VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

MĚŘENÍ VZDÁLENOSTI MOBILNÍM TELEFONEM POMOCÍ ECHA

ECHO-BASED DISTANCE MEASUREMENT ON MOBILE PHONE

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — JIŘÍ RICHTER
AUTHOR

VEDOUCÍ PRÁCE — doc. Dr. Ing. JAN ČERNOCKÝ
SUPERVISOR

BRNO 2015

Abstrakt

Cílem této práce je vytvoření aplikace pro mobilní zařízení na měření vzdálenosti pomocí ozvěny. Zaměřuje se na testování komponent zařízení a výběr signálu s vhodnými parametry pro měření. První částí práce je vyhodnocení schopnosti zařízení přehrávat a nahrávat zvuk, výsledkem je určení frekvence a síly zvuku, se kterým je zařízení schopno pracovat. Práce dále popisuje algoritmus pro měření vzdálenosti založeny na vzájemné korelaci vyslaného a přijatého signálu. Další částí je popis implementace výše zmíněného algoritmu v aplikaci pro operačním systému Android s řešením komunikace a synchronizace mezi jednotlivými součástmi. Implementace navrženého algoritmu v aplikace je v závěru práce otestována. Na základě zjištěných údajů je možné měření vzdálenosti na mobilním zařízení od půl metru do přibližně tří metrů, s ohledem na okolní hluk, s přesností na desetinu metru.

Abstract

The aim of this work is to create an application for mobile device to measure distance using echo. It focuses on testing of device components and signal selection with appropriate parameters. In the first part of work — evaluation of device ability to play and record sound — the result is determination of sound frequency and strength with which the device is able to work. Next, the work describes the algorithm to measure distance based on using cross-correlation of transmitted and received signals. Finally, the implementation of aforementioned algorithm in application for Android operation system is presented, with solution for communication and synchronization between individual components of application. Implementation of designed algorithm in application is tested. Based on the data, we can conclude that it is possible to measure distance from half a meter to approximately three meters, based on noises, with a precision of tenth of a meter.

Klíčová slova

sonar, měření vzdálenosti, mobilní zařízení, vzájemná korelace, charakteristika reproduktoru a mikrofonu mobilního zařízení

Keywords

sonar, distance measurement, mobile device, cross-correlation, characteristics of loudspeaker and microphone of mobile device

Citace

Jiří Richter: Echo-Based Distance Measurement on Mobile Phone, bakalářská práce, Brno, FIT VUT v Brně, 2015

Echo-Based Distance Measurement on Mobile Phone

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval pod vedením doc. Dr. Ing. Jana Černockého. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

. . . .
Jiří Richter
May 20, 2015

Poděkování

Děkuji panu doc. Dr. Ing. Janu Černockému za vstřícné a odborné vedení, trpělivost a cenné rady při psaní mé bakalářské práce.

Contents

Chapter 1

Introduction

Using one universal device for different daily activities is slowly becoming true and such device is the smartphone. This thesis is focused on the enhancement of its usability. Every smartphone has speaker, microphone and high computing capacity. These three properties are needed to use smartphone as device to measure distance using echo. The task of this thesis is to create and test an application which turns the smartphone into your own pocket meter.

I chose the theme of this work because I am using the smartphone for everyday activities and measurement of distance is the function I miss. To be able to measure the size of room or distance to object can be very useful when a meter is not around. Most of people take the smartphone with them everywhere. The other reason for choosing this theme is to enhance the capabilities of smartphone and to find its limits when it comes to work with its audio components.

During survey for this thesis a few applications which try to add this functionality to smartphone were found. Some of them are giving multiple results and the user has to determine the correct one, the others had many settings with complicated control of them. The task of this thesis is to create an application with simple control providing the user with one, most correct, result.

Chapter 2

Sonar and cross-correlation of signals

To use smartphone as device to measure distance, the discrete signal processing is needed. In this chapter, at first, we describe the idea of using smartphone device as sonar to obtain an idea of how the process should work. Finally, the fundamentals of discrete signal processing are discussed.

2.1 Sonar

Sonar is an object-detection system that uses sound waves for detecting and locating objects, measurement distance or speed of objects. Term “Sonar” is derived from English words that describes its purpose: sound navigation and ranging. In this thesis, the meaning of sonar is simply a device which is able to measure distance to object in specific direction.

Functionally, the sonar has a transmitter that emits sound waves called sound signals in predetermined directions. When these come into contact with an object, they are reflected or scattered in many directions. Sonar signals are reflected especially well by materials such as metals, seawater and wet ground but even other materials can reflect the signals well. The sonar signals that are reflected back towards the transmitter are the ones that make the sonar work. To work, th sonar also need receivers. These are usually, but not always, in the same location as the transmitter. Reflected sonar signals captured by receiving antenna are usually very weak but using methods of signal processing, the useful sonar signal is obtained. The principle schema of sonar is shown in figure 2.1

The smartphone uses its own components. It has one transmitter and one receiver. The transmitter is the loudspeaker of the device and the receiver is its microphone. Mutual position can change from device to device. Knowing the position of loudspeaker on specific device, is important to be able to use it. The transmitter of sonar has to be pointed in direction in which we measure the distance to the object to amplify the strength of the reflected signal. On most devices, the speaker is located on back of the device. To obtain the reflected signal with as high strength as possible, the receivers of sonar are also pointed in direction of object. Smartphones have their microphones located in the bottom of device which leads to impossibility to have both transmitter and receiver pointed in direction of object. In figure 2.2, the usual locationd of loudspeaker and microphone of device are shown. The difference between sonar and smartphone is in its transmitter and receiver. Sonar’s transmitter is constructed to be able to create ultrasound waves. The smartphone

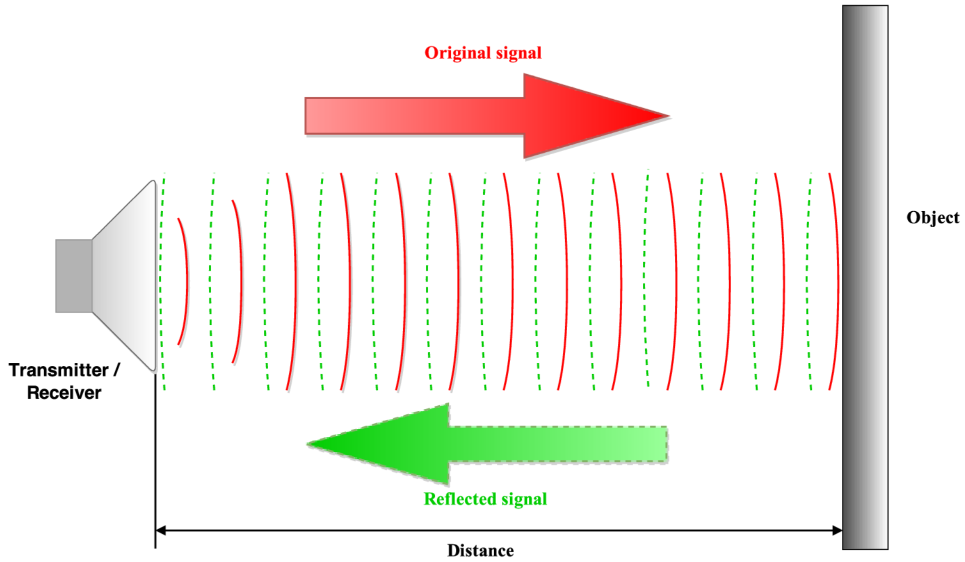


Figure 2.1: Principle of sonar.

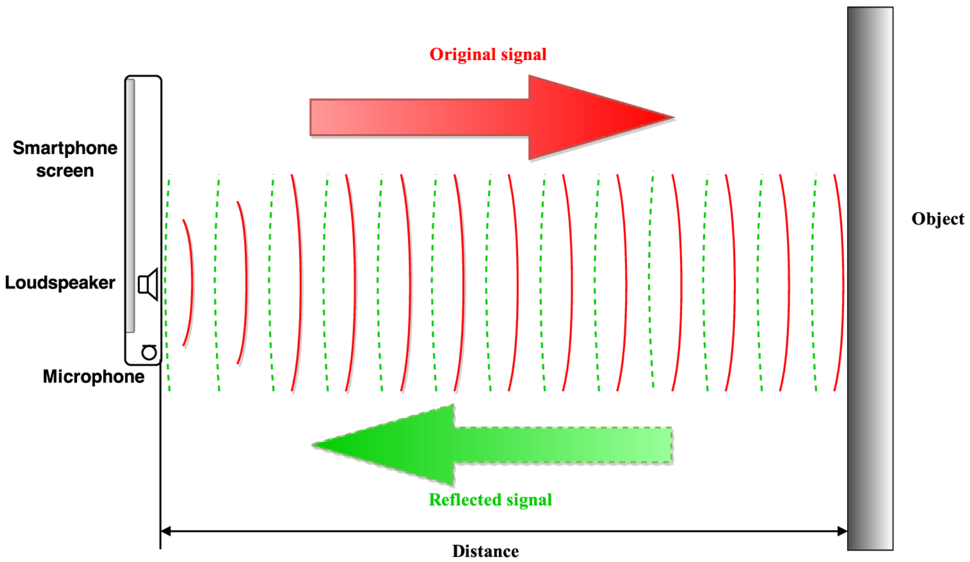


Figure 2.2: Smartphone used as sonar with usually placed components.

transmitter is constructed to create audible sound waves. Sound and ultrasound waves are vibrations that propagate as a mechanical waves of pressure through a medium such as air or water. The difference is that the frequency of sound wave is the same as frequency range of human hearing which is usually set from 20 Hz to 20 kHz, the ultrasound is using frequency ranging higher than 20 kHz. All sound waves propagate through the environment by speed of sound. As the use of ultrasound waves is impossible because of limitations of smartphone device, the rest of work is focused on signals audible sound waves.

The transit time method is used to measure distance with sonar, it is using time of flight of signal. The sonar transmits a short pulse of sound signal and measures the time it takes for reflection to get to the receiver. Then the distance is one half of product of round trip time (because the signal has to travel to the target and then back to the receiver) and the speed of signal. As the receiver is unable to detect the reflection while the signal is being transmitted, because the receiver recording strong signal directly from transmitter, the sonar has to stop transmitting before the reflected signal returns. This leads to limitations for range of possible distances to measure. If signal is short and has less total energy, sonar is able to measure short distances, but such signal is more likely to be lost in noise. In opposite case of longer signal with higher total energy, the sonar is unable to measure short distances because of higher noise caused by higher total energy and more reflections. In order to measure short and long distances, switching of different length signals with different power levels and silences is used. If only one signal is used, it is compromise with middle strength and length.

2.2 Digital signal processing

Through the air the continuous-time signal is distributed, but to be able to save it and process it in digital device, it has to be changed into digital signal. This is called analog to digital conversion and it is consist of two steps. The first step is to convert the continuous-time signal into discrete signal. Discrete signal is time series consisting of a sequence of quantities. It is obtained from continuous-time signal using sampling, then each value in the sequence is called sample. Digital signal is a discrete signal for which not only the time domain but also amplitude domain has been made discrete. Each sample can only have values from a discrete set. If that discrete set is finite, the digital signal can be processed by digital device.

Functionality of sonar is based on processing of digital signals. Signals are sent by sonar transmitter into specific direction where they are deformed and reflected to receiver. Recorded signal with receiver contains significant amount of noise and weak signal reflected from object. To recognize weak reflected original signal, sonar is using electronic amplifiers, filters and methods to determine similarity.

2.2.1 Cross-correlation

It is necessary to be able to establish the similarity between two sets of data. In other words, the correlation between the data is sought. Correlation can be defined mathematically and can be quantified. If the two digital signals, x_1 and x_2 are similar, then measure of their correlation $xcorr_{x_1,x_2}$ might be obtained using

$$xcorr_{x_1,x_2} = \sum_{n=1}^{N} x_1(n)x_2(n). \tag{2.1}$$

When the case of two independent and random data sequence is considered, the sums of products will tend towards a vanishingly small random number as the number of pairs of points is increased. This is because all numbers, positive and negative are equally likely to occur so that the product pairs tend to be self-cancelling on summation (see 2.5). By contrast, the existence of a finite sum will indicate a degree of correlation (see 2.3). A negative sum will indicate a negative correlation (see 2.4), it is an increase in one variable is associated with a decrease in the other variable. Cross-correlation produces a result which

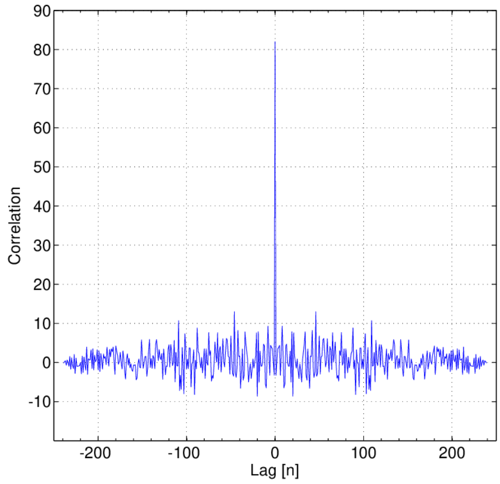


Figure 2.3: Positive auto-correlation.

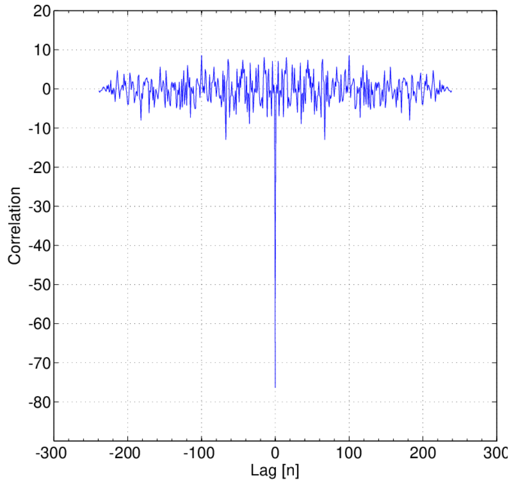


Figure 2.4: Negative auto-correlation.

depends on the number of sampling points taken. This can be corrected by normalizing the result to the number of points N by dividing it by N. As two correlated signals are often out of phase, this occur for example when x_1 is the original signal while x_2 is the reflected one recorded by sonars receiver with delay. To overcome such phase differences, it is necessary to shift, or lag, one wave form with respect to the other. Typically x_2 is shifted to the left to align the waveforms prior to correlation. The formula for the cross-correlation thus becomes:

$$xcorr_{x_1,x_2}(j) = \sum_{n=1}^{N} x_1(n)x_2(n+j) \tag{2.2}$$

Where the j represents the amount of lag which is the number of sampling points by which x_2 has been shifted to left. As in practice the phase relationship of x_1 and x_2 is not known, the correlation is computed for a number of different lags in order to find the largest value of the correlation which is then taken to be the maximum.

If the formula 2.2 is used to calculate the cross-correlation, the complexity of algorithm is order N^2. So for signal of size 16 samples, we need to calculate 256 values. Real digital signals are composed from hundreds or thousands of samples, this brute force algorithm is inapplicable. Therefore, the more efficient way to calculate cross-correlation using fast Fourier transform. The complexity algorithm is lowered to order $NlogN$, it means that for 16 samples size signal, 64 values is computed.

2.2.2 Cross-correlation using fast Fourier transform

Fast Fourier transform is an algorithm to compute discrete Fourier transform and inverse of it. Discrete Fourier transform converts a finite list of samples of a signal into the list

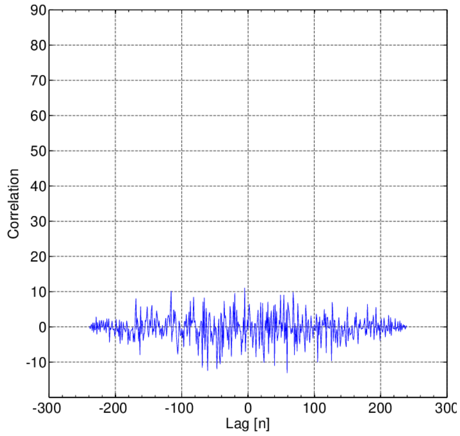


Figure 2.5: Cross-correlation of two independent and random signals.

of coefficients of complex sinusoids, ordered by frequencies. It can be said, that Fourier analysis converts the sampled function from its domain to the frequency domain. Using fast Fourier transform (FFT) and its inverse, the cross-correlation can be computed by formula 2.3 where $ifft$ is inverse FFT, fft is FFT, x_1 and x_{2rev} are signal for which correlation is calculated. x_{2rev} is reversed signal x_2.

$$xcorr_{x_1,x_2} = ifft(fft(x_1). * fft(x_{2rev})) \tag{2.3}$$

Chapter 3

Mobile device and measurement of its characteristics

Smartphone is a mobile device these parts has its characteristics, most important for this work are parts which are working with sound waves. In the next part the measurement of characteristic of these parts is performed. The device used for measurement is a representative smartphone from low end category LG P500.

Smartphone is using one of the most common operating system Android. This is UNIX based system with applications programmed in Java language. In Android, there are several options how to access components as speaker and microphone. There is number of defined classes to play and record sound waves. The differences between them are the possibility of modification and level of management. The most suitable ones for our purposes are *MediaPlayer* class to play sound and *AudioRecord* class to control recording. The characteristics is measured using the following equipment:

- Dictaphone M-audio Micro Track II SAP: 1000198462
- Measuring microphone Behringer ECM 8000
- Mobile device LG P500
- Speaker Redstar RS-501A

3.1 Application for playing and recording sound

To measure the characteristic of devices component a simple application was created. This application is able to play and record the sound after the user press one of the two buttons. This application is called *SoundTesting* and UI of it is plotted in 3.1. When the *start playing* button is pressed, the instance of *MediaPlayer* class is launched in new thread. The instance of *MediaPlayer* class is created function “create”. Once the instance is created the function *setOnCompletitionListener* is set. This function is called when the playing of sound is finished and it contains functions “stop” and “release” which have to be called on instance of media player in this order to release the resources. When the media player finishes the playing of sound set by radio button, the separate thread is terminated.

When the *start recording* button is pressed, new thread — for recording 5,10 or 15 seconds of sound — is created, time of recording is chosen by radio buttons. *AudioRecord* class is used for recording, this class enable to set source of recording, sampling frequency

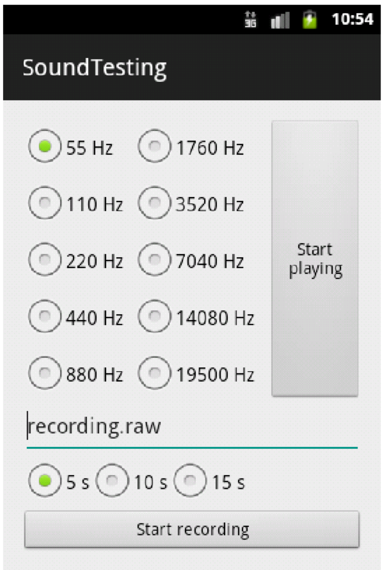


Figure 3.1: User interface of application.

and audio format. Signal is recorded into buffer and before the thread is terminated, it is saved into file with name “recording.raw”. The user can manually change the name of file.

The *AudioRecord* class has to have set the sampling frequency, audio format and minimum sized buffer to store the data. These are set as follows:

- Sampling frequency: 48 000 Hz
- Audio format: PCM 16 bit, MONO
- Minimum size buffer: calculated using function of *AudioRecord* class “getMinBufferSize”. The sampling frequency and audio format is needed. Minimum size buffer is than multiplied by value N_n to be able to store n seconds of sound. N_n is integer obtained by round up the result of formula:

$$N_n \doteq \frac{t \cdot F_s}{N_{min}}, \tag{3.1}$$

where F_s is sampling frequency of recording in Hertz and t is length of sound wave is seconds.

3.2 Measurement of characteristic of loudspeaker

The characteristic of loudspeaker has two parts, the first part is frequency characteristic and it is used to determinate optimal frequency for sound waves. The second part is focused on optimal value of device volume. Correct frequency range and signal strength can highly affect the ability of device to measure distance.

3.2.1 Frequency range

Theoretically, the loudspeaker of smartphone supposed to be able to play frequency range which the human ears are capable to hear. It is commonly given from 20 Hz to 20 000 Hz.

The range is different for every person and for most people, the upper limit decline over age. We expect that the frequency range of speaker is smaller.

Sine waves of different frequencies with length of two seconds are chosen for testing, frequencies are set in full range of human hearing: 55 Hz, 110 Hz, 220 Hz, 440 Hz, 880 Hz, 1 760 Hz, 3 520 Hz, 7 040 Hz, 14 080 Hz, 19 500 Hz. Example of sine wave with frequency 3 520 Hz is plotted in figure 3.2. These sine waves are separately played on mobile device and recorded by external microphone connected to Dictaphone. To guarantee quality of recorded signal, the sampling frequency is set to 96 kHz, encoding is chosen as PCM and resolution is 16 bit. Process of recording of sine waves is the following:

1. Take sine wave with lowest frequency 55 Hz.

2. Start recording on dictaphone.

3. Play actual sine wave on mobile device using *SoundTesting* application.

4. Stop recording on dictaphone, recorded signal is automatically saved to file.

5. If all sine waves were not played, take the one with first higher frequency than actual and go to step 2.

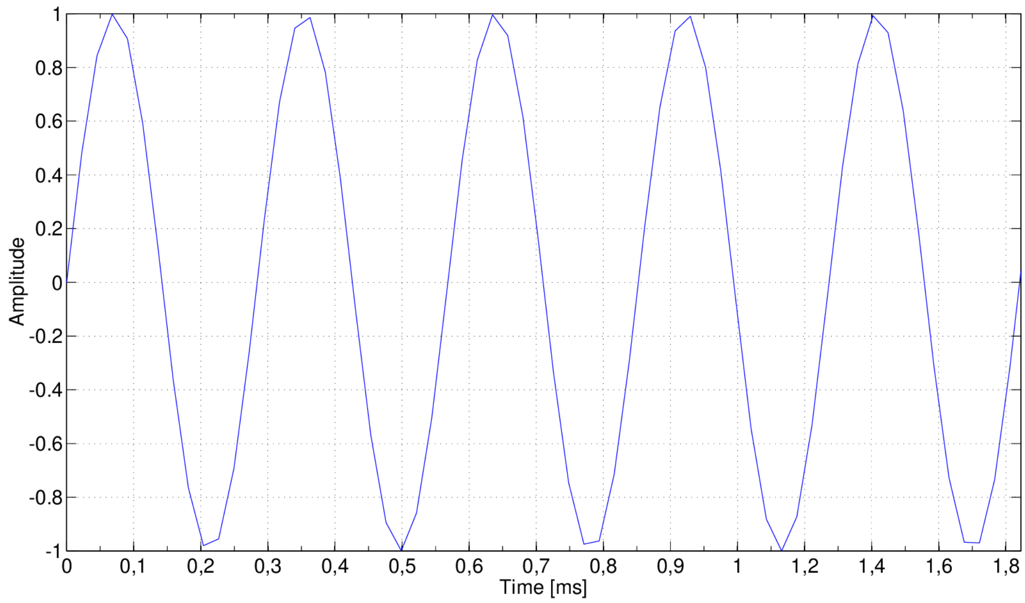


Figure 3.2: Original sine wave of frequency 3520Hz.

All recorded sine waves from dictaphone are loaded into Matlab and their analysis is performed. The analysis has two steps. In the first step, the sine wave is plotted and in the second step, spectral analysis is performed. Recorded sine wave of frequency 3520 Hz is plotted in figure 3.3. It is inspected visually if it corresponds to the original wave. Then, the spectral analysis of recorded sine wave is generated in Matlab. Spectral analysis shows us representation of different frequencies in signal, because sine wave of one frequency is used for playing and recording, it should dominantly contain only one frequency. Spectral analysis of sine wave 3 520 Hz is plotted in figure 3.4.

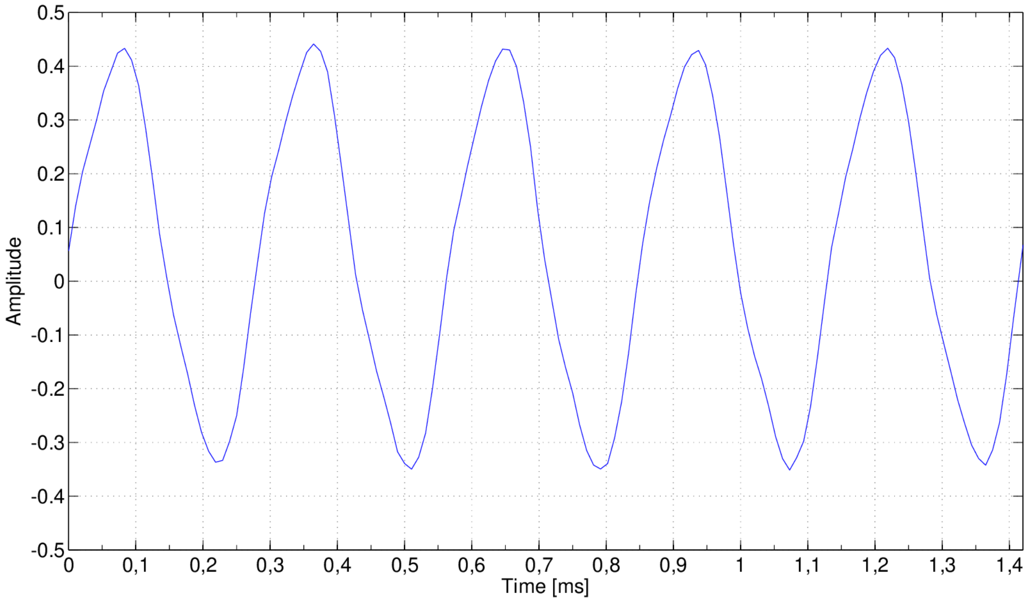


Figure 3.3: Recorded sine wave of frequency 3520Hz played by device.

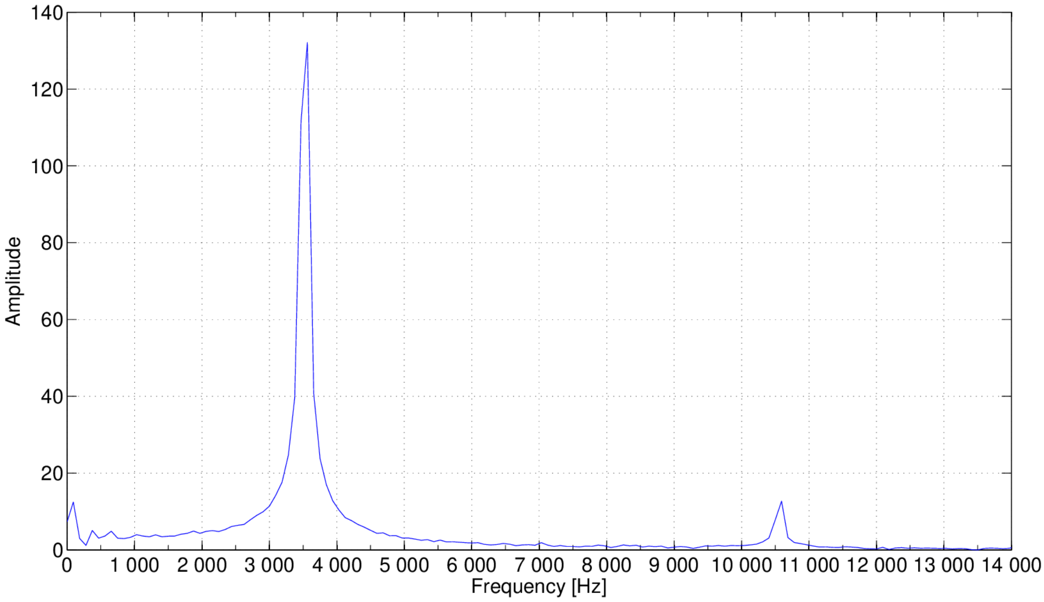


Figure 3.4: Spectrum of sine wave of frequency 3520Hz played by device.

Spectral analysis in figure 3.4 shows us that the recorded signal mostly contains frequency 3 520 Hz, other frequency highly represented in signal is 10 650 Hz, this is harmonic frequency three times bigger then fundamental frequency. Harmonic frequencies are present in signal because of its differences from pure sine wave. The frequency of sine wave from mobile device loudspeaker is slightly bigger than the original frequency but it is still able to play it cleanly. The opposite case is plotted in figures 3.5 and 3.6 where the analysis of sine wave with frequency 220 Hz is performed. The recorded sine wave is not visually corresponding to the original and from the spectral analysis, we can see, that signal contains only small amount of higher frequency, approximately 300 Hz. Sine waves of all testing frequencies and their spectras are attached in appendix A. As result of this measurement,

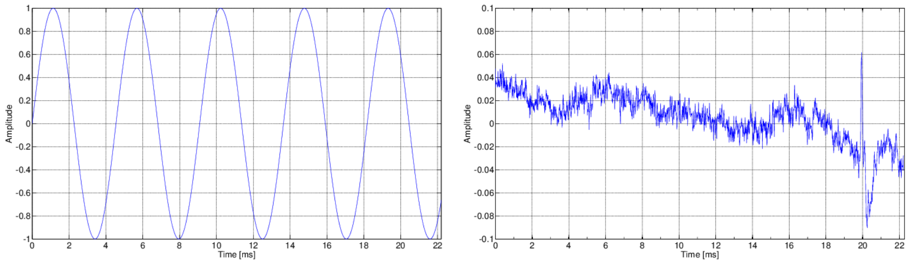


Figure 3.5: Original (left) and recorded (right) sine wave of frequency 220 Hz.

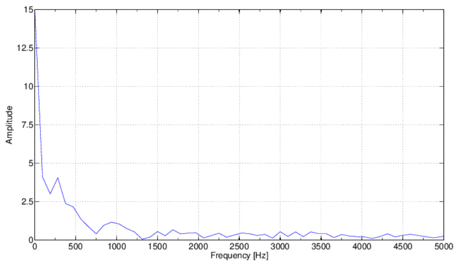


Figure 3.6: Spectrum of sine wave 220 Hz recorded by device.

we found that loudspeaker is able to play cleanly frequencies from approximately 1 760 Hz to 14 080 Hz. Because the frequency range of every loudspeaker can change a little and also the deformation of signal is bigger the closer we are to the limits of this range, for measurement, limited range 2 000 Hz - 7 500 Hz is chosen. Other characteristics are measured only for this frequency range.

3.2.2 Strength of signal

Next, we try to determinate optimal volume settings of device for measurement. Optimal value of volume is such value with which the speaker of device is able to play the signal without over modulation and it is strong enough to measure longer distances. With volume settings of device the strength of signal is easily manipulated, devices with Android OS have discrete values for it. It is value from 0 to 20 with a step of size 1.

For finding optimal value of volume, the sine waves of frequency range from 2 000 Hz to 7 000 Hz with step of the 500 Hz and maximal amplitude 1 has been chosen. All of them have length of two seconds. Process of speaker amplitude testing was the following:

1. Set maximum volume on Mobile device.

2. Start recording on external Dictaphone.
3. Play signal containing two seconds of each of testing frequencies.
4. Stop recording on Dictaphone and save recorded sound in file.
5. Plot the recorded signal from point 4 in Matlab and inspect it visually. If over modulation appeared on recorded signal, continue with step 6. Otherwise continue with step 7.
6. Lower volume of device by one and go to step 2.
7. Set actual volume as optimal device volume for measurement.

Signal with overmodulation is plotted in figure 3.7, maximum volume 20 is used to obtain this signal. The signal played using optimal volume of value 17 is plotted in figure 3.8.

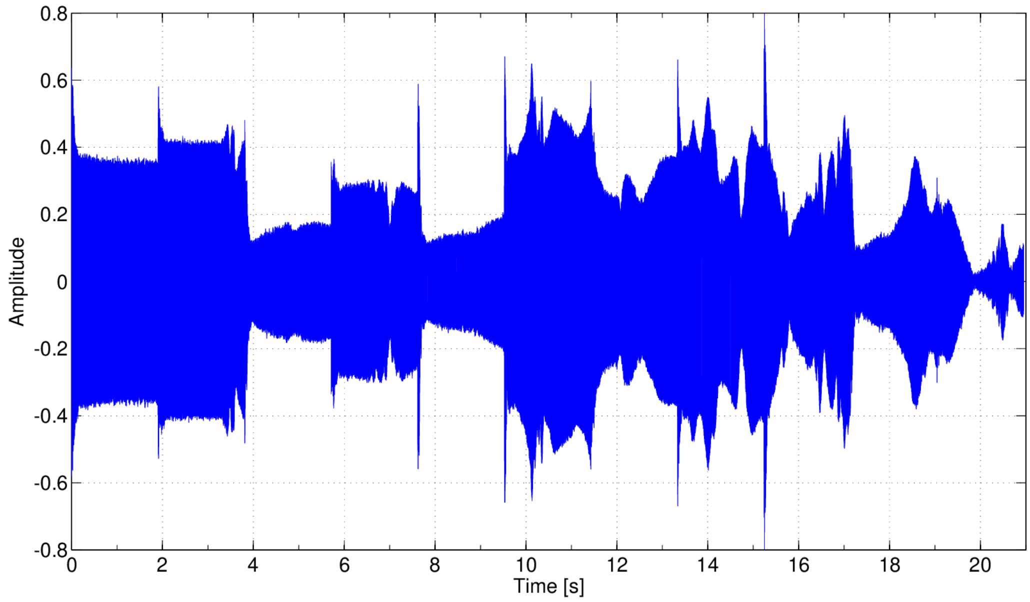


Figure 3.7: Recorded sine waves from 2 000 Hz to 7 000 Hz with step of 500 using maximum volume.

From figure 3.7 and 3.8 it can be seen, that the length of every wave is little shorter than original, this is caused by playing them on mobile device which cuts of a few milliseconds from each. For other description, we consider the length of every sine wave 1.9 seconds. The lower frequencies sine waves from 0-4 seconds are well played, when over modulation appears. But with the higher frequency, the amplitude of signal became unstable. On the other hand the sine waves played with optimal volume has stable amplitude for each frequency. Nevertheless the amplitude is stable, from figure 3.8 is seen, different strengths of signal for different frequencies. Sine wave of frequency 4 000 Hz is played with the highest strength, the opposite, the sine wave with lowest strength is the one with frequency 5 500 Hz. When this is compared to over modulated signal, the strongest signal has been automatically suppressed by the device to not cause the damage to loudspeaker, for low strength signal, we expect that it is well played, but the amplitude of this signal is very unstable. The different strengths of sine waves of different frequencies are caused by the components of the device. These are sound card and loud speaker.

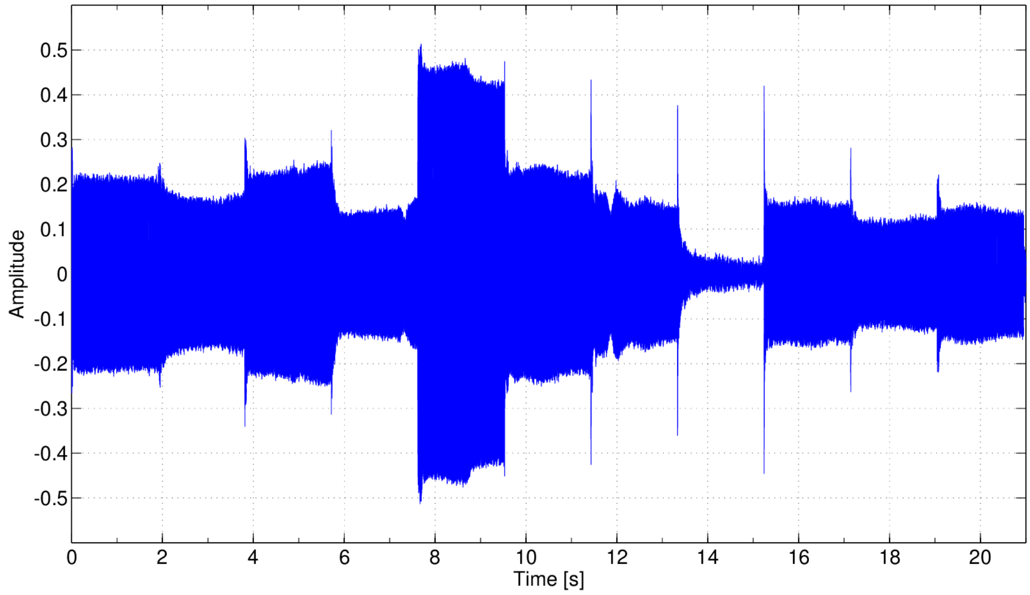


Figure 3.8: Recorded sine waves from 2 000 Hz to 7 000 Hz with step of 500 Hz using optimal volume.

3.3 measurement of characteristic of microphone

This section verifies if microphone of mobile device is able to record the sound of the same parameters as the loudspeaker is able to play. For that reason the characteristics of microphone is measured only for frequency range from 1 760 Hz to 14 080 Hz. The sound used for testing are sine waves: 1 760 Hz, 3 520 Hz, 7 040 Hz, and 14 080 Hz.

The process of looking for signal is the following:

1. Take sine wave with the lowest frequency 1 760 Hz.
2. Write the string to *SoundTesting* application, this string is used as name for file where the recorded is stored.
3. Start recording of five seconds on mobile device using *SoundTesting* application.
4. Play the actual sine wave on quality speaker.
5. When the playing and recording are finished, recording is saved to file with name from step 2.
6. If all sine waves were not played, take the one with the first higher frequency than the actual and go to step 2.

All recorded sine waves from *SoundTesting* application are loaded into Matlab and the same analysis as for the speaker is performed. Recorded sine wave of frequency 3 520 Hz is plotted in figure 3.9, original sine wave can be seen in figure 3.2, and it is visually inspected if it corresponds to the original wave. Its spectral analysis is plotted in figure 3.10. The shape of sine wave recorded by device visually corresponds to the original. From spectral analysis in figure 3.10, it can be seen that the recorded signal contains only frequency approximately 3 520 Hz. As result of this measurement, we found that the microphone

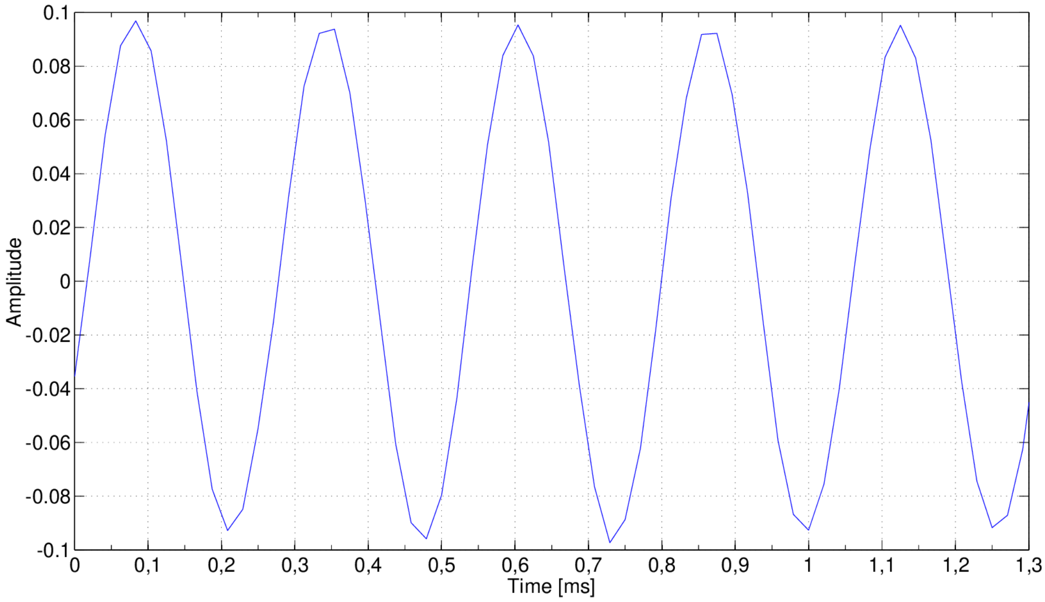


Figure 3.9: Sine wave of frequency 3 520Hz recorded by device.

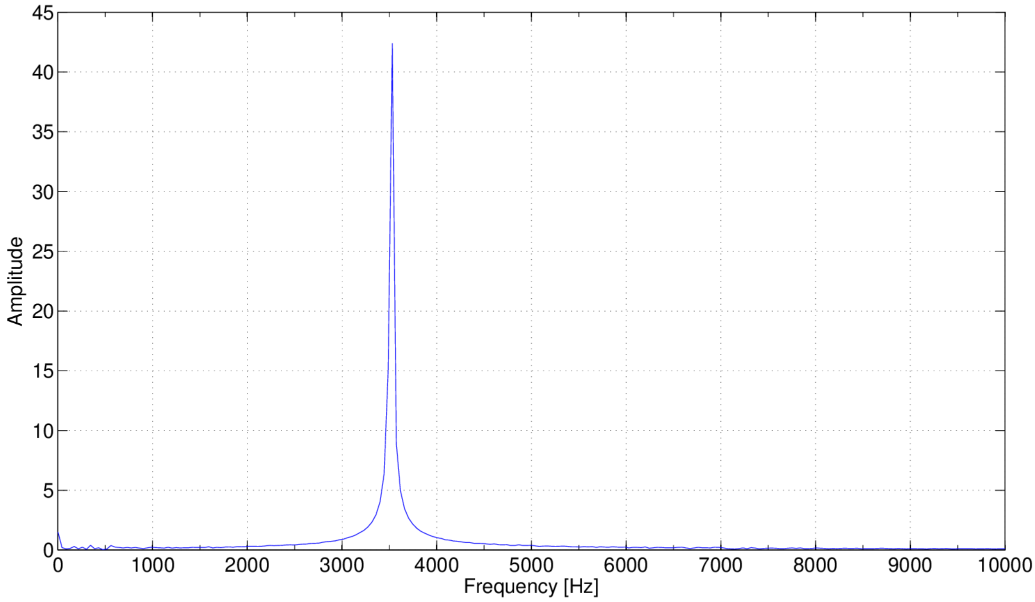


Figure 3.10: Spectrum of sine wave of frequency 3 520 Hz recorded by device.

of device is able to record cleanly all frequencies which the device is able to play. The frequency range for measurement from 2 000 Hz to 7 500 Hz remain unchanged.

Chapter 4

Distance measurement algorithm

In this chapter, the algorithm for distance measurement is described. Before creation of the algorithm we have to determinate shape of sound waves. We already found the frequency range and strength which are limited by device components characteristic. Its shape highly affects the usability of cross-correlation which is used for detection of sound wave.

4.1 Signals for measurement

The optimal shape of signal is chosen from four candidates which are cosine wave, chirp, white noise and filtered white noise. These were chosen because they represent simple signals with good usability for cross-correlation. Hann window is used for all of candidates to limit their duration and smooth the edges. If a signal contains fast transition of amplitude such as from value zero to value one, the speaker is unable to follow these changes and playes the signal inaccurately. Often, the clicking noise is heard. Hann window is plotted in figure 4.1. The highest frequency the signals contain is 7 500 Hz (except for white noise, see 4.1.4), to meet the sampling theorem and devices abilities to play signal, the sampling frequency 48 kHz is chosen.

4.1.1 Repetition of signal

If the measurement is performed only once there is possibility that the reflection is not perfect and it is lost. For better performance, the measurement has to be repeated. When the repeated measurements are combined the measurement is enhanced in two ways: the first one is that non-perfect reflections are suppressed. The second one is that the noise recorded by the device during measurement is attenuated, this is caused by the fact that amplitude and frequency of noise is different for every measurement, which leads to reducing the effect of it when they are summed. To easily repeat measurement, the final signal is composed of parts of active signals and parts of silences. These two parts alternate. The number of alternations indicates the number of measurements, each active part generates one. An example of final signal with five repetition and shape of cosine wave is plotted in figure 4.2. For cosine wave and chirp shapes, the active part for each repetition is identical. The reason for this is that these signals have parameters that are always the same. On the other hand the white noise is different every time it is generated, therefore the active part of every repetition is also different. The example of white noise wave with five repetitions is plotted in figure 4.3. While the measuring signal is played and recorded, the device has to be at the same stable position. Longer signal can contribute positively to the measurement

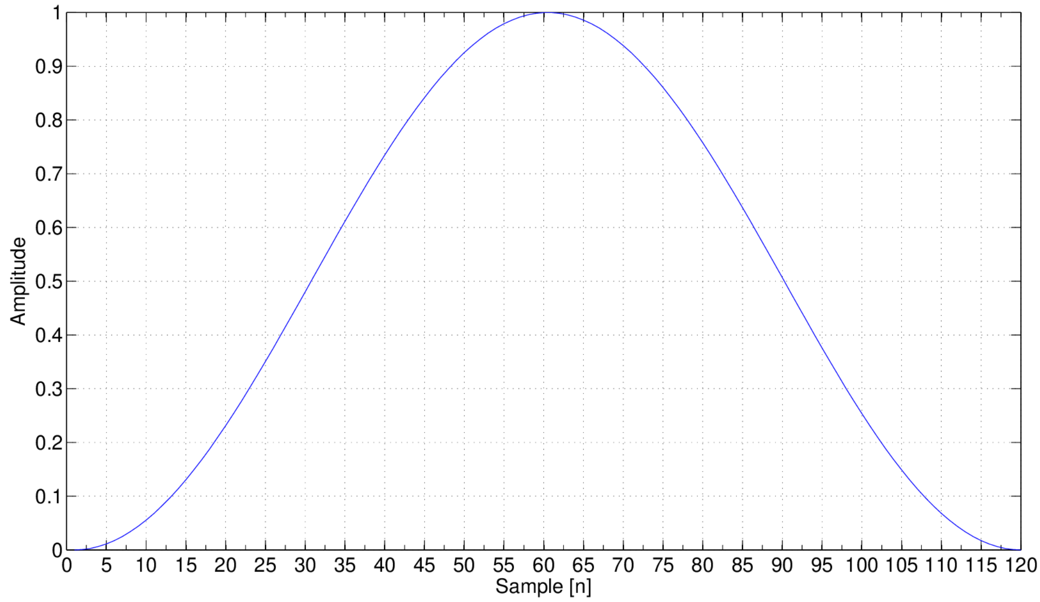


Figure 4.1: 120 points Hann window.

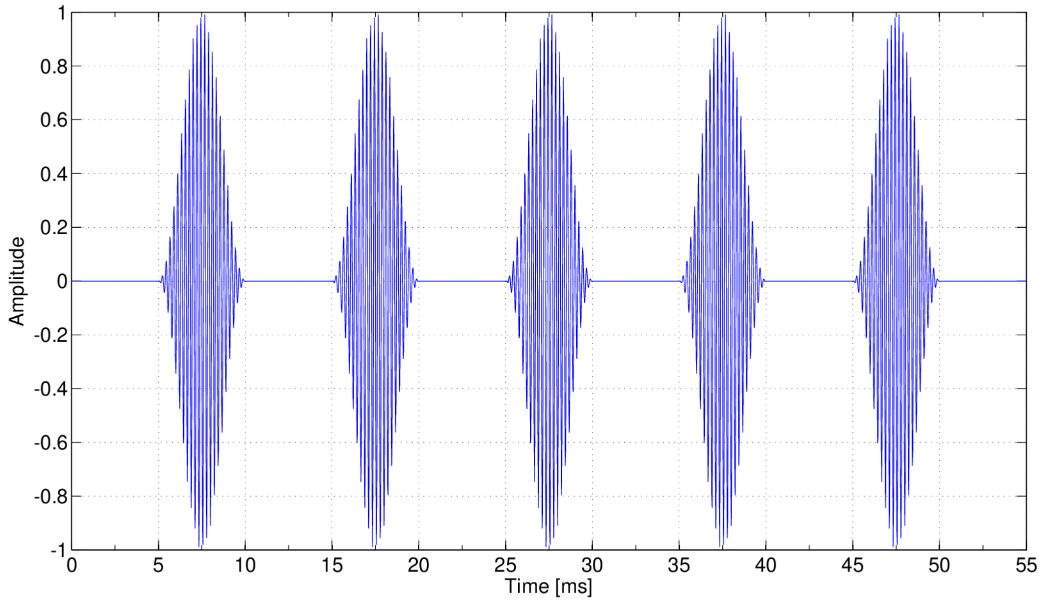


Figure 4.2: Cosine wave 4 500 Hz with repetiton.

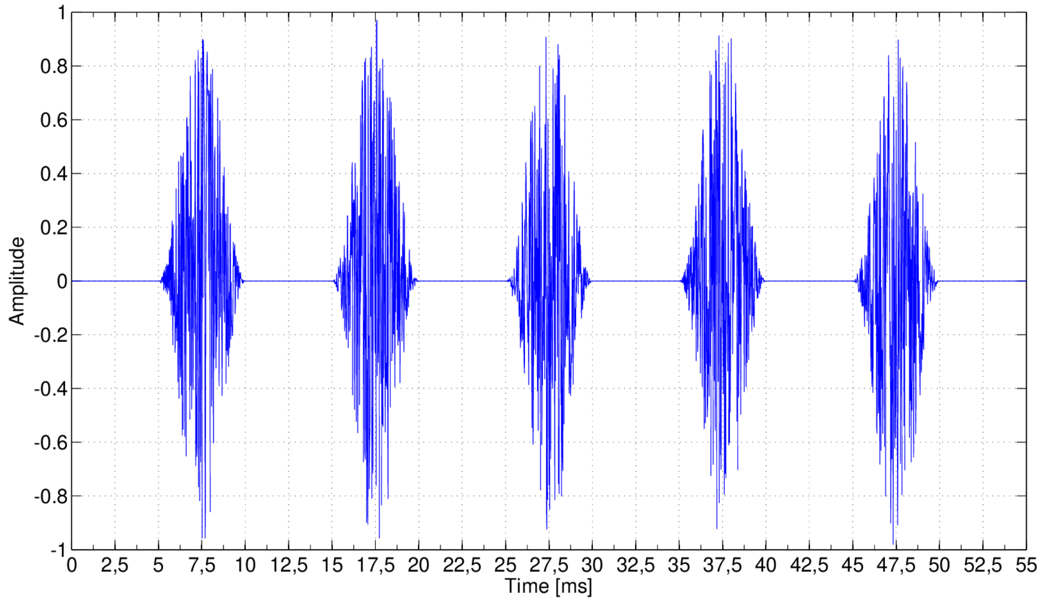


Figure 4.3: White noise with repetition.

but it also leads to the need to hold the device at the same position for long time. To choose compromise between usability for user and ability to measure distance precisely, the length of signal is chosen to approximately one and half second. The length of active signal l_{active} is chosen five millisecond and length of silent l_{silent} is chosen twenty milliseconds to fit forty repetitions in one second long signal. For calculation of length of signal we used

$$l_N = l_{silence200} + Nl_{active} + Nl_{silence} + l_{silence200}, \tag{4.1}$$

where N is number of repetition and length of signals is in milliseconds, 200 milliseconds of silence $l_{silence200}$ is also added to start and end of signal. The silence between measurements is inserted to easily find reflections of signal in recording. This leads to restriction of possible measured distance. The minimum distance the application is theoretically able to measure is approximately 0.45 meters, for calculation we used

$$d_{min} = \frac{\frac{l_{active}}{2} c_{sound}}{2}, d_{min} = \frac{\frac{0.005}{2} 343.46}{2}, d_{min} = 0.4293m, \tag{4.2}$$

where l_{active} is duration of active signal and c_{sound} is speed of sound. If the distance is shorter, the reflected sound is lost in transmitting signal, which is considerably stronger. Maximum theoretical distance limited by source signals is approximately 3.5 meters, for calculation we used

$$d_{max} = \frac{l_{silence} c_{sound}}{2}, d_{max} = \frac{0.02 \cdot 343.46}{2}, d_{max} = 3.4346m, \tag{4.3}$$

where $l_{silence}$ is duration of silence and c_{sound} is speed of sound. If the distance is longer, the reflected sound is lost in next transmitting signal. These limits are verified in chapter 6.

4.1.2 Cosine wave

Cosine wave is chosen as the first candidate. Five milliseconds of signal used for testing are plotted in figure 4.4. Autocorrelation of this cosine wave is plotted in figure 4.5. Because

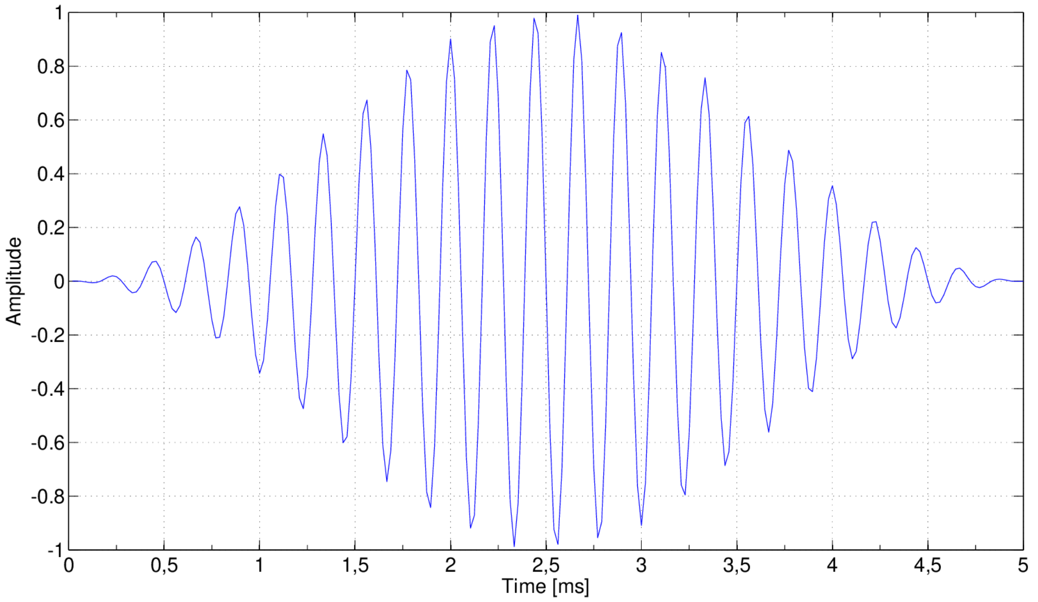


Figure 4.4: Five milliseconds of cosine wave windowed by Hann window.

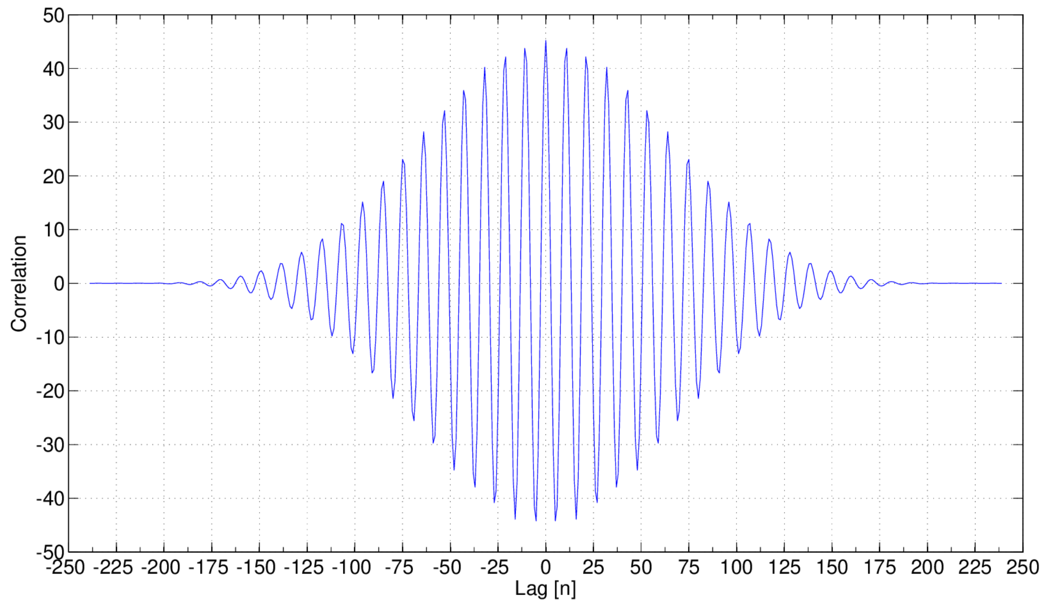


Figure 4.5: Autocorrelation of cosine wave from 4.4.

cosine wave is periodical, its cross correlation has large side lobes. These lobes complicate to the location of the coorrect in cross-correlation because there will be high number of false maxima. This leads to decision that cosine wave is a not good signal for measurement.

4.1.3 Chirp

As the next candidate a chirp wave is chosen. Chirp is cosine wave shaped signal with growing frequency over time. Frequency range is chosen accordingly to chapter 3 from 2 000 Hz to 7 500. A five milliseconds long chirp wave is plotted in figure 4.6. Autocorrelation of it is plotted in figure 4.7. If autocorrelation of chirp is compared to the one of cosine (figure

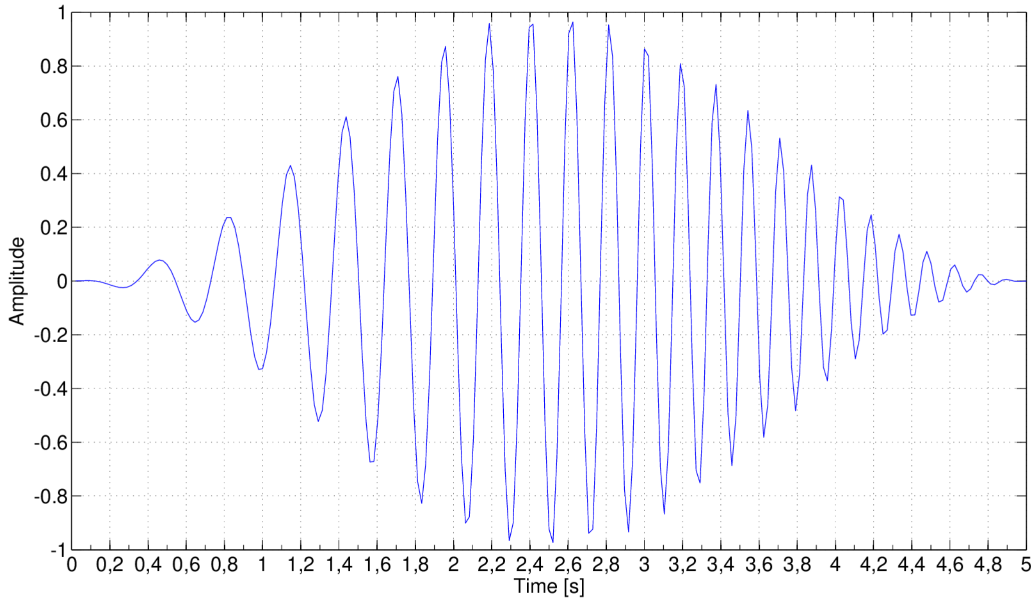


Figure 4.6: Five milliseconds of chirp wave windowed by Hann window.

4.5), the side lobes are significantly smaller and narrower. It is caused by the factor that every period of chirp has a different frequency. This makes chirp wave a suitable signal for distance measurement and more testing will be performed with it in chapter 6.

4.1.4 White noise

White noise wave is not meeting the conditions from chapter 3 because its frequency range is unlimited. Five milliseconds of it are plotted in figure 4.8. Every part of white noise is independent on other parts which leads to ideal cross-correlation with itself. This cross-correlation is plotted in figure 4.9. Autocorrelation of white noise has insignificant side lobes and sharp correlation in zero lag. This makes it perfect signal for distance measurement but because of characteristics of device components it cannot be played and recorded cleanly. Because of its superior autocorrelation, an additional testing is performed in chapter 6.

4.1.5 Filtered white noise

The last candidate is white noise wave filtered by band-pass filter to correspond to the limits of device characteristics founded by chapter 3. The frequency characteristic of filter

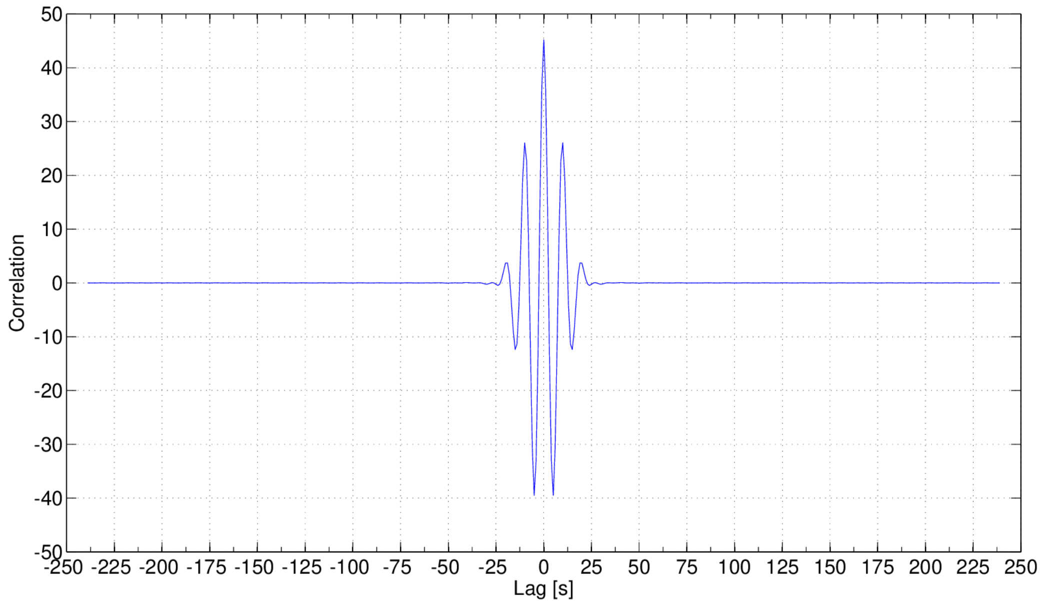


Figure 4.7: Autocorrelation of chirp wave from 4.6.

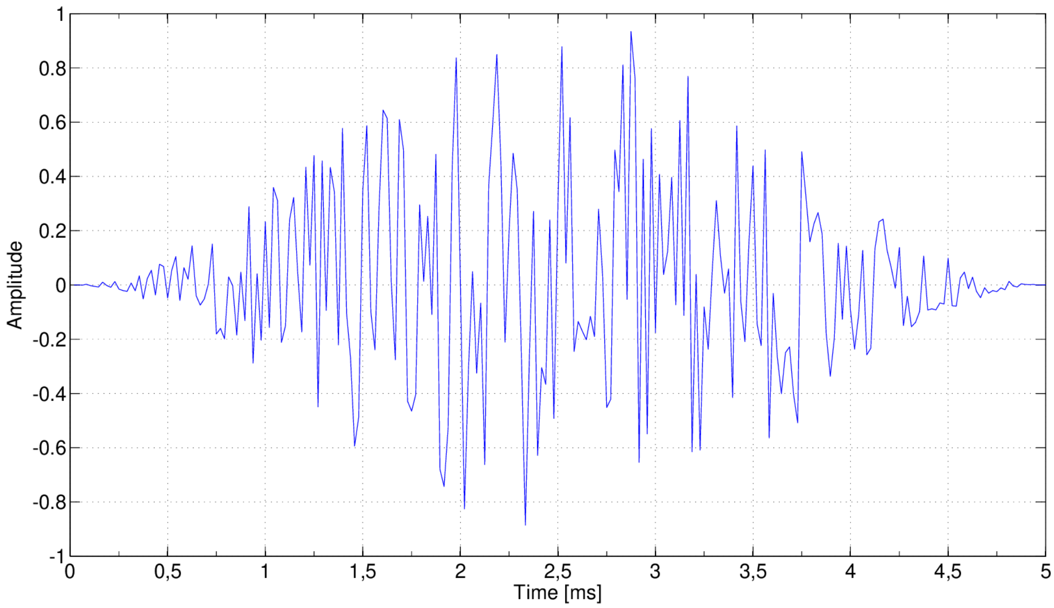


Figure 4.8: Five milliseconds of white noise wave windowed by Hann window.

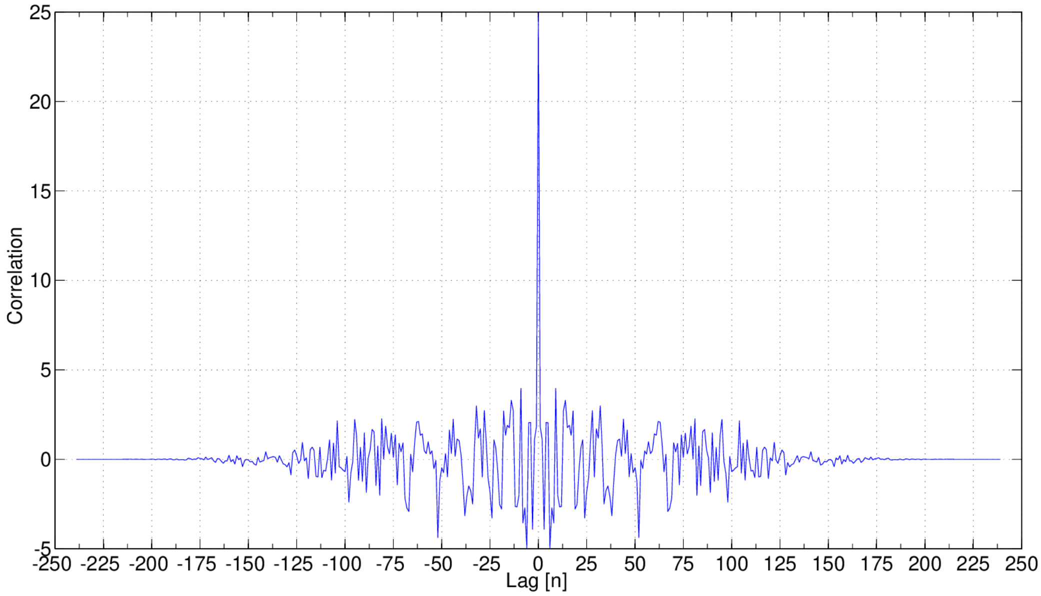


Figure 4.9: Autocorrelation of white noise wave from 4.8.

is plotted in 4.10 and five milliseconds of filtered white noise are plotted in figure 4.11. Autocorrelation of it is plotted in figure 4.12. Filtered white noise is one of the candidates because it represents the signal which the device is able to play and record cleanly and has superior cross-correlation. Auto-correlation of filtered white noise has smaller but significantly wider side lobes than the chirp described in section 4.1.3. Wide side lobes can cause enlargement of minimal distance the application can measure.

4.2 Distance measurement algorithm

The algorithm of distance measurement performs several steps.

1. Start recording on device, wait 100 milliseconds to give the device time to .
2. Play the original sound wave on device with speaker pointed to distant object. Once the playing is finished, wait 100 milliseconds.
3. Stop the recording on device.
4. Locate first value bigger than treshold and mark it *startOfSeparation*. Separate recorded signal to 25 ms long individual measurement, starting from *startOfSeparation* minus three milliseconds.
5. Perform cross-correlation of each individual measurement with corresponding original signal.
6. Calculate mean cross-correlation from all cross-correlations.
7. Locate the first maximum in mean cross-correlation, this corresponds to signal recorded directly through device, with negligible delay.

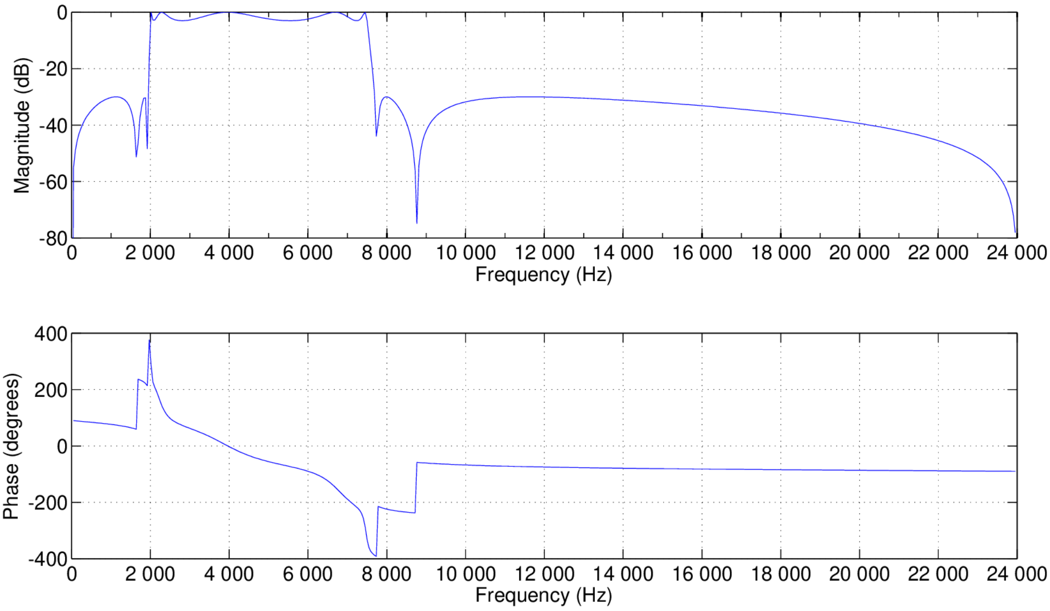


Figure 4.10: Magnitude and phase response of band-pass filter.

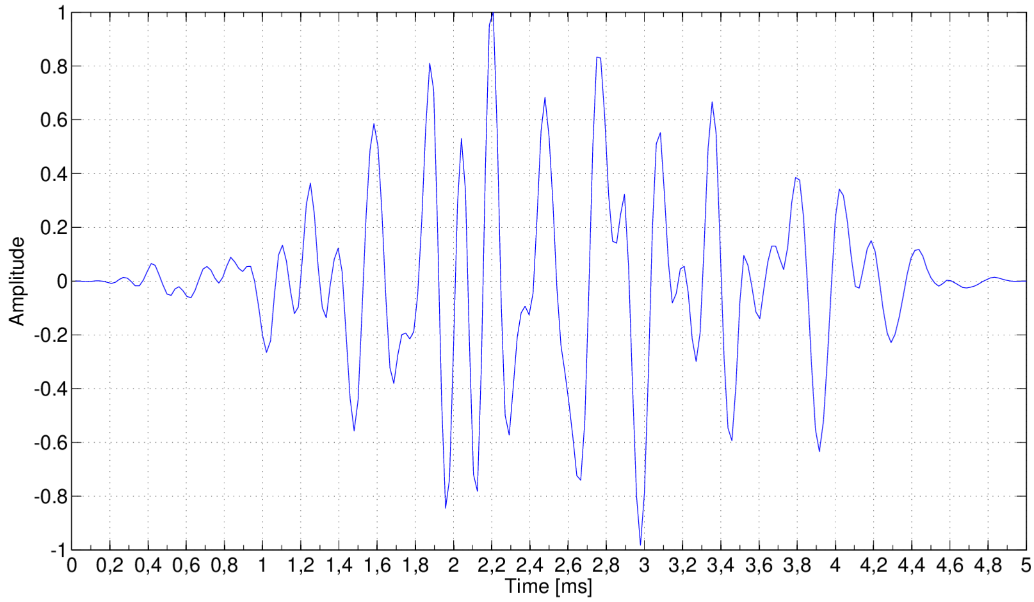


Figure 4.11: Five milliseconds of filtered white noise windowed by Hann window.

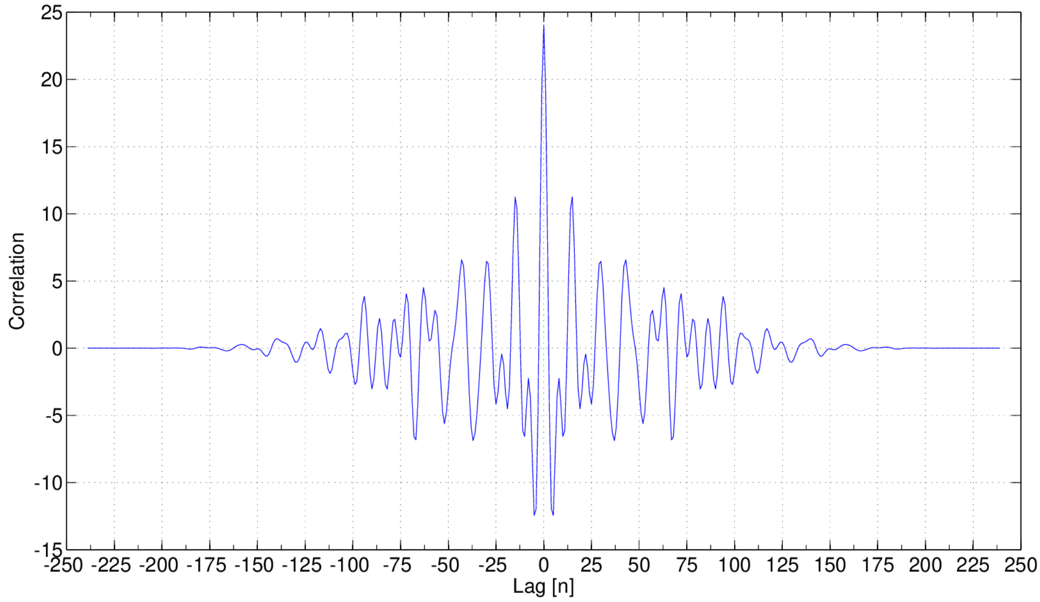


Figure 4.12: Autocorrelation of filtered white noise wave from 4.11.

8. Locate second maximum in the rest of cross-correlation, starting with the first maximum + N samples. This corresponds to reflected signal from object.

9. Calculate number of samples betweem the first and the second maximum, calculate distance from samples using formula 4.4.

Separation of individual measurement (step 4) is performed by simply cutting the recorded signal into parts of length 25 milliseconds. This length has been decided in section 4.1.1. The start of first individual measurement is found by locating the first value bigger than the threshold, it is position where the first active part starts playing. When the separation is done, the next step is to perform cross-correlation for every measurement. Cross-correlation is calculated using fast Fourier transform 2.2.2 and formula 2.3. Average cross-correlation is calculated from individuals by using arithmetic mean.

In average cross-correlation, we are looking for two maxima. The first one is caused by the signal which is recorded during playing and the second one is caused by the reflected signal which is recorded after the playing of sound is finished. The first maximum is much larger, because the signal recorded directly through device is stronger, than the one reflected. To be able to locate both maximums correctly, we have to locate the first and then start looking for the second one which is located later. An example of average cross-correlation is plotted in figure 4.13, because we are also looking for negative cross-correlation, the absolute value of cross-correlation (figure 4.14) is used for looking for maxima.

The first maximum is located around sample 900 and the correct second one is located around sample 1110. Values before the first maximum are not interesting for us, but any of the local maxima after it can be the correct second maximum. To find the correct one, a two steps approach is used. The first step is to skip 135 samples after the first maximum, as the cross-correlation is big around the first maximum, it is caused by side lobes (figure 4.12). The second step is special maximum search using window, for searching the limited range from the index of first maximum plus 135 samples to the end of cross-correlation

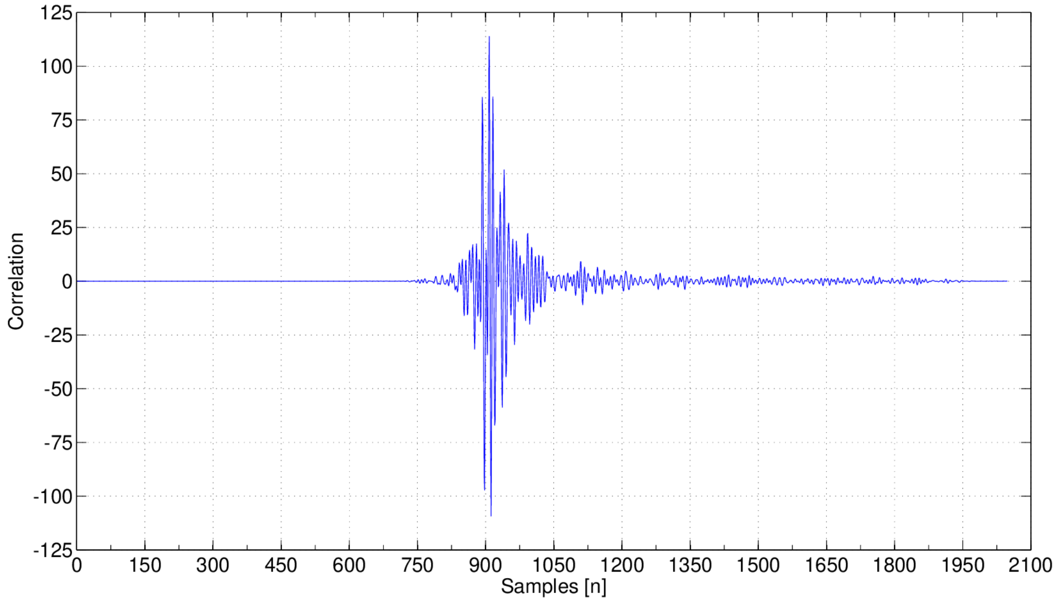


Figure 4.13: Average cross-correlation of filtered white noise.

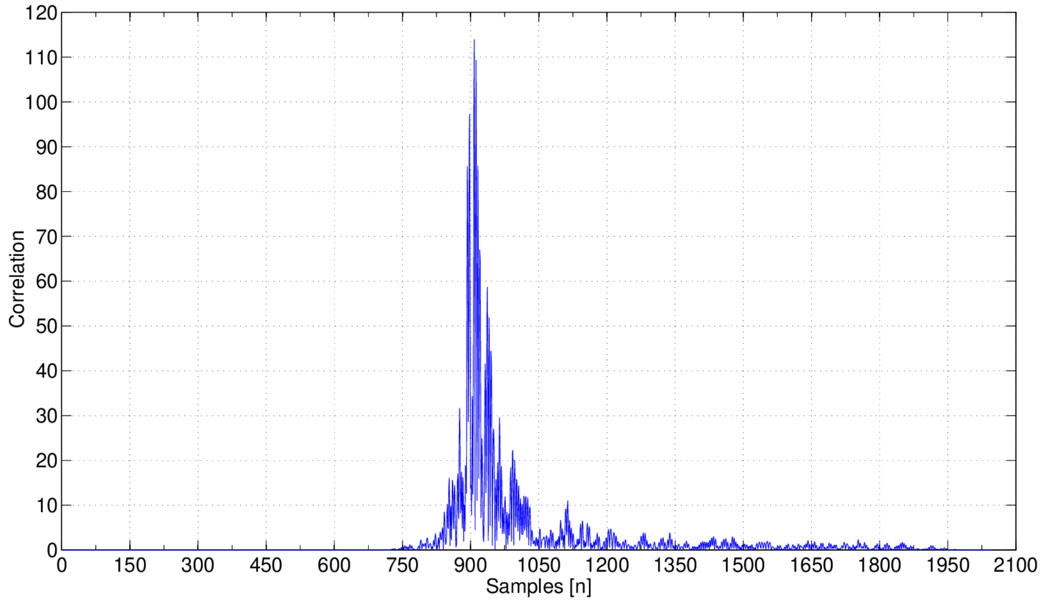


Figure 4.14: Average absolute cross-correlation from figure 4.13.

curve:

1. Set *window start* and *current maximum* to zero. Set *current index* to index of first maximum plus 135.

2. Take window of fifty samples from limited range started with *current index* and calculate the sum of it.

3. If the sum is bigger than the *current maximum*, set the sum as *current maximum* and *current index* as *window start*.

4. If *current index* plus size of window is outside the limited range, go to step 5. Otherwise increase *current index* by ten and go to step 2

5. Locate maximum in *window start* plus fifty and declare the index of it as second maximum.

When both maxima are located, the distance is calculated from their indices using

$$d = \frac{r_s \cdot N}{2F_s}, \tag{4.4}$$

where d is distance, r_s is speed of sound, N is number of samples and F_s is sampling frequency of recorded signal from which the cross-correlation is calculated. The $\frac{1}{2}$ is used because the number of samples samples represents distance from device to object and back.

Chapter 5

Application

In this chapter, the logic and design of application are described. It is divided into two main parts. The first is focused on user interface and the second is focused on logic and computational part where the communication and function of four threads is described. The application is called *Echo Distance measurement*.

5.1 User interface

The user interface (UI) is designed as simple as possible. It contains one button with which the measurement is started and a text field where the measured distance is displayed. UI is launched in the main thread, separate from other parts. This is because the computing of distance is not done immediately and UI has to respond for user command at that time. Otherwise, the user has the impression that the application is frozen. The UI is plotted in figure 5.1.

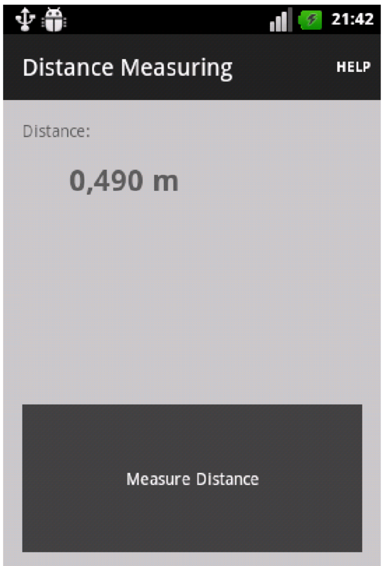


Figure 5.1: User interface of application.

5.2 Logic and computational part

Because of restrictions of the environment, the recording and playing of signal is running in separate threads to be able to run concurrently. Every time the button is pressed, new recording and playing threads are created. They need to be synchronized with each other, this is guaranteed by shared variables with exclusive access. While the playing and recording threads are communicating, every message between them is delayed by 100 milliseconds, this delay is added to communication to give additional time to devices components to start recording and finish playing. To achieve efficiency and to keep UI responsive, the computational part is also launched in separate thread. The scheme of threads communication in application is plotted in figure 5.2. As *countDownTimer* which is used for 100 milliseconds delay can be created only in the main thread, all communication between thread runs through the UI thread.

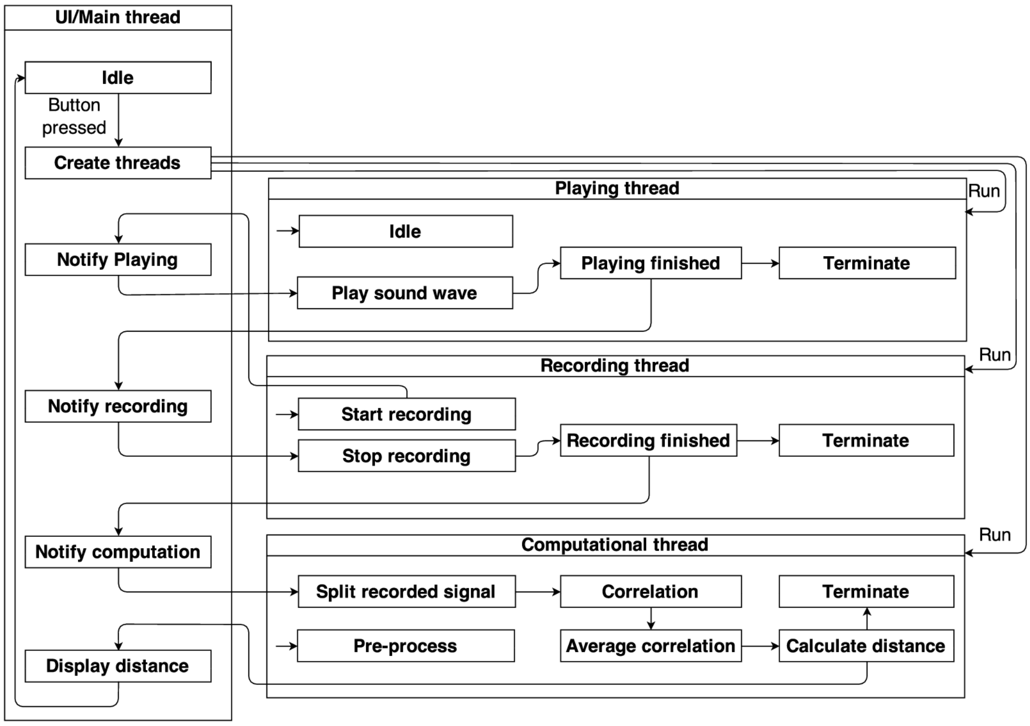


Figure 5.2: Scheme of application functionality and communication between threads.

5.2.1 Playing thread

Playing thread is launched when the button is pressed, its classes and functions are the same as in *SoundTesting* application (see section 3.1). Before the sound can be played, the recording has to be started (see the following section). After it is done, with a delay of 100 milliseconds the *playing thread* is notified and starts playing the sound wave. When playing is finished, the *recording thread* is notified and this thread is terminated.

5.2.2 Recording thread

When the button in the application is pressed, the *recording thread* is launched. Classes and functions used are the same as in *SoundTesting* application (see section 3.1). The difference is that the recorded signal is not saved in file because it is immediately processed by *computational thread*. No parameters are needed to launch the thread, the instance of *AudioRecord* class is initialized and recording starts. When this thread is notified by *playing thread*, with a delay of 100 milliseconds the recording stops. Before the thread is terminated, it notifies the *computational thread* and the recorded signal is handed over to it.

5.2.3 Computational thread

This thread calculates the distance from original signal, recorded signal, sampling frequency and speed of sound. When the button is pressed, the original signals are loaded into memory to be used for cross-correlation, when finished, the thread is waiting for the *recording thread* to hand over the recorded signal. The splitting, average cross-correlation and distance calculation with searching for maximums is performed, algorithms described in chapter 4. When the distance is calculated, it is handed over to the UI thread to be displayed on the user screen.

Chapter 6

Testing

In chapter 4.1, three signals were chosen as usable for measurement of distance: chirp, white noise and filtered white noise. To find the most reliable of them and to assess the distances which the application can measure, test measurement was performed. It is composed from measurement of thirteen distances with ten measurements at each of them. Because of the algorithm used the minimum distance it can measure is half a meter (see section 4.1.1). The maximum is approximately three and half meters (see section 4.1.1) but we expect it will be less because the reflection of sound is significantly alternated and the surrounding noise corrupts the measurement of such distance. The testing was performed outside with low amount of noise. We considered measurements with precision to one tenth of a meter as succesful one. The object to which the distance is measured was one square meter sheet of iron leaning against the wall. The photo of measurement can be seen in figure 6.1.

Figure 6.1: Measurement settings.

The results of measurement of the first distance, half a meter are plotted in graph 6.2. From this figure, it can be seen, that the application is able to measure thus distance

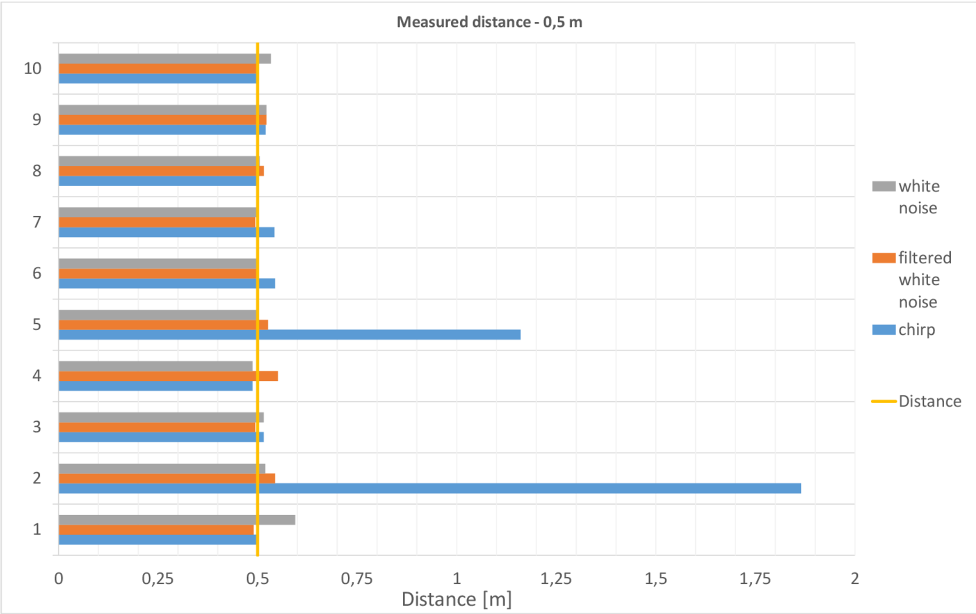


Figure 6.2: Measurement of half meter with chirp, filtered white noise and white noise.

well. The best results achieved with filtered white noise which was able to measure the distance in all cases. The white noise was also able to measure distance in all cases but with slightly worse precision. On the other hand, the chirp failed to measure distance in two cases, in other measurements it achieved the same precision as filtered white noise. The next testing distance was one meter and the results are in graph 6.3. The chirp failed to measure this distance in all cases,it returned always the same distance of 0.85 meters. White noise and filtered white noise achieved similar accuracy in all tenth measurements, the first mentioned achieved a slightly higher precision in some of them. The measurement of one and half meter is plotted in graph 6.4. The measurement for this distance achieved similar results as the one for one meter. The precision of chirp is on edge of one tenth of meter — the measured values are around 1.4 meters. The white noise and filtered white noise even achieved again similar results with slightly higher precision for white noise. Both signals measure approximately about five centimeters shorter distance. The white noise started to fail with distance two meters, see figure 6.5. The best performing signal was the chirp which was able to measure distance in all cases and with the highest precision, filtered white noise was able to measure this distance with precision one tenth of meter in seven out of ten cases. The last distance which was measured is three meters, the results are plotted in graph 6.6. The chirp and white noise are unable to measure this distance in nine out of ten measurements, filtered white noise achieved much better results for this distance. It is able to measure distance three meters in eight out of ten cases. The results of all thirteen measured distances are plotted in graphs in appendix B.

The number of successful measurements for individual signals is plotted in graph 6.7, ten measurements were performed for every distance and a successful measurement means that the distance is measured in precision to one tenth of meter. If a signal achieves ten in individual distance, it was able to measure the distance in all ten cases. In graph 6.8, the average value of measured distance from successful cases is plotted.

The testing demonstrated that the most precise and safe signal for measurement is

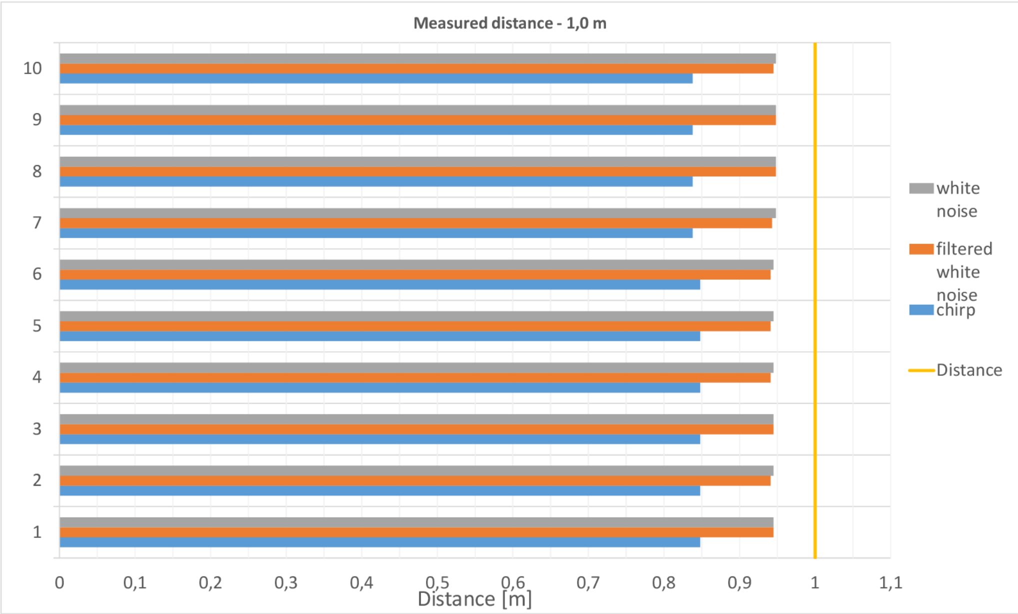


Figure 6.3: Measurement of one meter with chirp, filtered white noise and white noise.

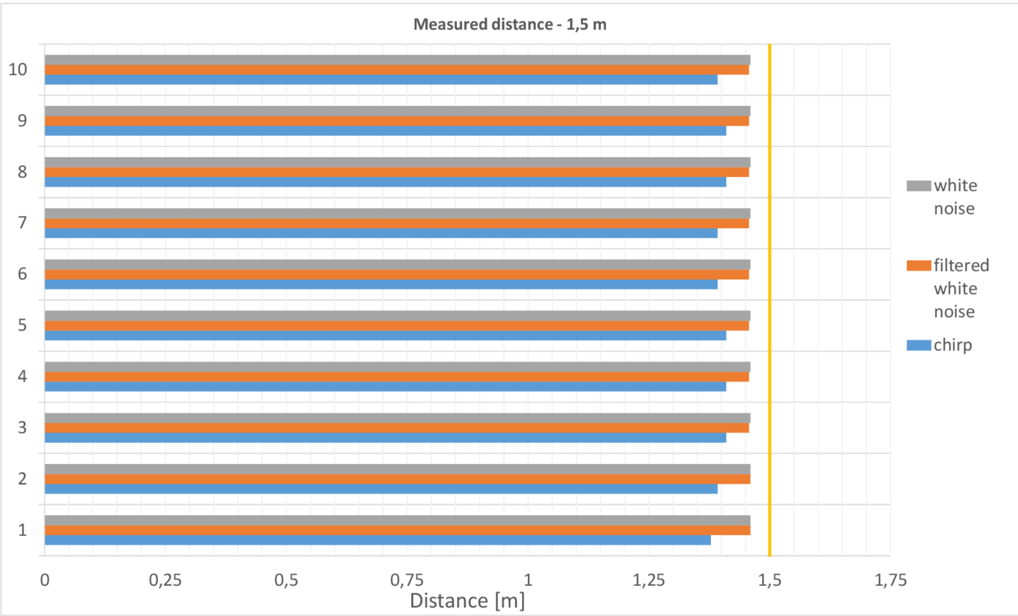


Figure 6.4: Measurement of one and half meter with chirp, filtered white noise and white noise.

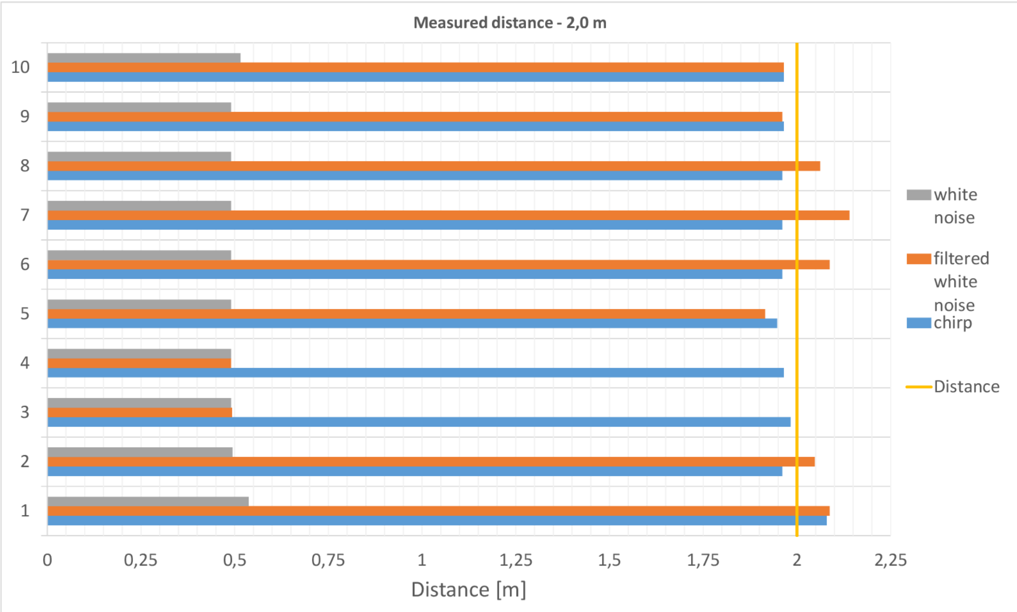


Figure 6.5: Measurement of two meters with chirp, filtered white noise and white noise.

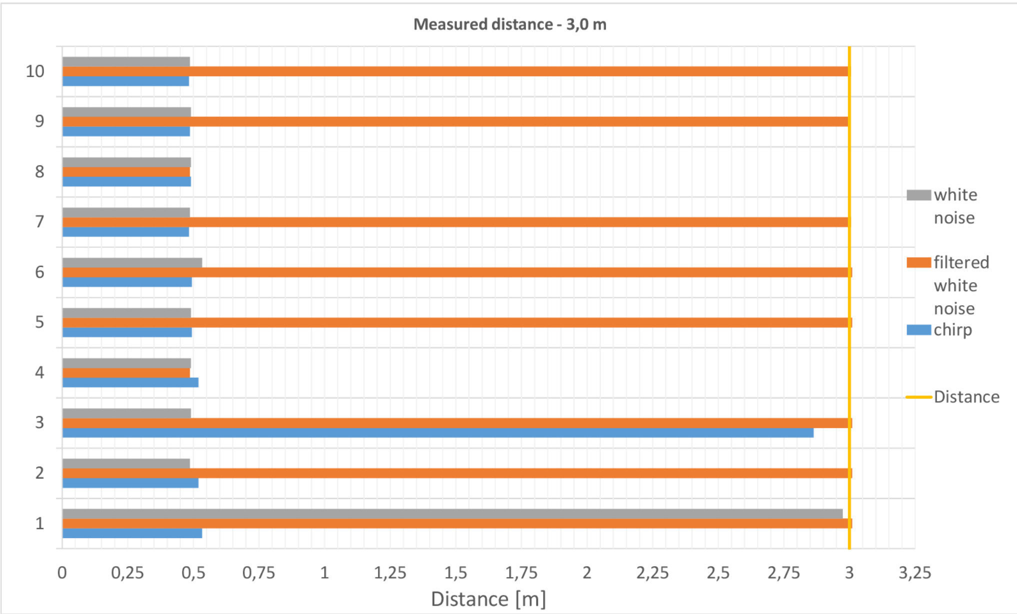


Figure 6.6: Measurement of three meters with chirp, filtered white noise and white noise.

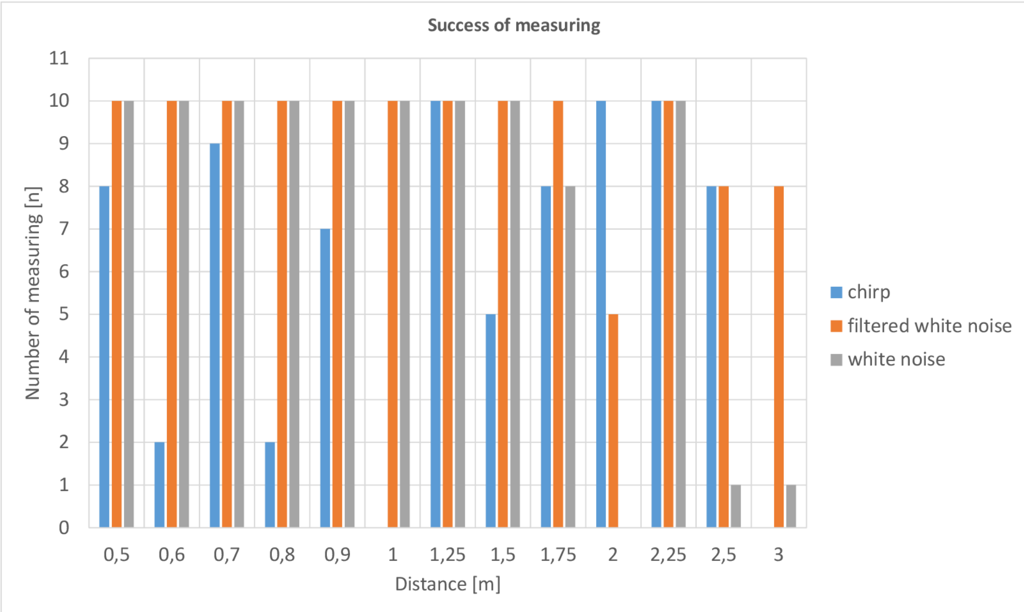


Figure 6.7: Success of measurement for chirp, white noise and filtered white noise.

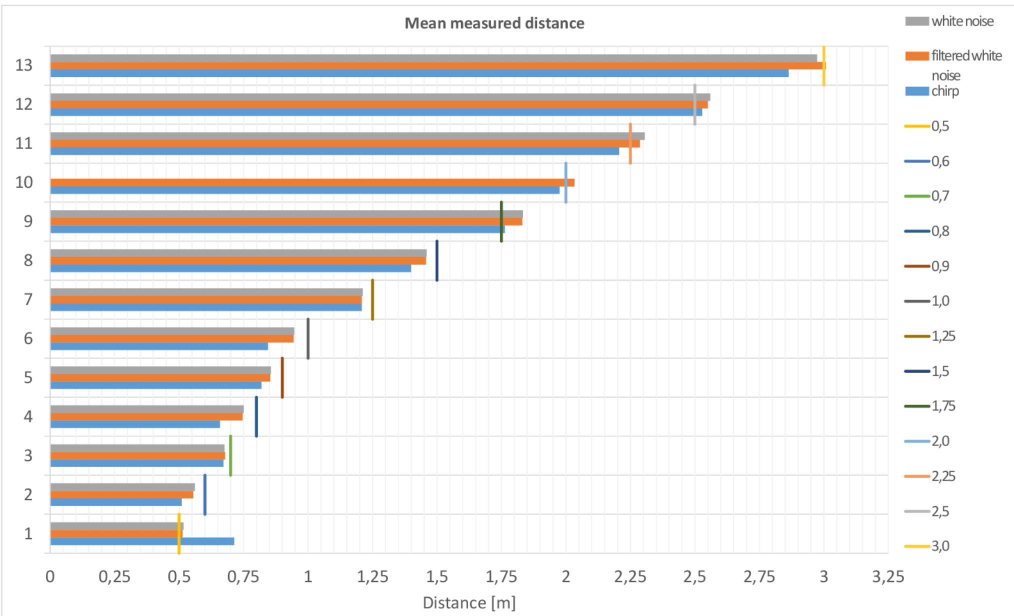


Figure 6.8: Average measured distance for successful cases of chirp, white noise and filtered white noise.

filtered white noise. It is able to measure short distances with higher precision than chirp and it can also measure the longer distances where the white noise starts to fail. The shortest distance, as was already aforementioned, is half a meter because of limitations of the algorithm used. The longest distance which was tested is three meters, trying to measure longer distance mostly leads to fail which are caused by loss of reflected signal in the surrounding noise. In an environment with a low level of noise and an object which can reflect the sound well, it is possible to measure longer distances up to approximately three and a half meters (see section 4.1.1).

Chapter 7

Conclusion

7.1 Summary

The results of the work are the design of algorithm and creation of application which is enable to measure distance using echo on a mobile device. To use echo, the characteristics of loudspeaker and microphone of the device were analyzed, the frequency characteristics show that the loudspeaker and the microphone are able to work well with frequencies from 2 000 Hz to 7 500 Hz. The loudspeaker of the device play the sound and the microphone records the reflection from the object. To locate reflection in recorded signal, the cross-correlation is used, the usability of cross-correlation is highly influenced by the shape and frequency of used sound. Three shapes were suggested as usable for using: chirp, filtered white noise and white noise. The first two mentioned ones contain frequencies in the range of well-played frequencies by the device. The last one, white noise, does not have a frequency limit. To perform testing of these three signal in real time situations, an application for mobile device with operating system Android was created. The algorithm used by the application is performing forty measurements at once to overcome the weak reflections and burried in noise. It contains evaluation of cross-correlation of recorded and played signal using fast Fourier transform. In the resulting coefficients, two maxima are located. The first one represents the sound recorded during playing and the second one represents the recorded reflection. The time between these two maxima is coverted to the distance using speed of sound. The application with designed algorithm achieves the best results using filtered white noise with frequency range 2 000 Hz to 7 500 Hz, the range of distances it is able to measure is from half a meter to approximately three meters taking into account the noise environment.

7.2 Future work

The actual version of application used for measurement takes a few seconds to measure the distance, it is caused by splitting the forty individual measurement and calculating cross-correlations using double precision. To speed up the application, the separation can be omitted using only indexes instead of separate signals. Mobile device is slow when working with double precision which would suggest using fixed point arithmetic.

The algorithm to calculate distance currently using cross-correlation and forty individual measurement to enhance the results. The length of one measurement is limited, otherwise the reflection will be mixed with and lost in high strength transmitted sound. Instead of using of forty individual measurements and cross-correlation to calculate distance, a different approach can be chosen: using one continuous signal for transmitting and recording, the distance is then calculated using phase interference of transmitted and reflected sounds.

Bibliography

[1] Glen Ballou. *Handbook for sound engineers.* Boston: Focal Press, 2008. p. 597-641. ISBN 9780240809694.

[2] Android developers. Android developers. [online]. `https://developer.android.com/index.html`, [cit. 2008-11-28].

[3] E FRIAUF. Hearing. e-neuroforum. [online]. `http://link.springer.com/article/10.1007/s13295-014-0062-8`, 201409, roč. 5, č. 3, s. 51-52 [cit. 2015-05-13].

[4] IFEACHOR, Emmanuel C. a Barrie W JERVIS. *Digital signal processing: a practical approach.* Harlow: Prentice-Hall, 2002. s. 104-316. ISBN 0-201-59619-9.

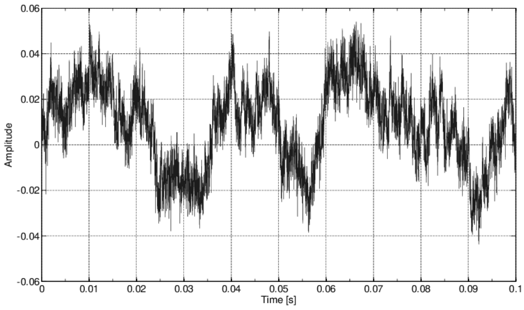
[5] Jiří JAN. *Číslicová filtrace, analýza a restaurace signálů.* Brno: Vysoké učení technické, 1997. s. 89-111. ISBN 80-214-0816-2.

[6] PRABHU, K.M.M. *Window functions and their applications in signal processing.* Hoboken: Taylor and Francis, 2013. p. 138. ISBN 978-1-4665-1584-0.

[7] A WAITE. *Sonar for practising engineers. 3rd ed.* Chichester: Wiley, 2002. ISBN 0471497509.

Appendix A

Testing of loudspeaker using sine waves



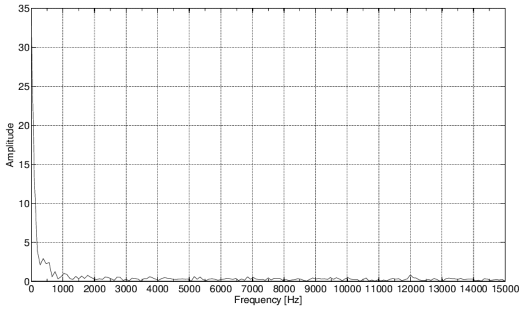
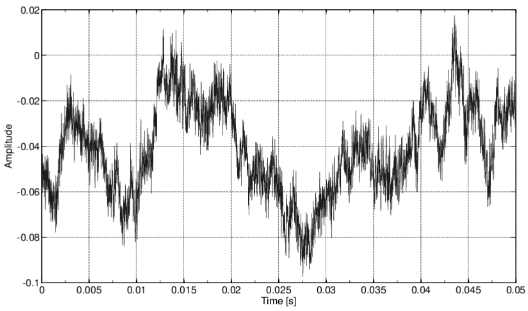


Figure A.1: Recorded signal 55 Hz and its spectrum.



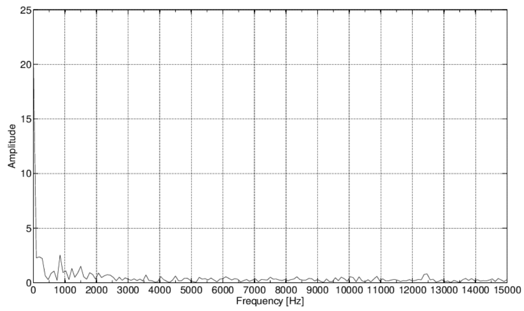


Figure A.2: Recorded signal 110 Hz and its spectrum.

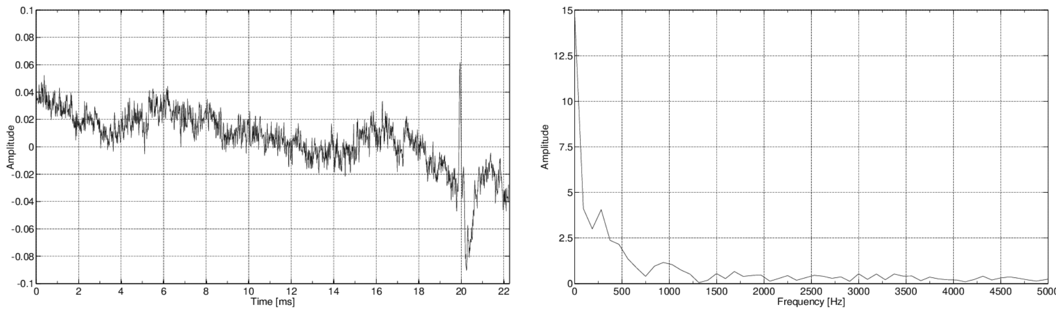


Figure A.3: Recorded signal 220 Hz and its spectrum.

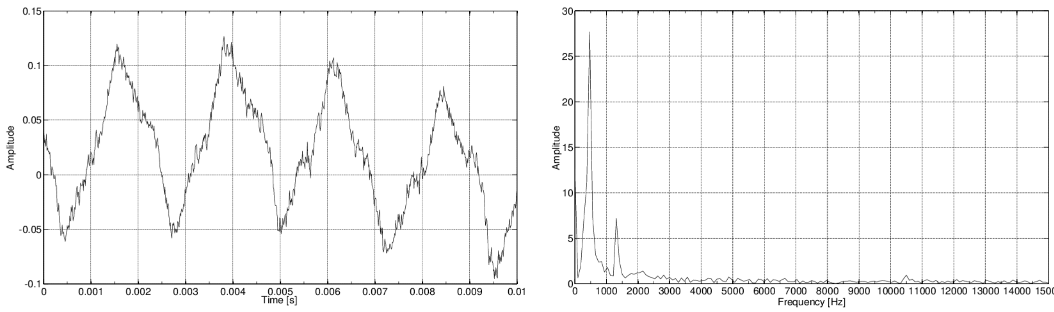


Figure A.4: Recorded signal 440 Hz and its spectrum.

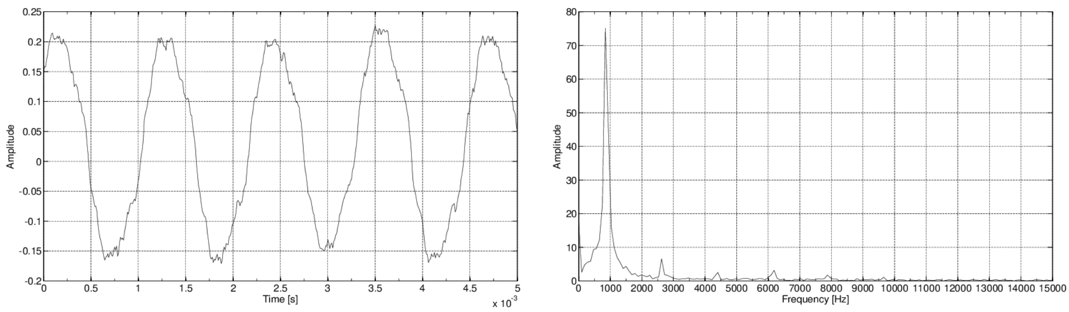


Figure A.5: Recorded signal 880 Hz and its spectrum.

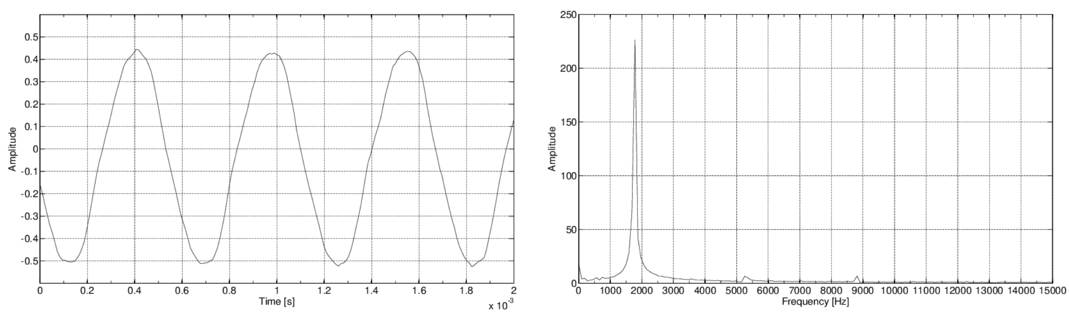


Figure A.6: Recorded signal 1 760 Hz and its spectrum.

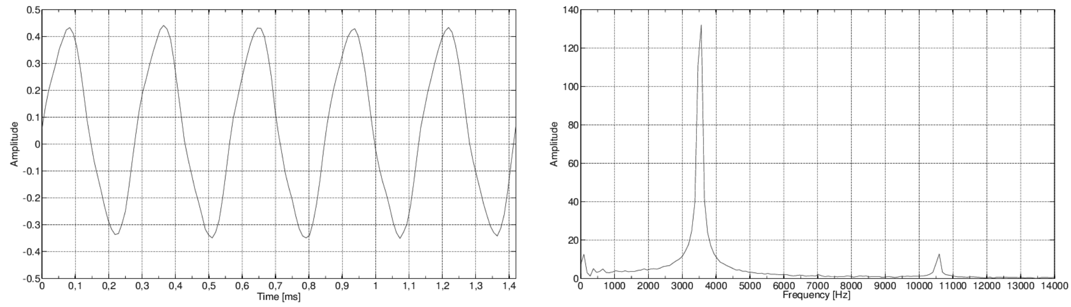


Figure A.7: Recorded signal 3 520 Hz and its spectrum.

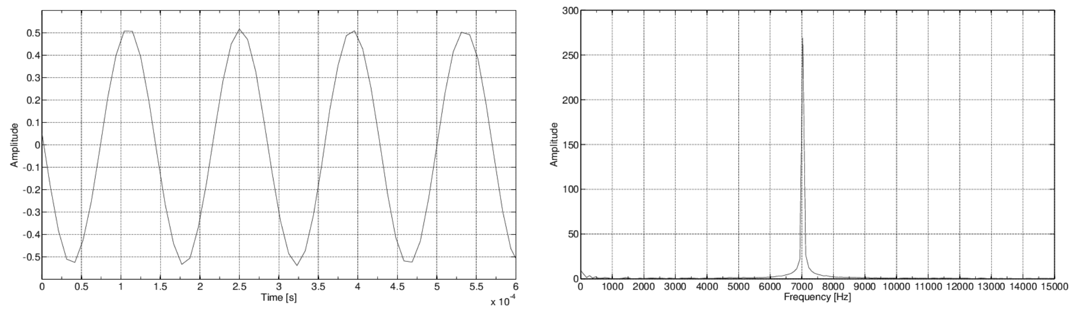


Figure A.8: Recorded signal 7 040 Hz and its spectrum.

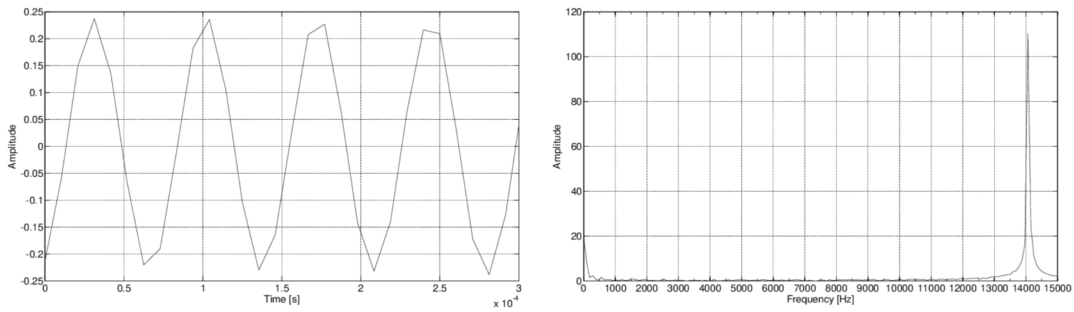


Figure A.9: Recorded signal 14 080 Hz and its spectrum.

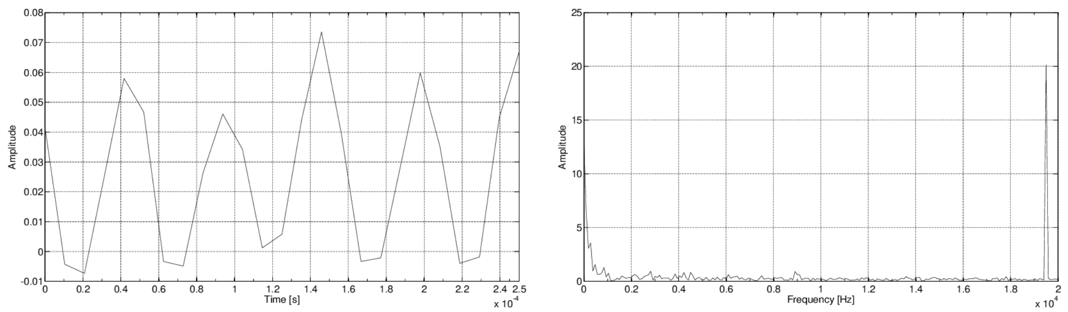


Figure A.10: Recorded signal 19 500 Hz and its spectrum.

Appendix B

Measurement of distance

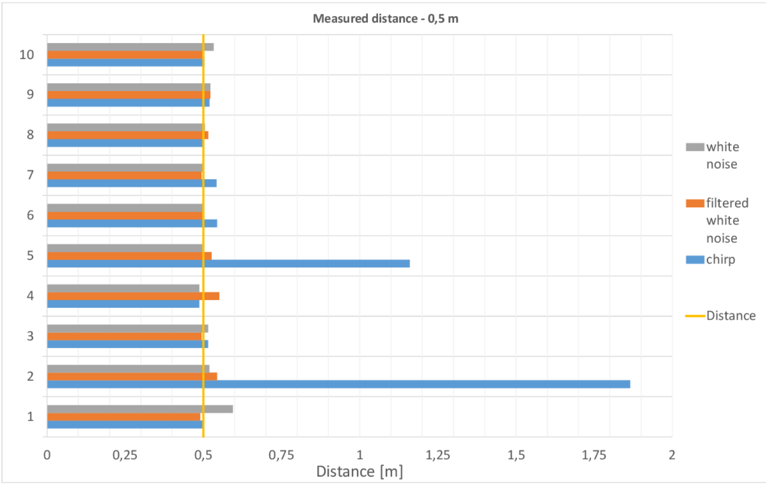


Figure B.1: Measurement of 0.5.

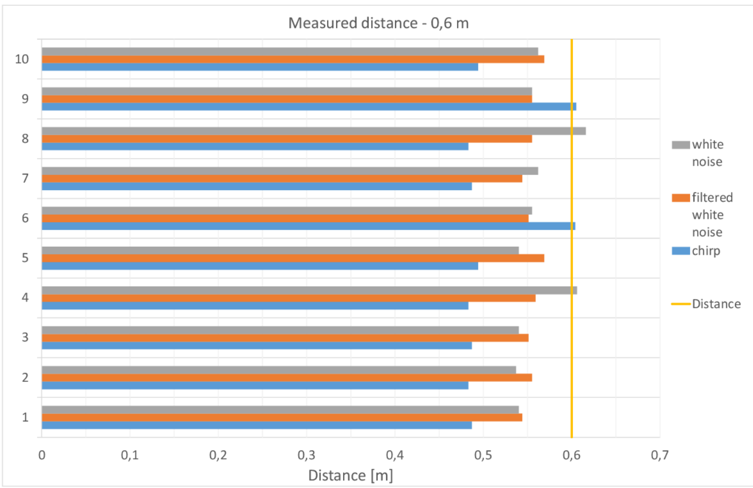


Figure B.2: Measurement of 0.6 meters.

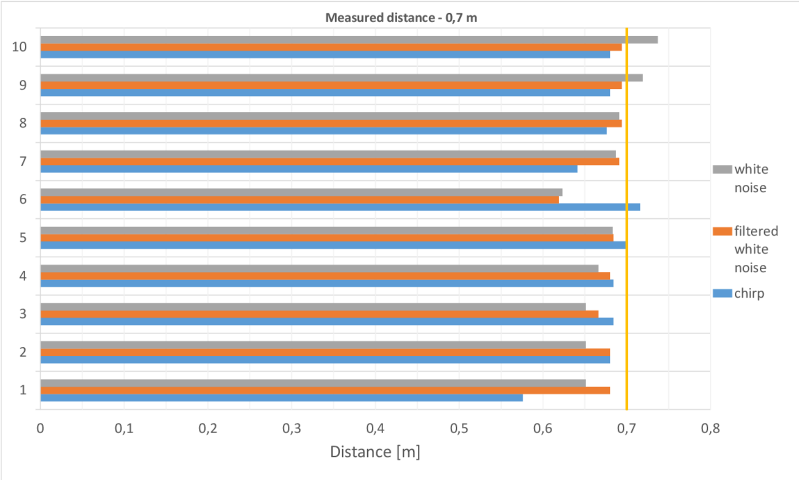


Figure B.3: Measurement of 0.7 meters.

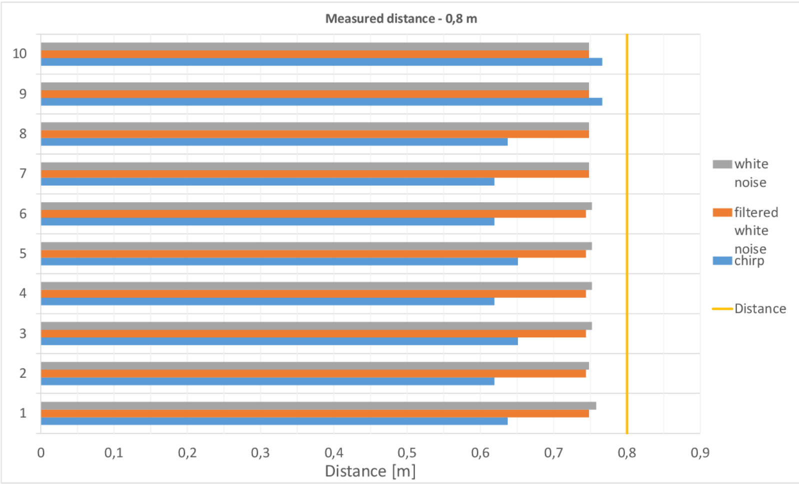


Figure B.4: Measurement of 0.8 meters.

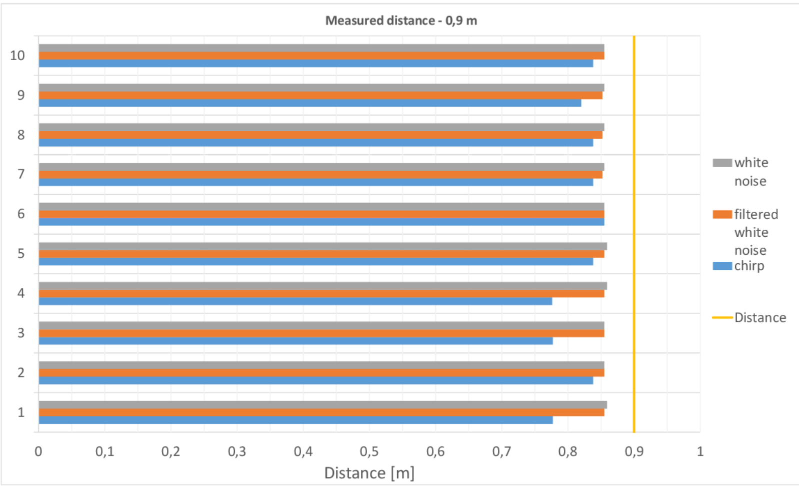


Figure B.5: Measurement of 0.9 meters.

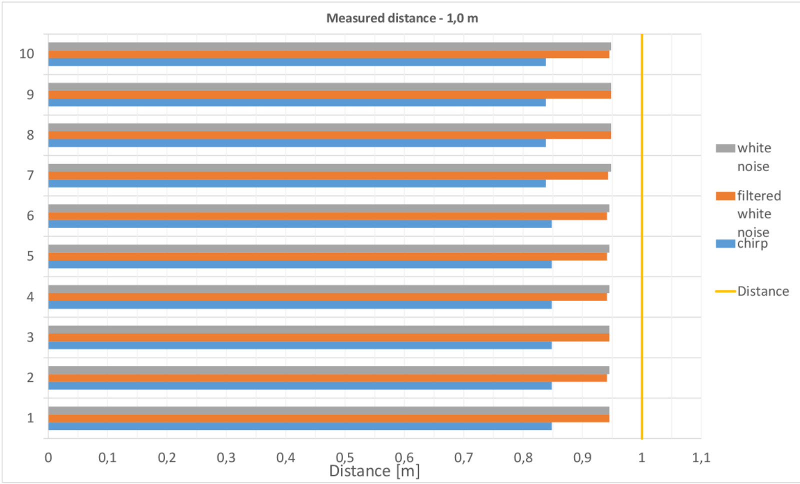


Figure B.6: Measurement of 1.0 meters.

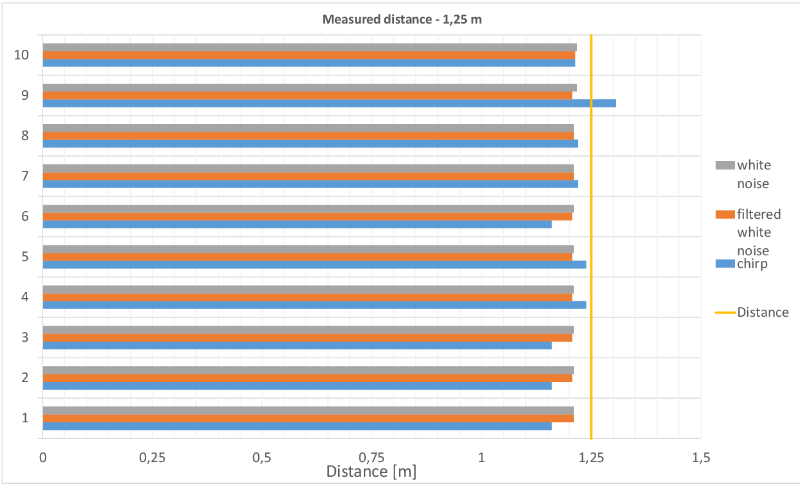


Figure B.7: Measurement of 1.25 meters.

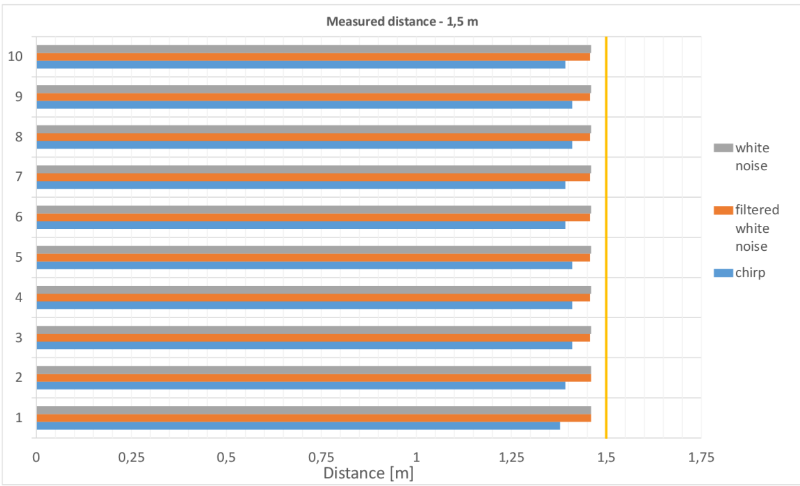


Figure B.8: Measurement of 1.5 meters.

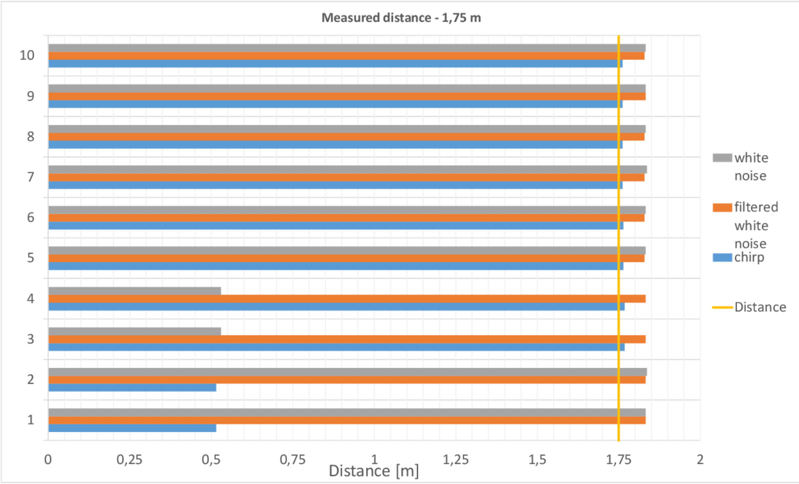


Figure B.9: Measurement of 1.75 meters.

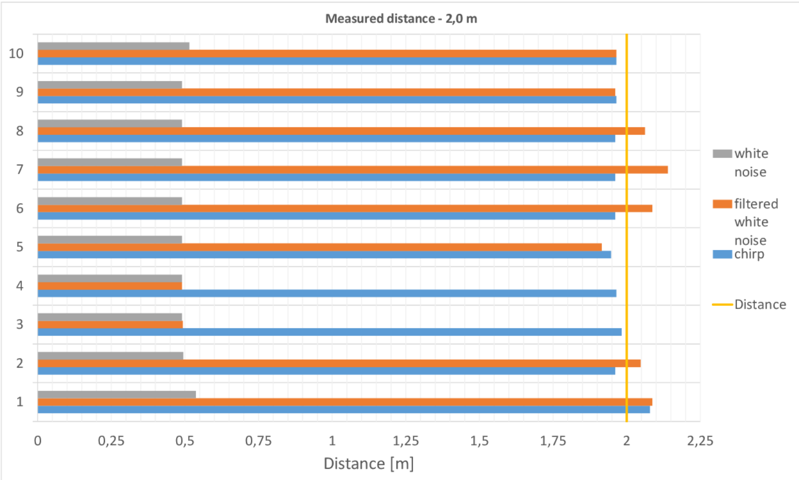


Figure B.10: Measurement of 2.0 meters.

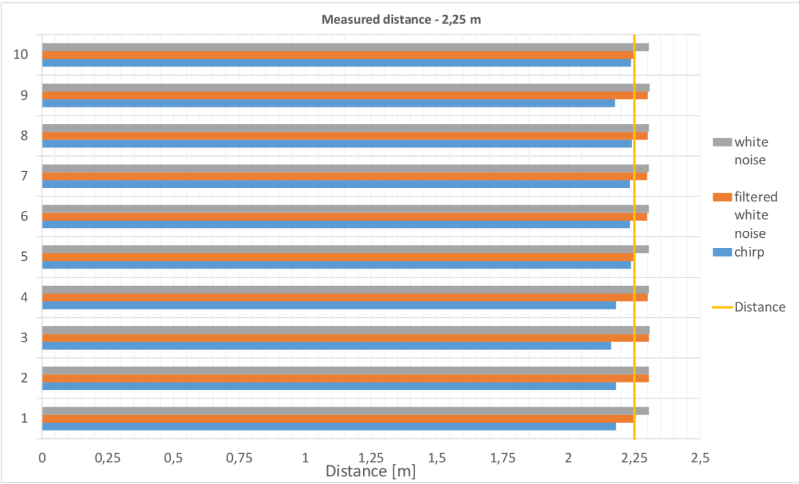


Figure B.11: Measurement of 2.25 meters.

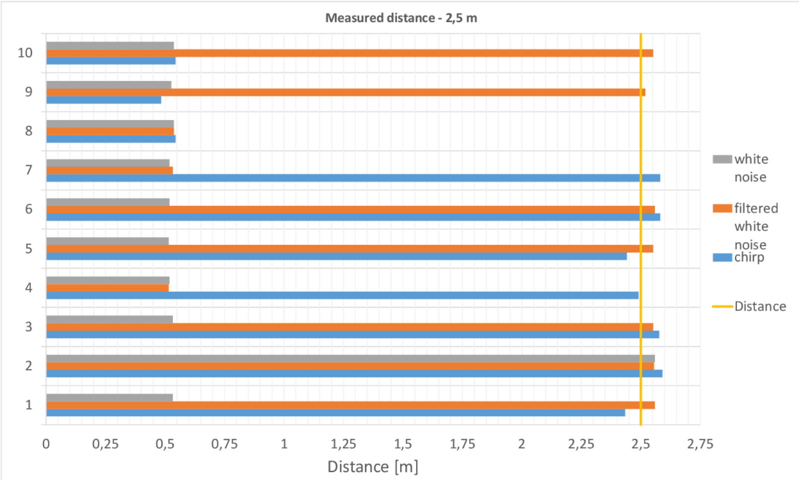


Figure B.12: Measurement of 2.5 meters.

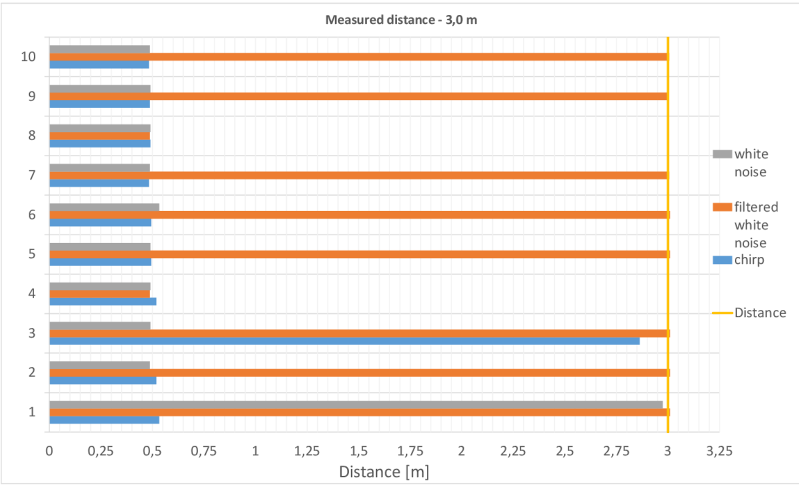


Figure B.13: Measurement of 3.0 meters.

Appendix C

CD content

- Source code
  - Android studio project - application *Echo Distance measurement*
  - Andorid studio project - application *soundTesting*
- Text
  - Electronic version - pdf file
  - LaTeXsource code - zip file
  - Graphs and pictures in pdf format.
- Poster
  - Pdf version
  - Microsoft Word version

Appendix D

Poster