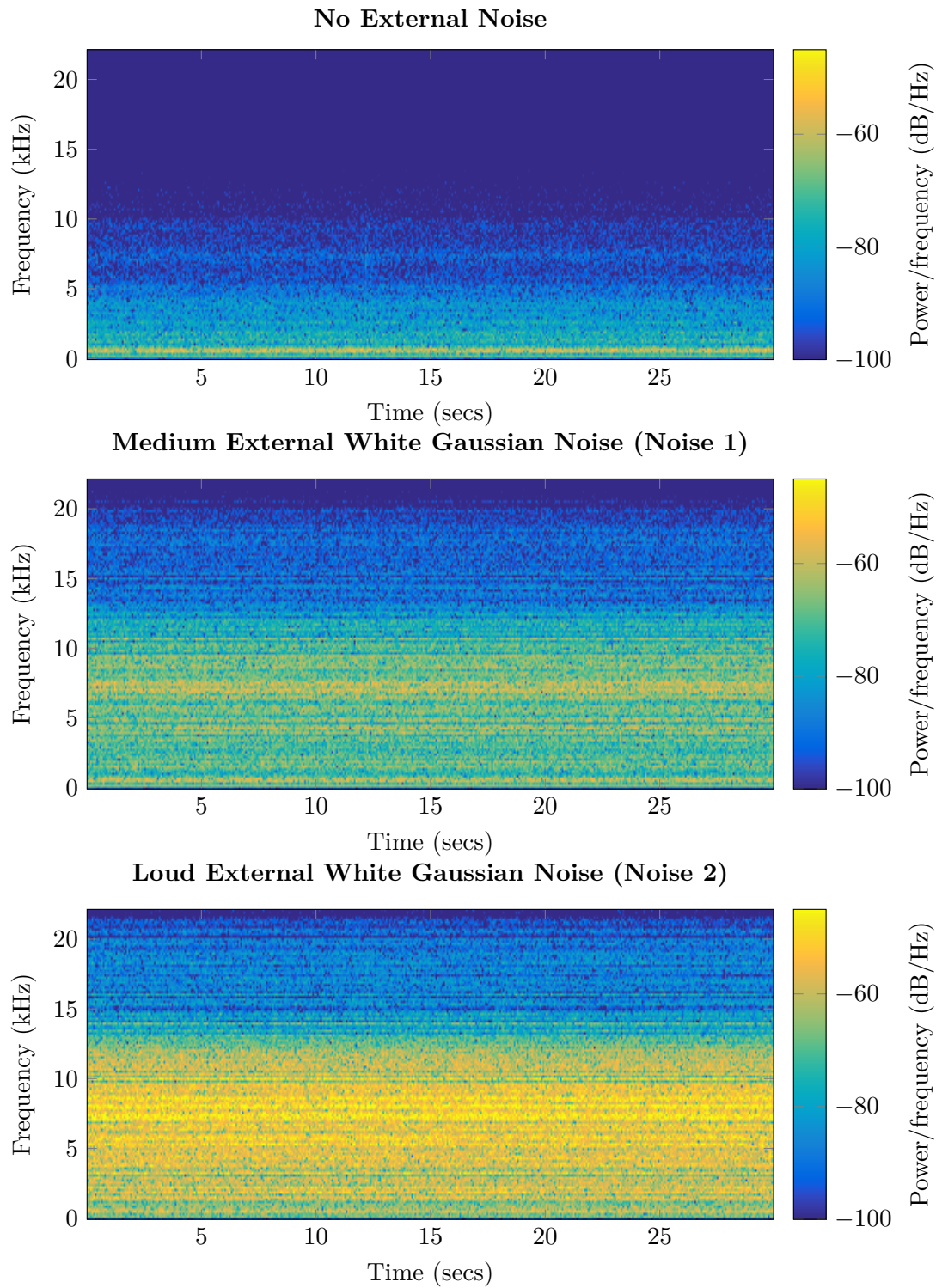
This experiment includes multiple data sets. Measurements of CBOK transmission under noise influence can be reviewed in table 4.5.1. Data for CBOK transmission with reduced volume can be taken from table 4.5.1. Results of experiments with MCOK transmission in a noisy environment are listed in table 4.5.1.

$f_1$ [Hz]	$f_2$ [Hz]	$T_{\text{Sym}}$ [ms]	$T_G$ [ms]	Noise	R [Bytes s <sup>-1</sup> ]	Correct Bytes	Total Bytes
2500	5000	5	1	1	14.13	212	220
					14.33	215	224
					15.07	226	232
2500	5000	5	1	2	12.73	191	192
					13.33	200	204
					12.87	193	196

**Table 4.5.1:** Measurement results for experiment 4. The noise type references the signals described in figure 4.5. This data shows, that CBOK is not affected much by external noise.

$f_1$ [Hz]	$f_2$ [Hz]	$T_{\text{Sym}}$ [ms]	$T_G$ [ms]	Noise	R [Bytes s <sup>-1</sup> ]	Correct Bytes	Total Bytes
2500	5000	5	1	0	12.8	192	192
					13.33	200	200
					13.67	205	208
2500	5000	5	1	1	0.33	5	15
					0	0	40
					0	0	24
2500	5000	5	1	2	0	0	0
					0	0	0
					0	0	0

**Table 4.5.2:** Measurement results for experiment 4b. In this experiment, the volume of the transmitter was reduced to 30 % of its original volume, while keeping the noise levels from the previous experiment.

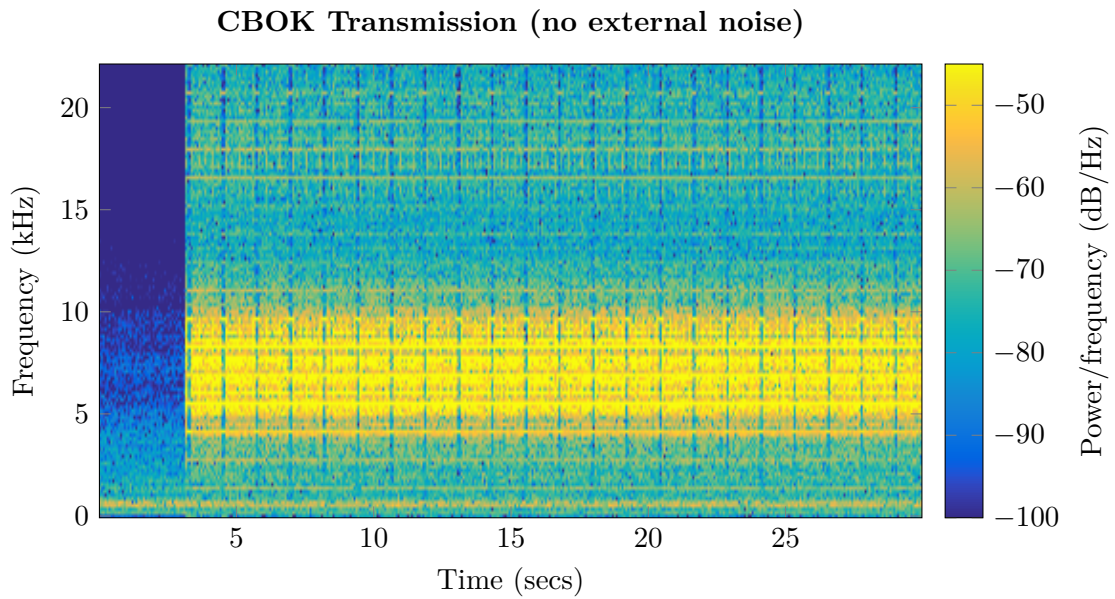


**Figure 4.5:** Spectrogram of the received noise. The noise generated by the robot's fans is clearly visible in the first subplot. The externally generated white noise is reasonably flat in the frequency range from 0.2 kHz to 10 kHz.

$f_1$ [Hz]	$f_2$ [Hz]	$f'_1$ [Hz]	$f'_2$ [Hz]	Rolloff	$T_{\text{Sym}}$ [ms]	$T_G$ [ms]	Noise Type	R [Bytes <sup>-1</sup> ]	Correct Bytes	Total Bytes
2500	5000	5000	7500	0.1	5	1	1	24.87	373	384
								15.6	234	312
								16	240	248
2500	5000	5000	7500	0.1	5	1	2	0	0	44
								3.93	59	137
								13.87	208	364
								15.8	237	380

**Table 4.5.3:** Measurement results of experiment 5. This data shows that the drops in error free data rate are higher for MCOK than for CBOK.



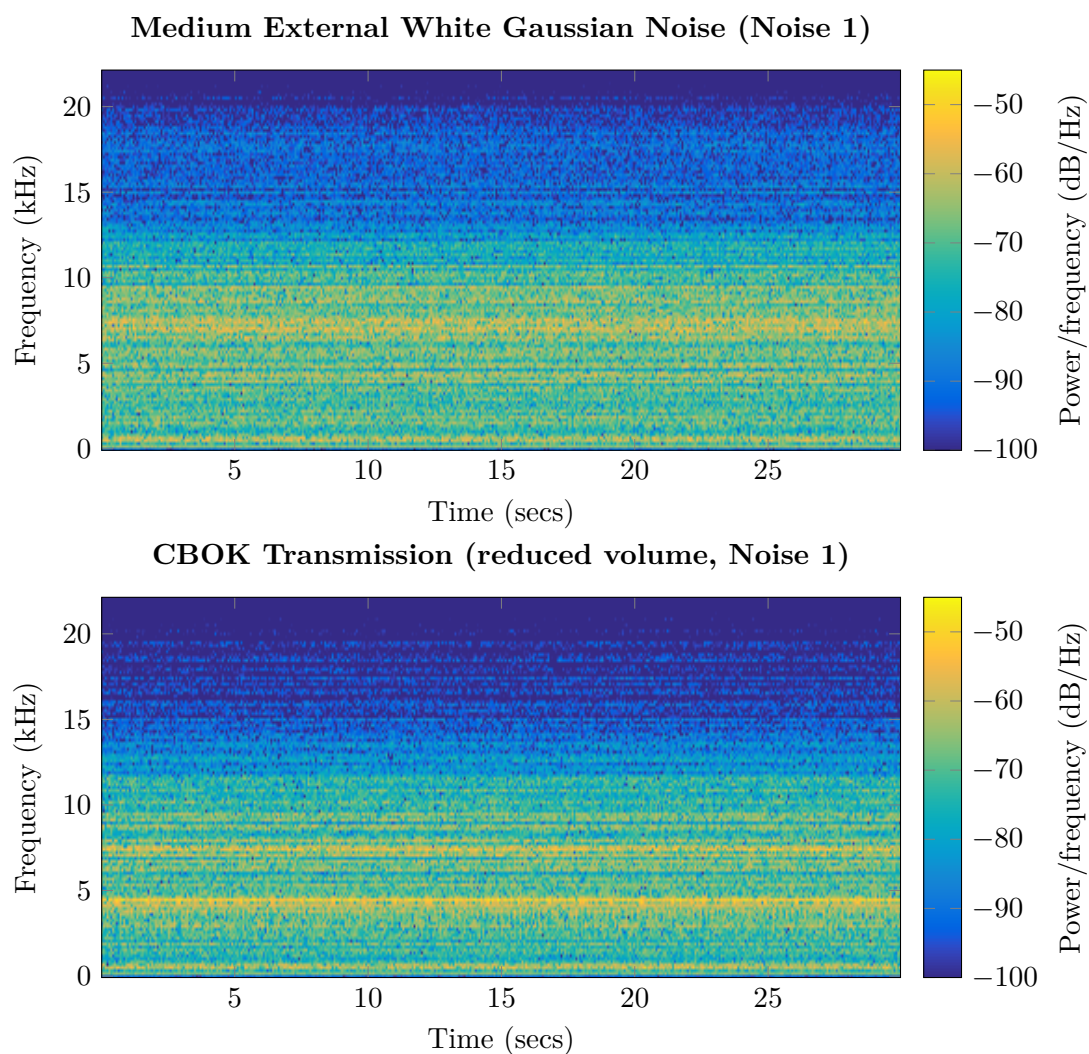


**Figure 4.6:** Spectrogram of a CBOK transmission with  $f_1 = 5$  kHz and  $f_2 = 7.5$  kHz. The transmission starts after 4 s of measurement. This transmission uses no pulse-shaping, the harmonics are observable quite clearly outside the transmission band. No external noise was generated in this measurement.

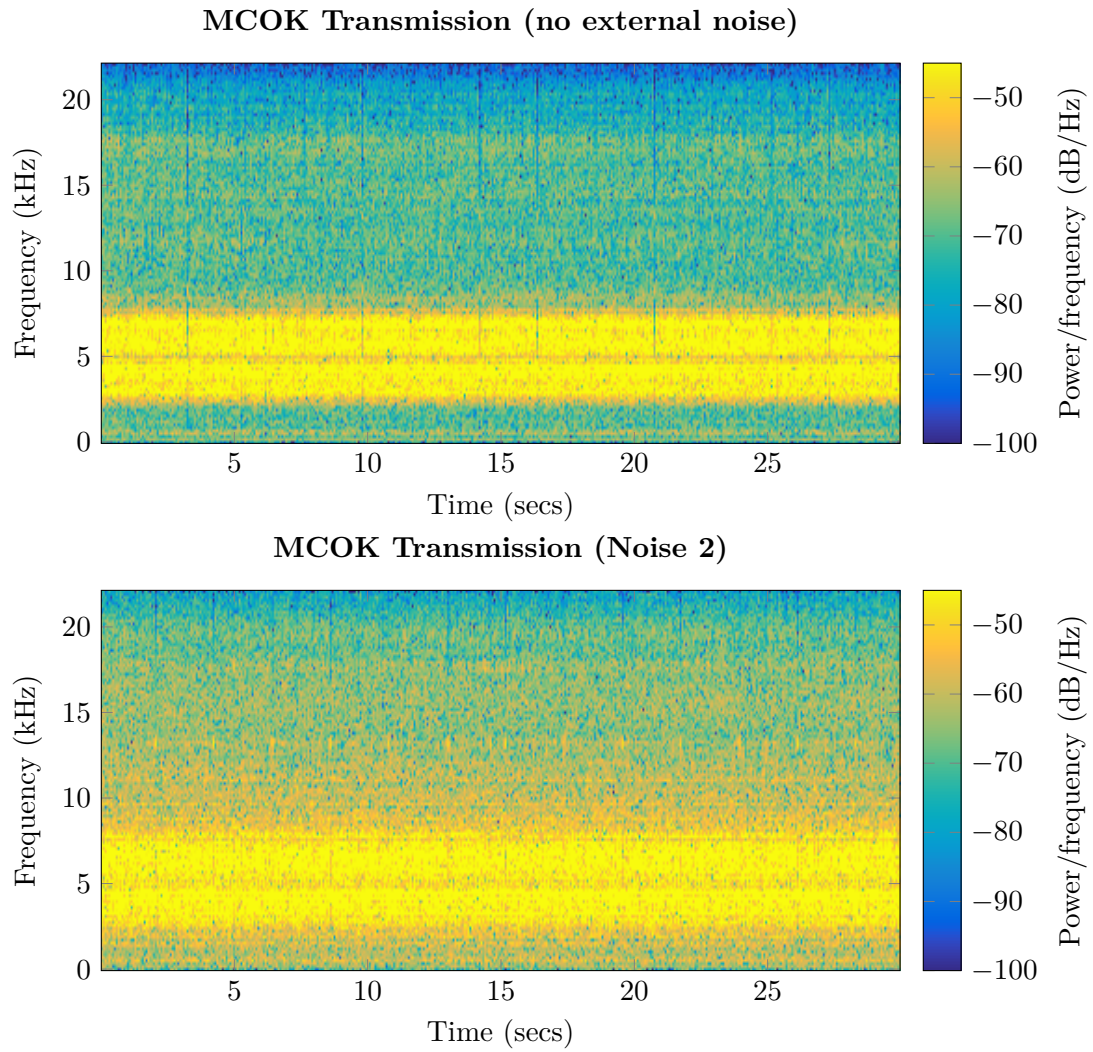
#### 4.5.2 Analysis

The experiments that were conducted to measure the noise influence can only yield qualitative results. This is because the generation of an exact SNR at the receiver is very complicated and only two different noise levels were used.

To get an intuitive grasp of the quality of the transmission channel under noise influence, spectrogram plots visualize the noise power vs. the signal power. A CBOK transmission without external noise influence can be reviewed in figure 4.6. The error rate for this transmission is as low as  $E_{\text{byte}} = 0$ , as shown in section 4.3. If the noise level is increased, generally the error rate increases as well.



**Figure 4.7:** Spectrogram of a CBOK transmission with reduced volume. In comparison to the noise spectrogram without any signal, no clear transmission is detectable in the range 2.5 kHz to 5 kHz. This leads to very high error rates.



**Figure 4.8:** Spectrograms of MCOK transmission with different external noise levels. The two bands for the different chirp symbols are distinguishable in the plot. Note that due to a roll-off factor of  $\beta = 0.1$ , the harmonics outside the transmission band vanished completely.



## Chapter 5

# Summary

### 5.1 Summary

This thesis presents the implementation of a method of communication between two robots that uses the aerial acoustic channel. In chapter 2, chirp-based acoustic communication is introduced and the multiple stages of the transmitter and the receiver are discussed. Especially the synchronization between transmitter and receiver is of high importance for an error free communication. Therefore envelope detection is used to filter high frequency side-peaks from correlation signals. An extension of the CBOK modulation scheme [1] that uses chirps of different frequencies for N-Ary modulation is proposed. This modulation scheme reaches transmission speeds of up to  $29 \text{ Bytes}^{-1}$  in LOS transmission, while the CBOK transmission reaches only up to  $22 \text{ Bytes}^{-1}$ . The transmission was tested under *RoboCup* conditions and proved to be functional in LOS situations covering a distance of more than 6.5 m.

The implementation of the chirp-based communication for the NAO robotics platform is outlined in chapter 3. To fulfill real-time constraints and reduce implementation time, FFTW and *PortAudio* libraries are used. In chapter 4 multiple experiments were presented that identify optimal frequency ranges and bandwidth limitations of CBOK and MCOK transmission. Also noise sensitivity is examined experimentally. Chirp-based communication is robust against noise levels where hearing protection was required in the laboratory and human conversation is no longer possible.

If the *RoboCup Technical Committee* decides that WiFi communication is no longer allowed in competitions, chirp-based communication could substitute it to broadcast some information to the robot players. This however would heavily disturb the fun of watching a robot soccer game, as the chirp sound is *very* intrusive for human listeners.

### 5.2 Outlook

In this work it is demonstrated that communication via aerial acoustic channel is feasible. This is mainly done by transmitting randomly generated data and counting the successfully transmitted bytes.

In a real world scenario, it is important that all data is received reliably and correctly. This could be ensured using different approaches: First, the overall byte error rate could be reduced by introducing forward error correction coding. Then it would be possible to minimize the number of synchronization errors by synchronizing with every individual symbol. To allow multiple robots to communicate with each other, a multiplexing technique needs to be implemented. As the number of robots could be as large as ten per game and the available bandwidth is limited with respect to the required bandwidth per robot, Frequency Division Multiple Access (FDMA) probably won't be applicable. However, each robot can listen (literally) on the channel easily. Future work should consider Time Division Multiple Access (TDMA) as an approach to allow multiple robots to broadcast information to each other.

A third field of research that should be considered, is a way to make acoustic communication less intrusive for human bystanders. As ultrasonic frequencies are not an option with the available hardware, the communication scheme must adapt to human hearing preferences.

Clearly, acoustic communication is inferior to wireless communication using electromagnetic waves and will be limited to very special use cases. For the sake of the ears of us all, let's hope *RoboCup* will not become such an use case any time soon.



# Bibliography

- [1] H. Lee, T. H. Kim, J. W. Choi, and S. Choi, “Chirp signal-based aerial acoustic communication for smart devices,” in *2015 IEEE Conference on Computer Communications (INFOCOM)*. IEEE, 2015, pp. 2407–2415.
- [2] I. F. Akyildiz, D. Pompili, and T. Melodia, “Challenges for efficient communication in underwater acoustic sensor networks,” *ACM Sigbed Review*, vol. 1, no. 2, pp. 3–8, 2004.
- [3] L. Freitag, M. Grund, S. Singh, J. Partan, P. Koski, and K. Ball, “The whoi micro-modem: an acoustic communications and navigation system for multiple platforms,” in *Proceedings of OCEANS 2005 MTS/IEEE*. IEEE, 2005, pp. 1086–1092.
- [4] C. V. Lopes and P. M. Aguiar, “Aerial acoustic communications,” in *Applications of Signal Processing to Audio and Acoustics, 2001 IEEE Workshop on the*. IEEE, 2001, pp. 219–222.
- [5] H. Kitano, *RoboCup-97: robot soccer world cup I*. Springer Science & Business Media, 1998, vol. 1395.
- [6] RoboCup Technical Committee, “Robocup standard platform league (NAO) rule book,” 2016. [Online]. Available: <http://www.informatik.uni-bremen.de/spl/pub/Website/Downloads/Rules2016.pdf>
- [7] —, “Robocup 2016 technical challenges,” 2016. [Online]. Available: <http://www.tzi.de/spl/pub/Website/Downloads/Challenges2016.pdf>
- [8] RoboCup Organizational Committee, “Robocup 2016 results,” 2016. [Online]. Available: <http://www.tzi.de/spl/bin/view/Website/Results2016>
- [9] A. Robotics, “Nao documentation v2.1,” 2016.
- [10] F. Bergmann, “Acoustic communication between two robots based on the nao robot system,” 2015.
- [11] L. Marple, “Computing the discrete-time “analytic” signal via fft,” *IEEE Transactions on Signal Processing*, vol. 47, no. 9, pp. 2600–2603, 1999.
- [12] M. Fowler, “Inversion of control containers and the dependency injection pattern,” 2004.
- [13] R. Bencina and P. Burk, “Portaudio—an open source cross platform audio api,” in *Proc. 2001 Intl. Computer Music Conf.(ICMC-01)*, 2001.
- [14] M. Frigo and S. G. Johnson, “Fftw: An adaptive software architecture for the fft,” in *Acoustics, Speech and Signal Processing, 1998. Proceedings of the 1998 IEEE International Conference on*, vol. 3. IEEE, 1998, pp. 1381–1384.

- [15] RoboEireann, “Robocup 2016 communications tester,” 2016. [Online]. Available: [https://github.com/roboeireann/no\\_wifi\\_challenge](https://github.com/roboeireann/no_wifi_challenge)