NTNU
Norwegian University of
Science and Technology

Performance of a Multichannel Audio Correction System Outside the Sweetspot.

Further Investigations of the Trinnov Optimizer.

Joachim Olsen Wille

Master of Science in Electronics
Submission date: June 2008
Supervisor: Peter Svensson, IET

Norwegian University of Science and Technology
Department of Electronics and Telecommunications

Problem Description

In the project, an existing room correction device from Trinnov audio should be analyzed. The analysis could use measurements carried out in a previous project.
A standard surround-sound setup with 5 loudspeakers should be set up in an anechoic chamber, together with a Trinnov Optimizer correction device. A microphone array should be used to measure the response in an area outside the sweetspot. Analysis with, e.g., Matlab should focus on the performance of the correction algorithms not only in the center/sweetspot, but also in the surrounding area. The correction device's claimed ability to correct for room reflections should be investigated.

Assignment given: 23. January 2008
Supervisor: Peter Svensson, IET

Abstract

This report is a continuation of the student project "Evaluation of Trinnov Optimizer audio reproduction system". It will further investigate the properties and function of the Trinnov Optimizer, a correction system for audio reproduction systems. During the student project measurements were performed in an anechoic lab to provide information on the functionality and abilities of the Trinnov Optimizer. Massive amounts of data were recorded, and that has also been the foundation of this report. The new work that has been done is by interpreting these results through the use of Matlab.

The Optimizer by Trinnov [11] is a standalone system for reproduction of audio over a single or multiple loudspeaker setup. It is designed to correct frequency and phase response in addition to correcting loudspeaker placements and cancel simple early reflections in a multiple loudspeaker setup. The purpose of further investigating this issue was to understand more about the soundfield produced around the listening position, and to give more detailed results on the changes in the soundfield after correction. Importance of correcting the system not only in the listening position, but also in the surrounding area, is obvious because there is often more than one listener. This report gives further insight in physical measurements rather than subjective statements, on the performance of a room and loudspeaker correction device.

WinMLS has been used to measure the system with single, and multiple microphone setups. Some results from the earlier student project are also in this report to verify measurement methods, and to show correspondence between the different measuring systems. Therefore some of the data have been compared to the Trinnov Optimizer's own measurements and appear

similar in this report. Some errors found in the initial report, the results from the phase response measurements, have also been corrected.

Multiple loudspeakers in a 5.0 setup have been measured with 5 microphones on a rotating boom to measure the soundpressure over an area around the listening position. This allowed the effect of simple reflections cancellation, and the ability to generate virtual sources to be investigated.

For the specific cases that were investigated in this report, the Optimizer showed the following:

- Frequency and phase response will in every situation be optimized to the extent of the Optimizers algorithms.
- Every case shows improvement in the frequency and phase response over the whole measured area.
- Direct frontal reflections was deconvolved up to 300Hz over the whole measured area with a radius of 56cm.
- A reflection from the side was deconvolved roughly up to 200Hz for microphones 1 through 3, up to a radius of 31.25cm, and up to 100Hz for microphones 4 and 5.
- The ability to create virtual sources corresponds fairly to the theoretical expectations.

The video sequences that were developed give an interesting new angle on the problems that were investigated. Other than looking at plots of different angles which is difficult and time consuming, the videos showed an intuitive perspective that enlightened the same issues as the common presented data of frequency and phase response measurements.

Contents

List of Figures

1

Introduction

Home cinema is becoming a standard of every home and never before have the average Joe been more aware of the quality of audio surrounding us on a daily basis. Mp3 players and Walkman mobile phones are in everybody's pockets, and we cannot escape the music anywhere. Music is close to synonymous with audio, and few people these days can say that they do not have a relationship with music or audio. Not everybody has knowledge about audio, but almost everybody has an opinion. Audio professionals are trying to push the limits on sampling, frequency range, dynamic range and overall quality, but does our equipment match the quality?

Multichannel audio reproduction systems have in the last few years been accepted into the line of consumer electronics. The evolvement from stereo reproduction developed in the 60's has been slow, and the problems with stereo has had very little effect on people compared to the price of upgrading to a bigger system. Movie theater systems have been about 20 years ahead of the home cinema systems with early production of films with multiple discrete audio channels.

Multichannel audio for home use has had a major breakthrough with the launching of the DVD format. Some of the former formats have been Quadrophonie, Ambisonics and Dolby Surround with four speakers, and Dolby Pro Logic with a five speaker setup similar to the 5.1 format most common today. All these formats have different ways of encoding and decoding audio information to and from two separate audio channels. The technological progress within audio equipment has made components a lot cheaper, and has eased integration into consumer products. The price of getting movie theater audio formats into the home has been drastically redused, and the DVD has contributed strongly to clear the way for price and convenience needed to bring multichannel surround sound into the home.

A major problem with high sound quality sound reproduction are the surroundings in which it is being played back. Controlled environments like sound production studios can be difficult enough, but yet a lot easier than the average livingroom or home theater. The sound studio scenario is usually only a single studio engineer listening in the sweetspot, and the walls are usually treated with absorbants to create an easy controllable environment. The loudspeakers are at a greater extent placed at the correct angles according to the mixing standard. The room has been customized to reproduce audio correctly. This report will show the simplest of scenarios recreated in an anechoic chamber to show the basic difficulties of high quality sound reproduction that of course apply for studio and home systems.

Production studios need to uphold the high product quality through the whole production chain. This requires analytic production studios with correct reference throughout the entire production line. A recording might go through several different sound recording, production, and broadcasting studios before it ends up at the listeners location. This means that the listening environment in each and every studio should be as equal as possible to prevent unwanted room influence on the recording. In many situations it may be difficult to provide such an environment when for example sitting in a very small room or a TV-networks Outside Broadcasting truck. Properties of the room are important due to possible perceived differences in sound quality in several distinct rooms.

The Optimizer by Trinnov [11] is designed to correct frequency and phase response in addition to correcting loudspeaker placements in a multiple loudspeaker setup. The Optimizer is a powerful computer with multiple soundcards doing realtime audioprosessing. The system consists of the computer itself and a four capsule microphone measuring both impulseresponse and position of each loudspeaker. The system will be corrected in the measured sweetspot, which would be the preferred listening position.

The Optimizer gives the sound engineer the possibility to maintain the same reference regardless of the studio he/she is working in. It will try to eliminate the differences between listening rooms and audio equipment to give the exact same listening experience anywhere.

Work is currently being done to implement Optimizer technology in surround receivers. The existing Optimizer is too expensive for home use. This kind of processing will undoubtedly give the listener an enhanced listening experience both through naturally sounding environments in cinema, correct localization of phantom sources and reproduction of music. In home cinema the loudspeakers are rarely given the right placements because it does naturally not fit with most livingroom interior. This kind of correction device claims to take care of out of position loudspeakers,

and imitate the intended phantom sources independent of loudspeaker placements. Customers can only trust subjective listening tests and manufacturers stated information when they ask the question: "Does it work?" Another problem in out of studio situations, like in the home, is that the number of listeners are rarely limited to one person. These persons are likely to sit around the sweetspot, but only one person will be in it. How well does this system work around the optimal listening position? This report will investigate and evaluate the Trinnov Optimizer in a physical and objective manner in the area surrounding the sweetspot to see what can be corrected in a multiple listener environment.

An Optimizer have been tested with one and five loudspeakers in an anechoic chamber. The soundfield have been measured using a single microphone and a five microphone boom rotated to record the soundpressure over a circle of about 60 cm radius around the sweetspot. The boom was rotated 3° for every impulse response measurement 360° around the sweet spot.

The effects of optimizing will be investigated through looking at flattening of the frequency and phase response, correcting wrongly placed loudspeakers in a 5.0 setup, and cancellation of simple reflections, both from the front and from the side, in the area around the sweetspot.

Theory will be briefly discussed in the next chapter. Chapter 3 will describe how the different measurements were performed, and the equipment involved in each measurement. The conclusive results of the measurements will be presented in chapter 4 and discussed in chapter 5. A conclusion will follow the discussion in chapter 6.

2

Theory

The theory concerning the different measurements will be briefly discussed and provide reference to in depth literature.

2.1 Frequency and phase response correction

The sampled impulseresponse of a system is $h(n)$. The Fourier transform will be:

$$H(f) = \sum_{n=-\infty}^{\infty} h(n) * e^{-j2\pi fn} = |H(f)| * e^{-j\measuredangle H(f)} \tag{2.1}$$

The amplitude response will be: $|H(f)|$, and the phase response: $\measuredangle H(f)$ The ideal response for an audio reproduction system would be to have a flat frequency and a linear phase response.

$$H(f) = \sum_{n=-\infty}^{\infty} \delta(n) * e^{-j2\pi fn} = e^{-j2\pi f} \tag{2.2}$$

$$|H(f)| = \left| e^{-j2\pi f} \right| = 1 \tag{2.3}$$

We see that if the impulse response had been a delta pulse, the frequency response would be flat across the entire spectrum. To get a flat spectrum through the audio spectrum, the pulse does not need to be a perfect delta pulse, but preferably as close as possible. The problem is that all reflections, from loudspeaker box and from walls and objects, and crossover filters introduce both frequency coloration and phase errors. Normal consensus of correcting any loudspeaker system and room has been to flatten frequency response using equalizing. Using an equalizer without taking into consideration the phase response this imposes on the system

also introduces phase errors and imprecise transient reproduction. Paper by Lipshitz, Popcock and Vanderkooy [10], discuss the audible effects of midrange phase distortions, and states that even small phase distortions are audible. This implies that to reconstruct a soundfield around a listener using multiple loudspeakers to create credible virtual sources (see next section), the phase response and alignment is critical.

2.2 Virtual sources and Ambisonics

Amplitude panning is commonly used in 5.1 surround systems, but works good only when sitting in the sweetspot (see Fig 2.1). In order to create credible virtual sources in an amplitude panning system, the listener must be positioned equidistant to the speakers in order to avoid an acoustical collapse.

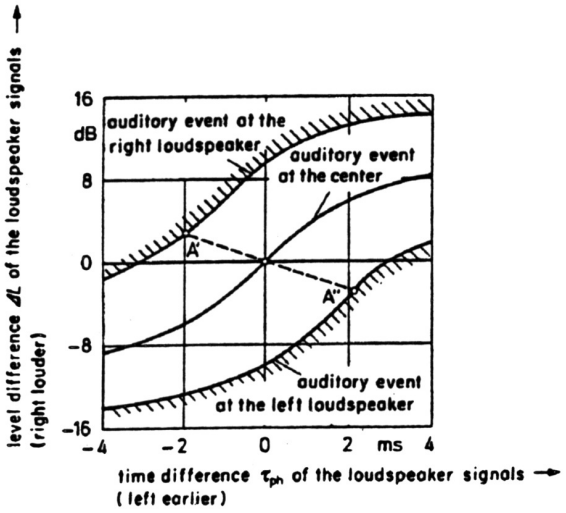
The listener to the right (see Fig 2.1(b)) experiences an acoustical collapse into the right loudspeaker because of the timedelay introduced by a nonsymmetrical position. This is why the center channel was originally introduced. One could introduce more loudspeakers to try to avoid this problem, but listeners in different positions will still not perceive the same acoustical origin (see Fig 2.1(c)). The perception of virtual sources in a standard stereo system depends on amplitude difference and time difference between the loudspeakers. The relation between these can be seen in figure 2.1(a).

To get a better localization, the system could use an Ambisonics encoder and decoder. The system investigated in this paper does not use Ambisonics directly, but is based on the same principles. The Ambisonics principle[1] is based on reconstructing the directivity of the soundfield around a point; the sweet spot, using spherical (or cylindrical in horizontal plane) harmonic functions to describe the soundfield through space[12]. With a multichannel system you could reconstruct a wavefront in the vicinity of the sweetspot to create the perception of a soundsource outside the speakers. The number of loudspeakers in your surround setup determines the order of Ambisonics that could be used, and the order determines the radius around the sweetspot the system can reconstruct the soundfield up to a certain frequency. A five channel surround system can support up to 2.order Ambisonics. Higher order Ambisonics[6] has not yet been commercialized due to the need of a large amount of loudspeakers, though it exists within academic circuits.

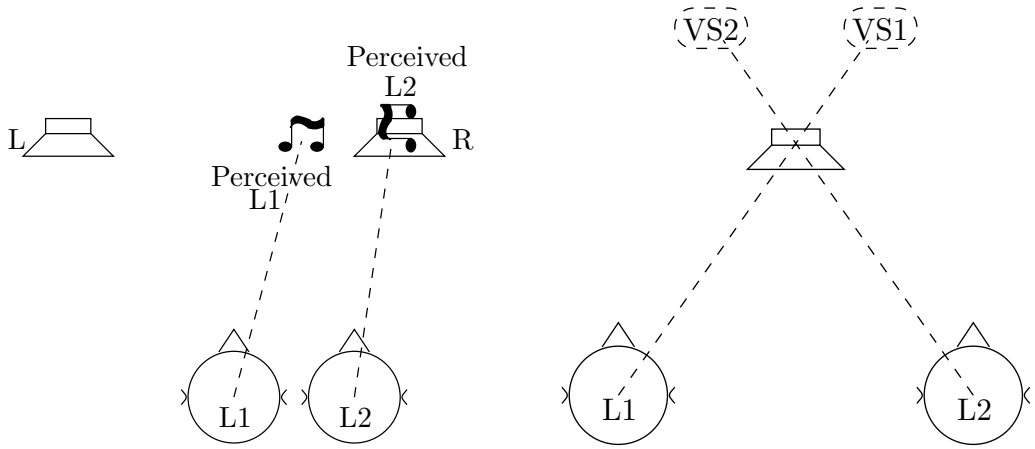
Equation 2.4 and 2.5 [2] states that five loudspeakers can only reproduce Ambisonics of order one in 3 dimensions, and order two in 2 dimensions.

$$L_{3D} \geq (N+1)^2 \tag{2.4}$$

$$L_{2D} \geq 2M+1 \tag{2.5}$$



(a)
Source location in stereo [5]



(b)
Perceived source

(c)
Problem with more speakers

Figure 2.1: Perception problem in amplitude panning

where L is the number of loudspeakers in the reproduction system, and $N = M$ is the order of Ambisonics. $M = k * r_0$ where $k = \frac{2\pi}{\lambda} = \frac{2\pi * f}{c}$ is the wavenumber, f the frequency, and c is the speed of sound. This shows that in a standard home theater system with a 5.1 system, only the first and second order can be used. In this paper only the horizontal plane will be discussed.

2.3 Reflection cancellation

Reflections cause frequency coloration and phase errors. A flattening of the frequency and phase response would imply that reflections also were compensated for. Early reflections can be compensated for with the process of digital deconvolution techniques. In practice this is the same principle used in noise cancellation. The loudspeaker will send out an inverse polarity copy of the reflected wavefront and they will cancel one another. When the reflections become many and turn into reverberation there is no longer a possible way to perform a functional deconvolution, and the use of linear phase equalization is much more efficient.[11]

The effect of a single reflection can be seen in figure 2.2, where we observe a strong combfilter effect on the signal with the reflection due to positive and negative interferences. The effect is repeated up through the spectrum, and the spacing between these interferences is related to the timedelay of the reflection.

2.4 Trinnov Optimizer

The Optimizer system consists of a sound processor to perform measurements and realtime audio prosessing, and a four capsule microphone. The Optimizer is inserted into the signalchain right before the systems amplifiers. The measuring microphone is placed in the listening spot and the machine will measure all channels and calculate inverse filters for flattening of the frequency and phase response from each speaker/amplifier/room response. Simple reflections will be cancelled to a certain extent. Loudspeaker positions will be compensated for by remapping signals to achieve correct localization according to a standardized 5.0 setup. This remapping method, and "moving" of the physical sources, relates to the functions of an Ambisonics[1] system. The soundfield around the listening position is calculated through the use of a Fourier-Bessel decomposition into a sum of spherical harmonic functions trough a remapping matrix that has a reference in i.e. ITU 775[3] (for a 5.1 system)[11]. This results in a correct reproduction of virtual sources even though the loudspeakers are not correctly positioned.

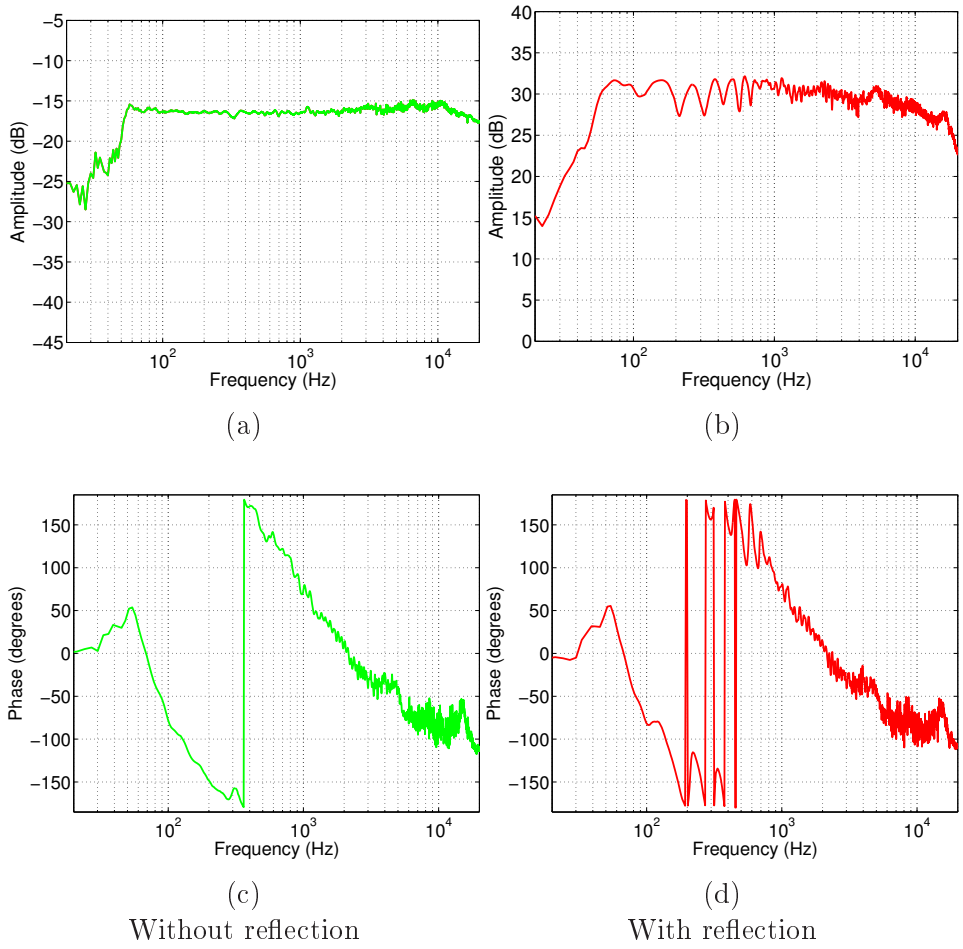


Figure 2.2: Frequency and phase response of a single speaker with and without a simple reflection. Decibel values are uncalibrated.

3

Measurement method

All measurements were performed in a former student project. Due to the complexity of some measurements, this information about how the measurements were performed is provided in this report as well. This chapter will show the different equipment and scenarios investigated in this report. This is necessary in order to understand the interpreted data. All measurements have been done in the NTNU acoustics anechoic chamber.

The systems that have been used for measuring are:

- Trinnov Optimizer, with calibrated four capsule microphone.
- WinMLS system, using single Norsonic microphone.
- WinMLS system, using 5 Norsonic Microphones on boom mounted on turntable.

The loudspeakers used are Dynaudio Acoustics BM6A, self powered loudspeaker. The different scenarios have been measured both with and without correction performed by the Optimizer. see figure 3.1

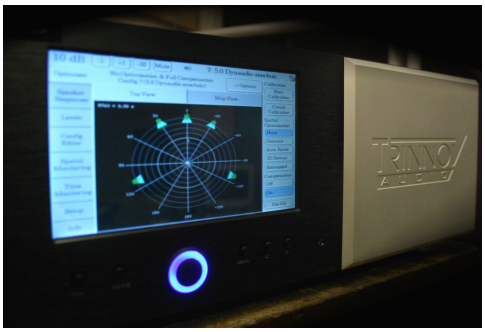


Figure 3.1: Trinnov Optimizer

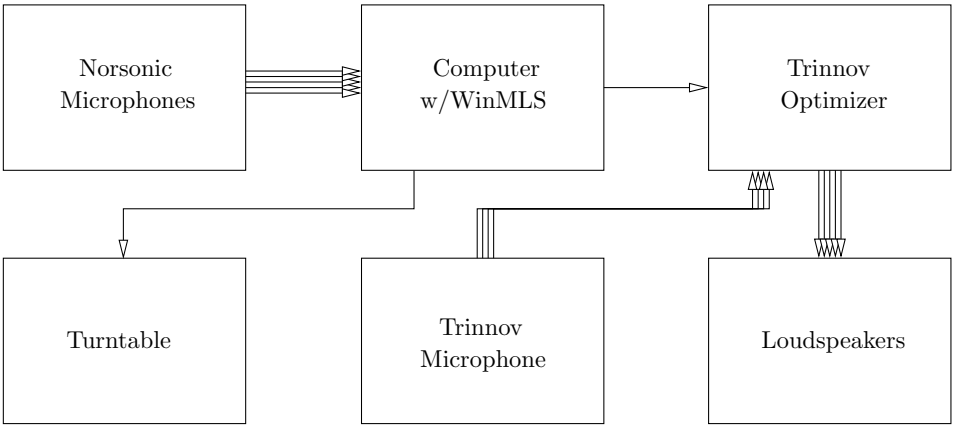
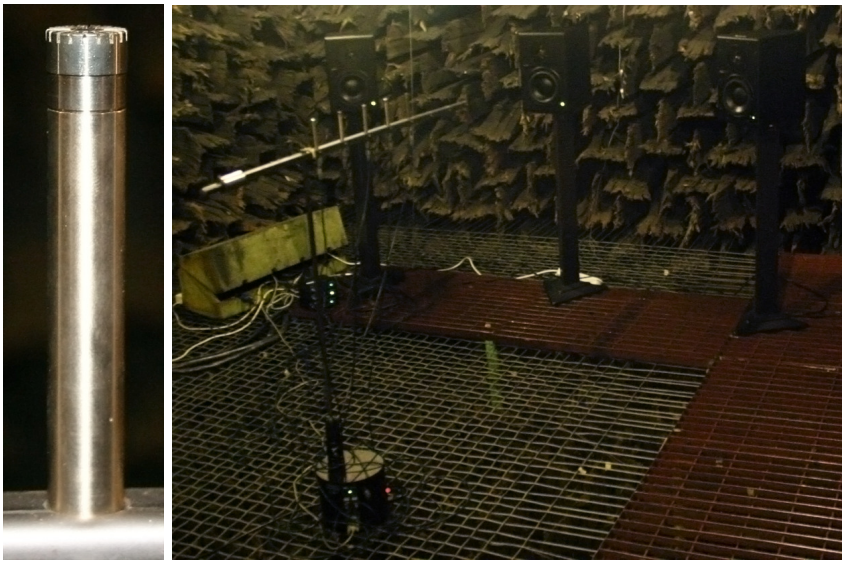


Figure 3.2: Blockdiagram of equipment

3.1 Using the Trinnov Optimizer

The Optimizer is inserted into the signal chain before the amplifiers, in this case before the self-powered loudspeakers (see blockdiagram 3.2). The four capsule microphone (see figure 3.5) is placed in the listening position facing towards the center speaker. Pressing the "Optimize" button on the Optimizer initiates a measuring sequence going through all speakers one at a time using an MLS signal to determine the impulse response from each and every one of the speakers. This procedure ensures two things: The Optimizer now knows the impulse response of each signal chain (all amplifiers and loudspeakers trough room), and the exact position of all the speakers in distance, and azimuth/elevation angles. The display of the Optimizer now shows both the optimal loudspeaker placements, and the position of the actual measured speakers. see figure 3.6. There are a couple of choices for



(a) Norsonic mic

(b) Setup with boom and turntable

Figure 3.3: Microphones, boom and turntable in anechoic room

correction: Compansation and Spatial optimization are the two main menus. Compansation turns frequency and phase flattening on and off. Spatial optimization presents the options: Distance, AutoRoute, 2d Remap, and Remapped. These determines wether to just correct distance to speakers in time and amplitude, or to remap the input signal to the actual speakers to correct the placement angle of the speakers.

3.2 Using the multiple microphone measurement setup

The Optimizer mic was placed in the listening position, and the "Optimize" sequence was performed. The Optimizer mic was now removed, and five Norsonic microphones were placed on a microphone boom in a straight line with a position of 6.25cm, 18.75cm, 31.25cm, 43.75cm, 56.25cm from the center of the microphone stand. see figure 3.8. Microphone number one is positioned closer to the stand to get a measurement close to what could be considered the radius of the listeners head. The radius of the area of recorded soundpressure was decided to be enough to see the effects of the system in the vicinity of the listening position. The distance between the microphones gives a spacial resolution of about $f = \frac{c}{\lambda * 2} = \frac{340}{0.125m * 2} = 1360Hz$ The microphone stand was mounted on a turntable positioned in the center of

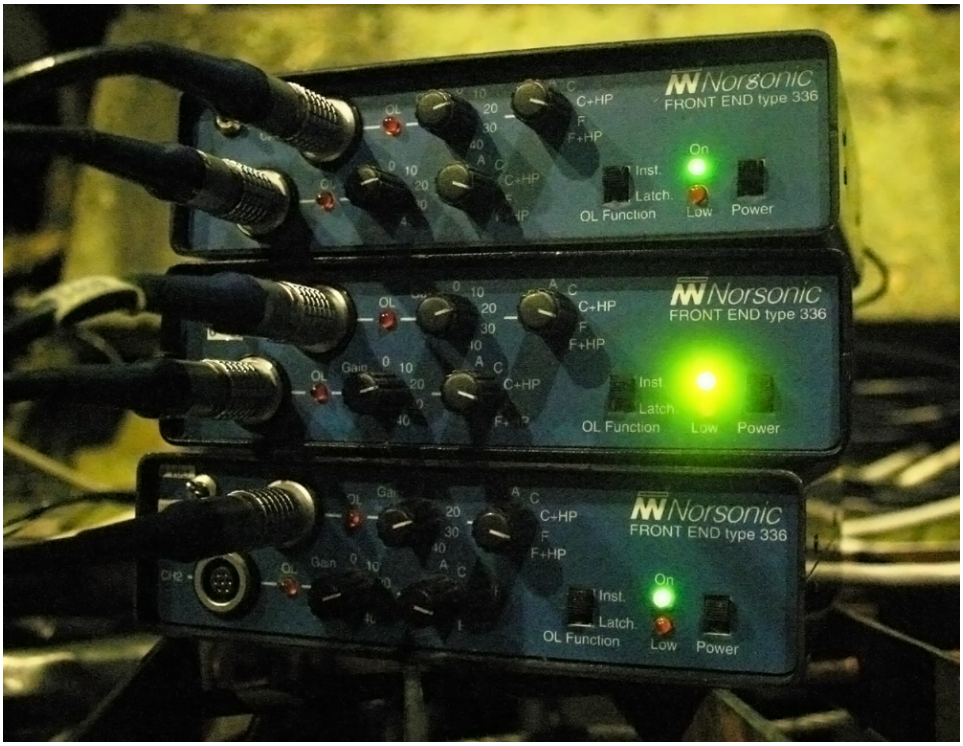


Figure 3.4: Microphone amplifiers

(a) (b)

Figure 3.5: Trinnov Optimizer microphone

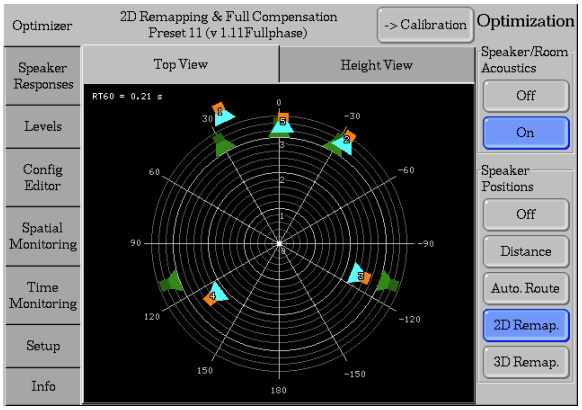


Figure 3.6: Loudspeakerpositions top view [11]

the listening position, and was able to rotate 360°. see figure 3.3(b).

The microphoneboom was rotated 360° with 3° increments to sample 5 microphones through 120 measured angles totaling 600 impulseresponses per measurement. See figure 3.7

3.3 Measuring scenarios

3.3.1 Phase and frequency response measurements

Sweetspot measurement

The measurements of phase and frequency response to provide background and reference to the validity of further results were done with a single loudspeaker in the anechoic room. This was done to minimize possible reflective objects and test the simplest possible environment. The speaker was first measured with the Optimizer system. Then a Norsonic microphone was placed right next to the Optimizer microphone capsules, and measured again with WinMLS. Position of loudspeaker relative to microphone is stated in table 3.1.

Speaker	*Distance*	*Elevation*	*Azimuth*
1	2.48m	-5.8°	-2.1°

Table 3.1: Measured position of loudspeaker

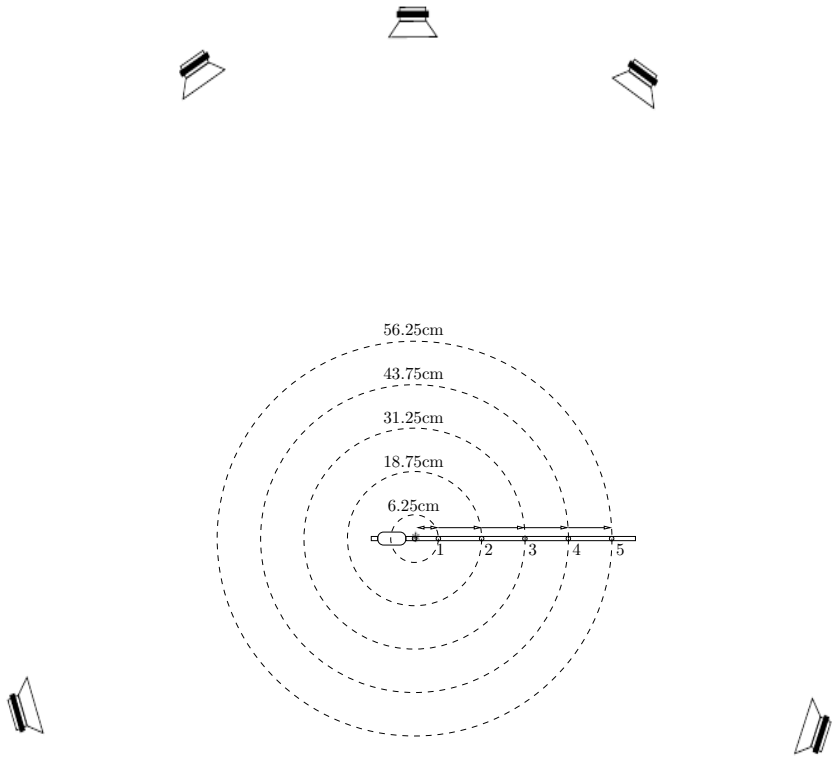


Figure 3.7: Microphone positions during one measurement

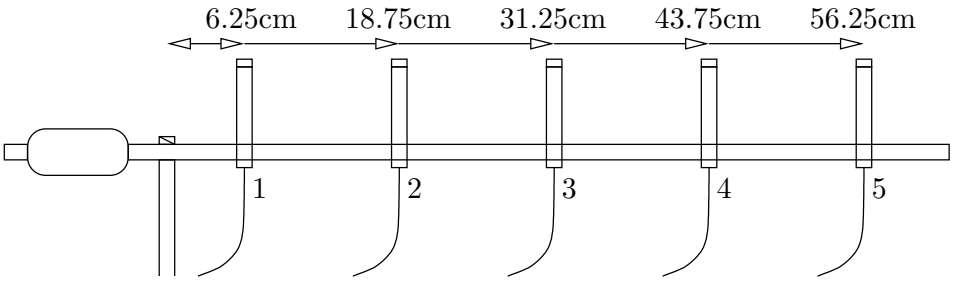


Figure 3.8: Microphone boom

Figure 3.9: Single loudspeaker measurements

Outside sweetspot

The measurements that provide information of the performance of the Optimizer outside the sweetspot were obtained using the microphone boom. See section 3.2.

3.3.2 Reflection cancellation

Measurements on the ITU 775 system were performed using the multiple microphone setup. See section 3.2. Two scenarios involving reflectors were investigated:

Backreflector

A reflector was placed behind the center loudspeaker to simulate a wall behind it (see fig 3.10). The measuring signal was played only through the center speaker. The measured soundpressure will indicate a direct sound and a delayed reflection causing the effects described in section 2.3. This was done to investigate the simplest kind of reflection caused by a single reflective surface behind the center speaker. The intention was to see what corrections the Optimizer was able to make in the area around the sweetspot.

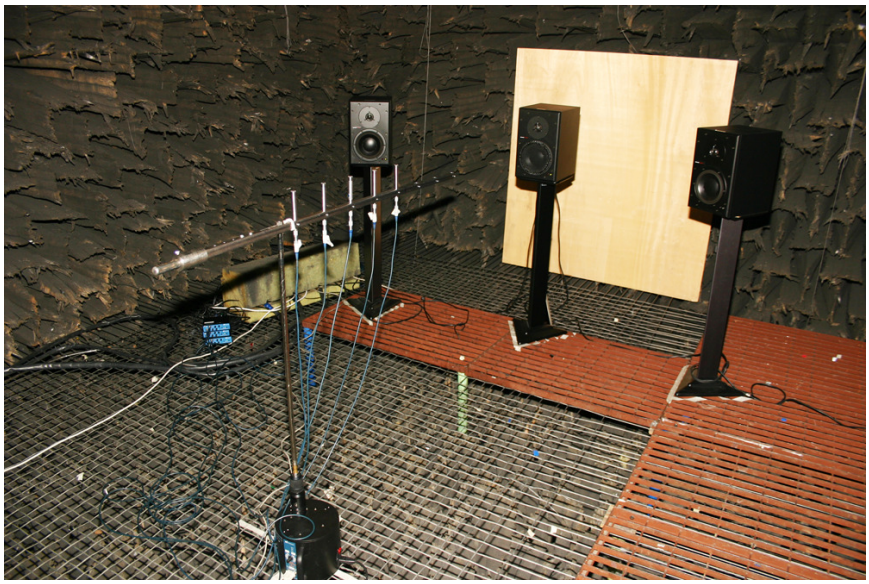


Figure 3.10: Frontreflector in anechoic room

Sidereflector

A reflector was placed on the side of the loudspeaker setup (see figure 3.11). The measuring signal was played only through the center speaker. The reflector will create a delayed reflection at an incidence angle different from the direct sound of the loudspeaker. This was performed because this kind of reflection will be much harder to cancel because its origin is not at the angle of any loudspeaker.

3.3.3 Loudspeaker placement correction and virtual sources

Five loudspeakers were placed perfectly according to the ITU 775[3] standard (see figure 3.12(a)). Sine sweep measuring signal was sent only to the rear right loudspeaker. Measurements were performed according to section 3.2. The right rear loudspeaker was moved (see figure 3.12(b)) and a new measurement was performed with the Optimizer to record the new loudspeaker positions. Another measurement (sec 3.2) was performed to record the soundfield from the loudspeaker in the wrong place. The Optimizer was now set to 2dRemap the signal back to its original position, which means it will try to create a virtual source at the original rear right loudspeakers position. New measurements (sec 3.2) were performed to record the corrected field.

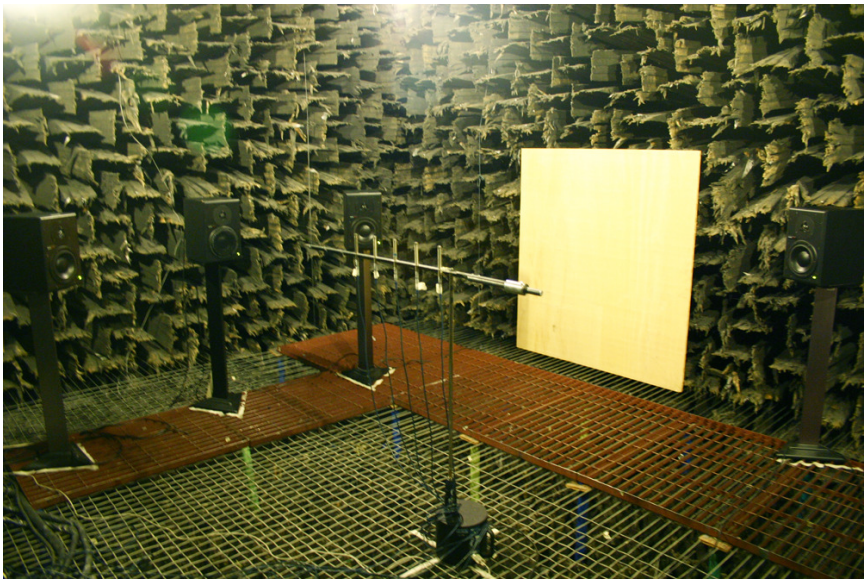
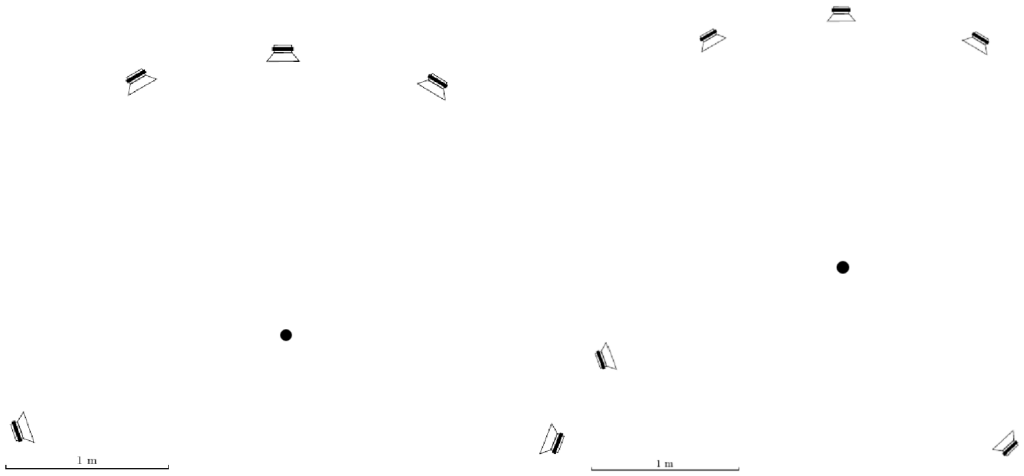


Figure 3.11: Sidereflector in anechoic room



(a)
ITU 775

Speaker	*Distance*	*Elevation*	*Azimuth*
1	1.84m	0.2°	29.9°
2	1.83m	−2.9°	−30.8°
3	1.77m	−2.5°	0.5°
4	1.75m	0.3°	109.5°
5	1.80m	−1.3°	−111.5°

(b)
Displaced surround right speaker

Speaker	*Distance*	*Elevation*	*Azimuth*
1	1.83m	1.7°	30.1°
2	1.82m	−2.1°	−30.8°
3	1.75m	−1.1°	0.4°
4	1.76m	0.5°	110.7°
5	1.67m	−1.7°	−137.0°

Figure 3.12: Trinnov Optimizer loudspeaker layout

4

Results

The results will be presented in this chapter. Some results may be similar to previously presented work, but provide substantial information for understanding and validity of further results. All performed measurements have been executed to further characterize the performance of the Trinnov Optimizer in an anechoic room, with single and multiple loudspeaker setups. All WinMLS measurements have been processed through Matlab in order to get these results. Short descriptions of of the Matlab prosessing will be mentioned in each section. WinMLS measurements were done using mentioned equipment. see list A

4.1 Background

4.1.1 Straightening the impulse response

Although it is implied that the impulse response must be close to a delta pulse in order to get an uncolored spectrum, it will be shown in figure 4.1 how the Trinnov Optimizer performs at this task. Parts of the code in B.1 was used, and can be viewed for detailed information about this process.

4.1.2 Frequency response measurements

The plots 4.2, show smoothed results for both Trinnov and WinMLS measurements. This states the similarities between the measurements, and are shown to validate the further use of only WinMLS measurements. Parts of the code in B.1 was used, and can be viewed for detailed information about this process.

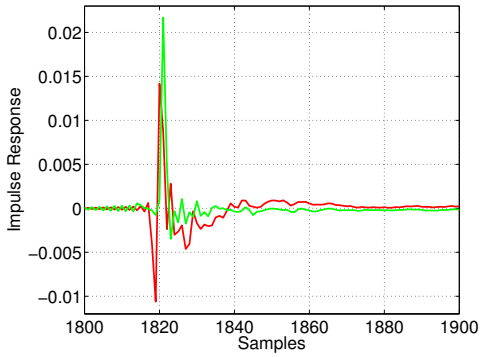
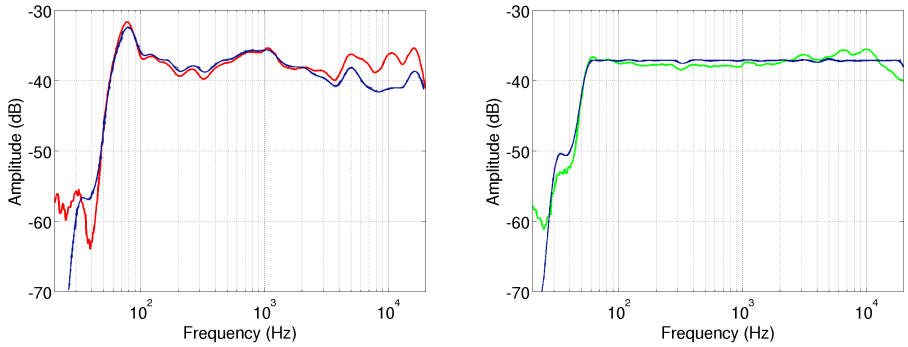


Figure 4.1: Straightening of the impulse response. Before; red, after; green.



(a)
Before correction
(blue:Trinnov)

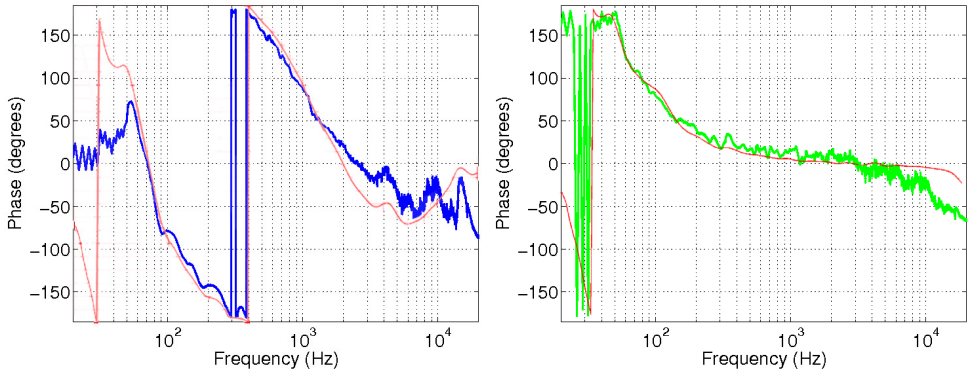
(b)
After correction
(blue:Trinnov)

Figure 4.2: Frequency response measurements using Optimizer and WinMLS systems. The data is smoothed, and decibel values are uncalibrated.

4.1.3 Phase response measurements

The measurements of the phase response on a single loudspeaker in an anechoic environment have been revised because of a frequency resolution error discovered in the matlab code. The new results show the correction of the loudspeakers phase response. To find the phase response of the loudspeaker, the initial(propagation) delay of the system must be removed. This has been performed in Matlab by using the slope of the total phase response [13]. This means; **Propagation delay has been removed when showing phase response data! All phase plots in this report include only the response of the loudspeaker and reflections that may follow.** The impulse response had to be upsampled to obtain the needed precision. Parts of the code in B.1 was used, and can be viewed for detailed information about this process.

These measurements were done using both the Trinnov Optimizer system with belonging calibrated microphone, and WinMLS system. The plots show Trinnov and WinMLS measurements before and after correction. see 4.3



(a)
Before correction
(red:Trinnov)

(b)
After correction
(red:Trinnov)

Figure 4.3: Phase response measurements using Trinnov Optimizer and WinMLS systems. Note that Trinnov measurements are smoothed, WinMLS measurements are not. Propagation delay has been removed.

4.2 The soundfield around the listening position

The plots 4.5 to 4.9 show the correction of the soundfield around the sweetspot at angles 0°, 90°and 180°. These measurements have been done using the five loudspeaker setup in the anechoic chamber. Audiosource was the center loudspeaker. These three angles were chosen just to see what the soundfield looks like at different positions around the sweetspot, also to get an idea of what was causing imperfections in the measurement setup. This can be seen by investigating small reflections that appear in the impulseresponses to try to determine what causes them.

The gathered data shows the behavior of the correction device in the area surrounding the sweetspot when there are no external interferences. This data 4.5 - 4.9 is obtained by using Matlab script B.1. $\frac{1}{6}$ octave band smoothing have been used on both frequency and phase responses to still see some detail, while this is also considered to be a greater resolution than what is perceived by the human ear.

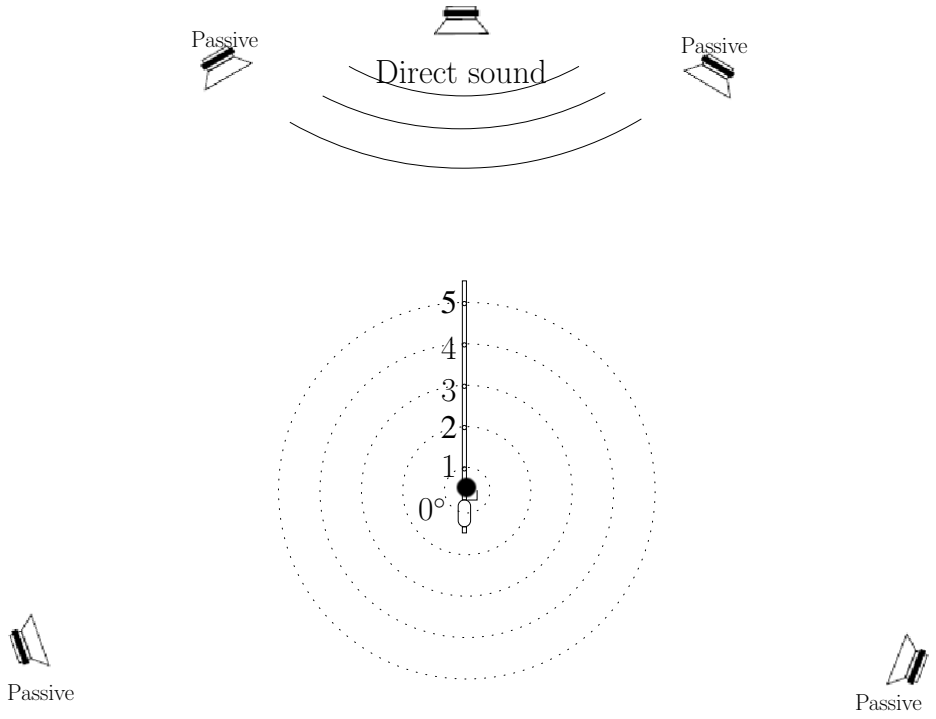


Figure 4.4: Situation for the plots on the following page.

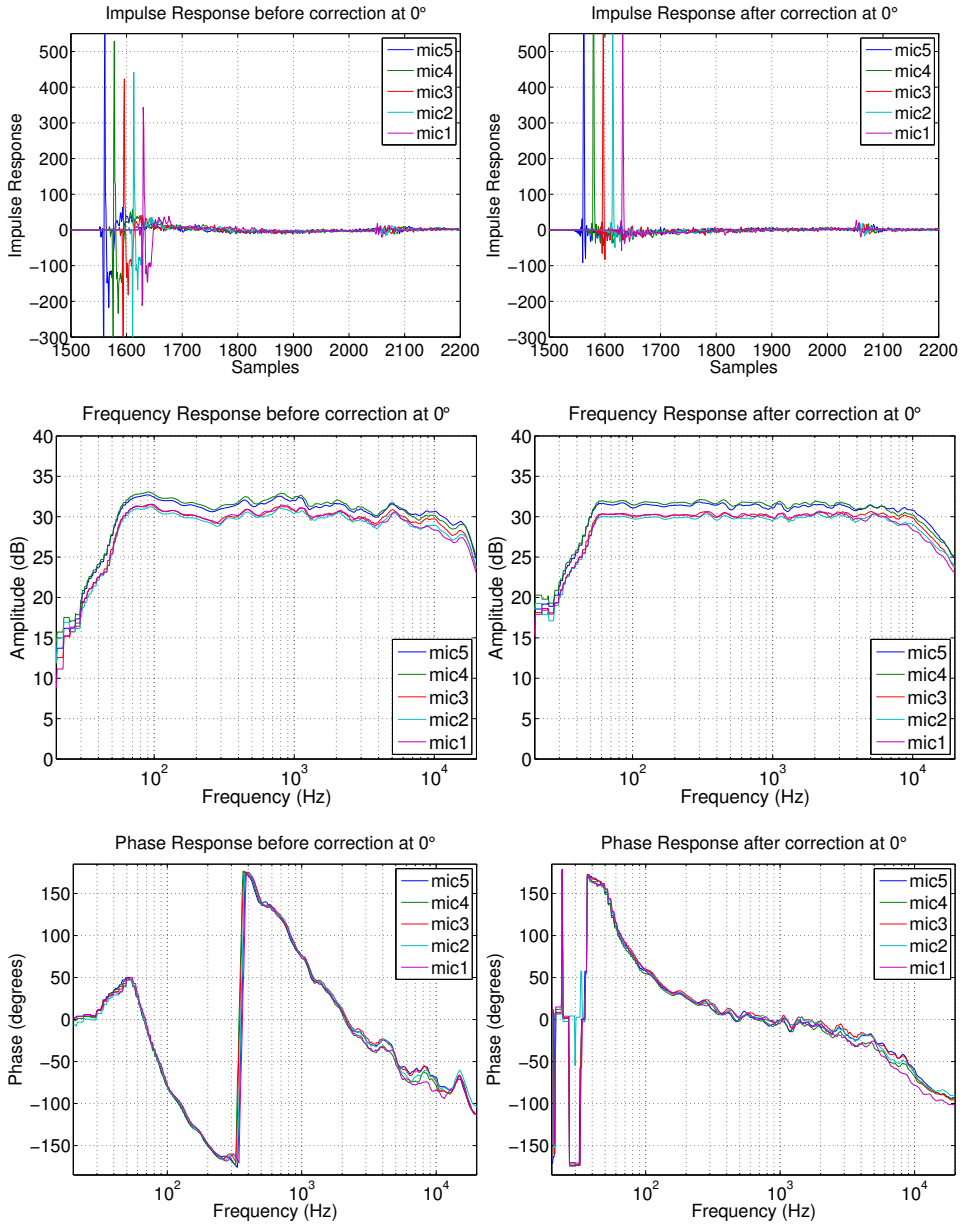


Figure 4.5: Responses around the listening position before and after correction at angle 0°. $\frac{1}{6}$ octave band smoothing have been used, and propagation delay has been removed in phase plots.

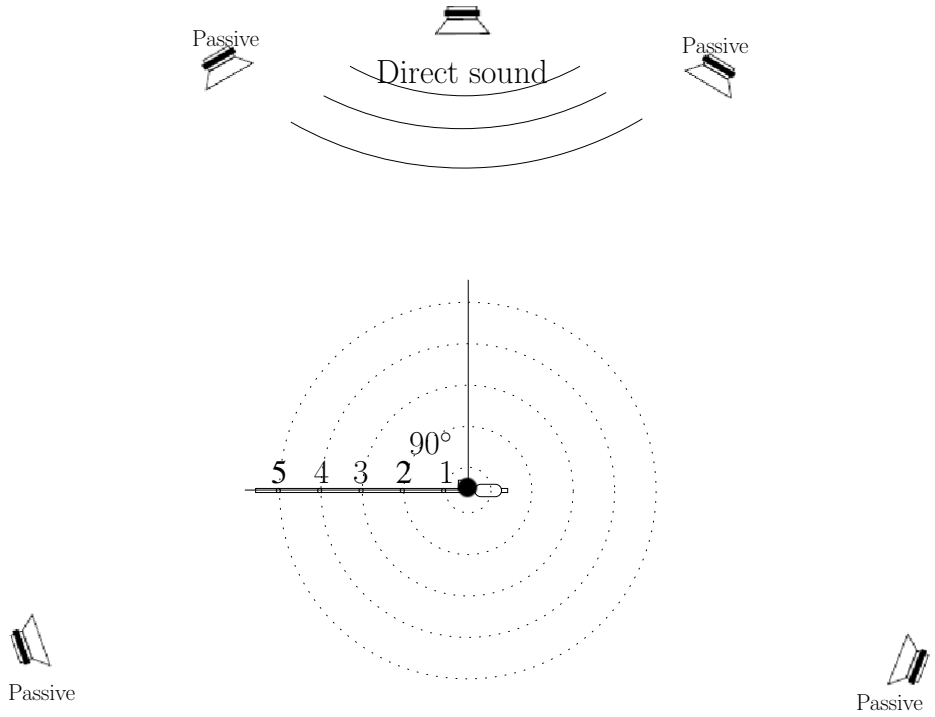


Figure 4.6: Situation for the plots on the following page.

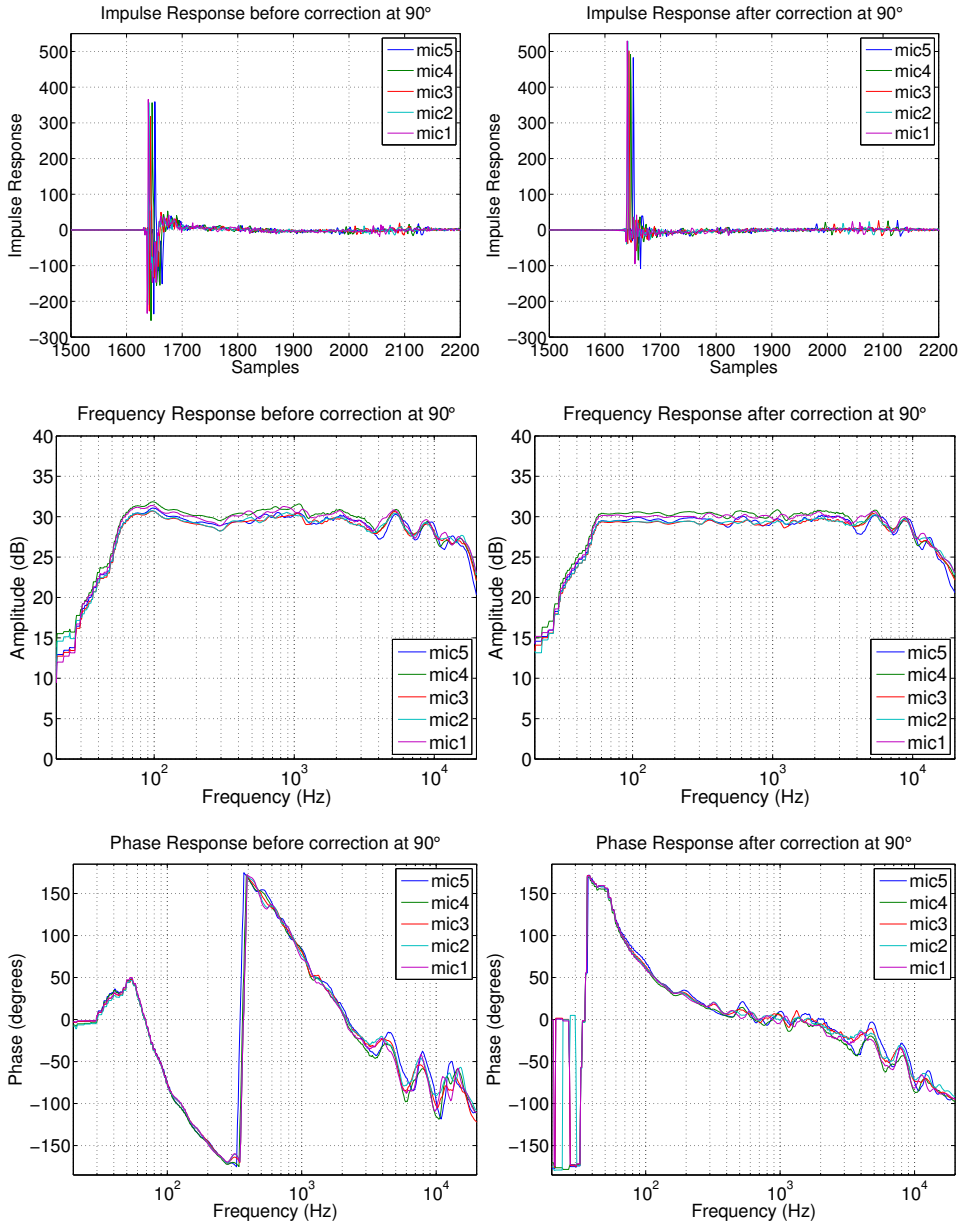


Figure 4.7: Responses around the listening position before and after correction at angle 90°. $\frac{1}{6}$ octave band smoothing have been used, and propagation delay has been removed in phase plots.

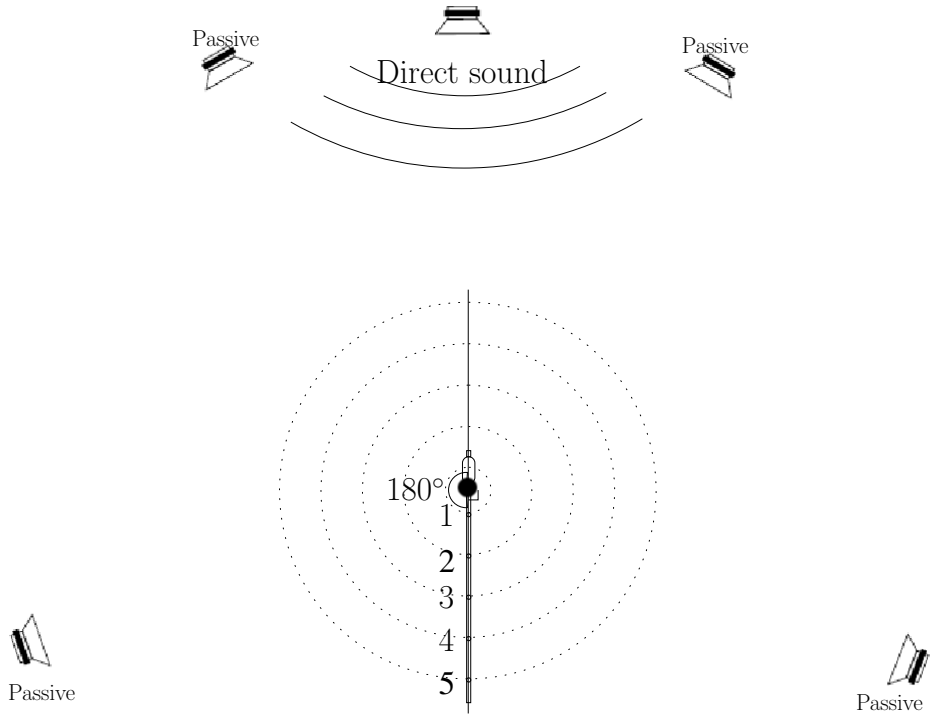


Figure 4.8: Situation for the plots on the following page.

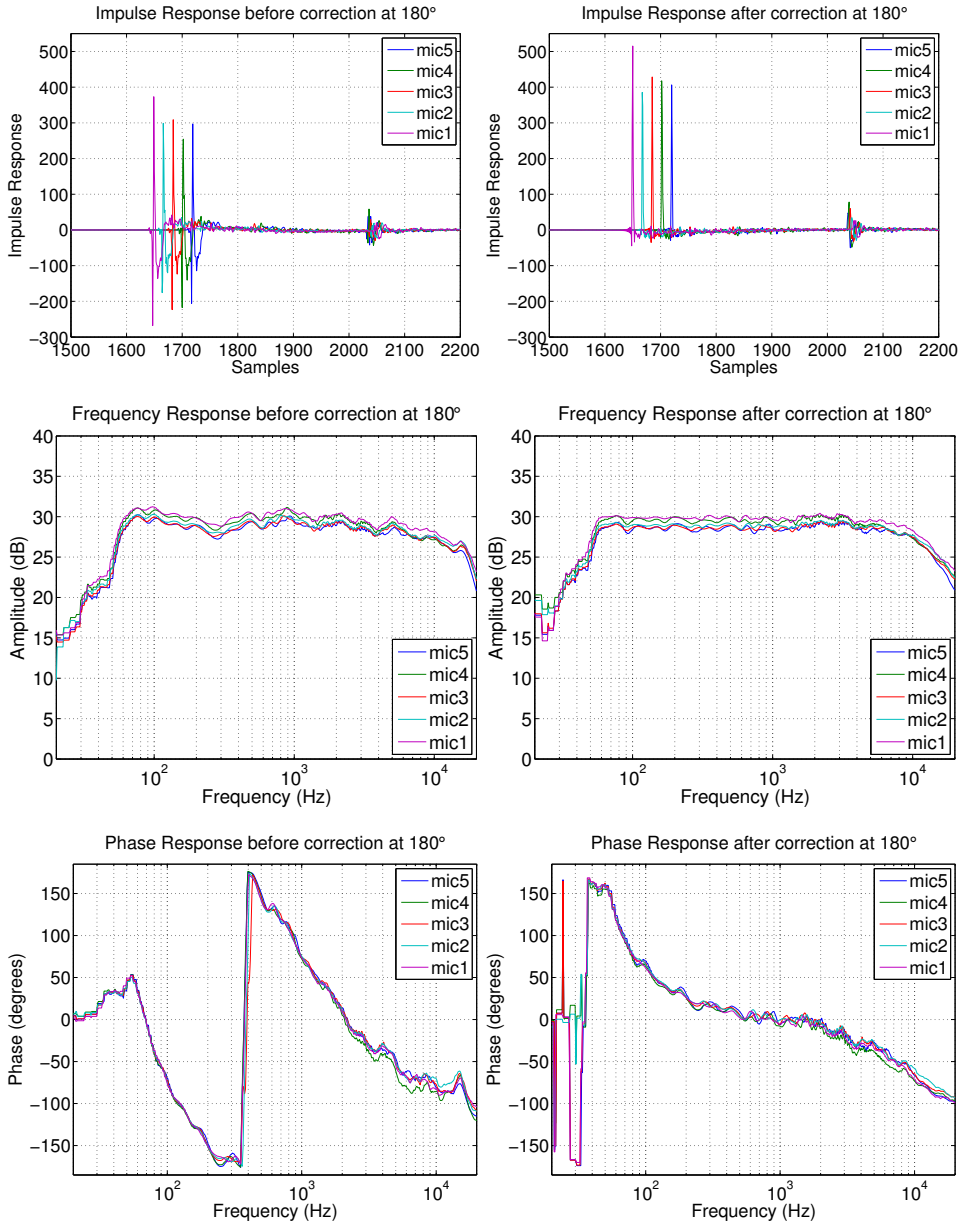


Figure 4.9: Responses around the listening position before and after correction at angle 180°. $\frac{1}{6}$ octave band smoothing have been used, and propagation delay has been removed in phase plots.

4.3 Video sequences

Trying to visualize a soundfield is not easy, but hopefully a few video sequences created will help enlighten and confirm the results obtained in the next chapters. The sequences were obtained from the measured impulseresponses and illustrate a delta pulse being played through a speaker and the soundfield propagating from it. An example of how the video layout appears is in figure 4.10. The top left visualization is a series of measurements containing some sort of error. The top right, is the same measurement corrected by the Optimizer. The bottom right is the same scenario without the error, thus the reference, or the way the soundfield is wanted. And the bottom left is the difference between the corrected scenario and the reference scenario. It must be noted that the most interesting of the four visualizations is the bottom left one. This indicates the difference between a scenario where the wrongful parts of the soundfield has been corrected and the desired soundfield.

The script in appendix B.2.1, gives information about which data is going to be processed by the video scripts. It calls on two functions, one that organizes data into big matrixes and creates the axis vectors B.2.2, and one that makes the video sequences by adding one plot at a time into an avi object B.2.3. The last mentioned script plots the 600 values of the impulseresponses over the measured area sample by sample, and puts the plot into an avi sequence. This means that at a given time/plot, the index of all the 600 impulseresponses will be the same. When these plots are played back through the avi file it is possible to see how the pulse propagates from the loudspeaker through the measured area.

The videos are from the front and side reflection scenarios as well as the misplaced loudspeaker scenario. The names of the videos will be stated in the respective sections. The videos have been tested and found working on VLC media player, a freeware media player for Windows and Linux.

4.4 Investigation of Reflection Cancellation

It has been shown that the Optimizer was able to cancel the effects of a reflection up to 300Hz [13]. The reflection investigated was a simple reflection from behind the center loudspeaker, and the results were only seen from the microphone closest to the center of the listening position. We will now take a look at more measurements from the backreflection scenario, and add another case, the sidereflection. This data was obtained by using a script similar to the one in appendix B.1.

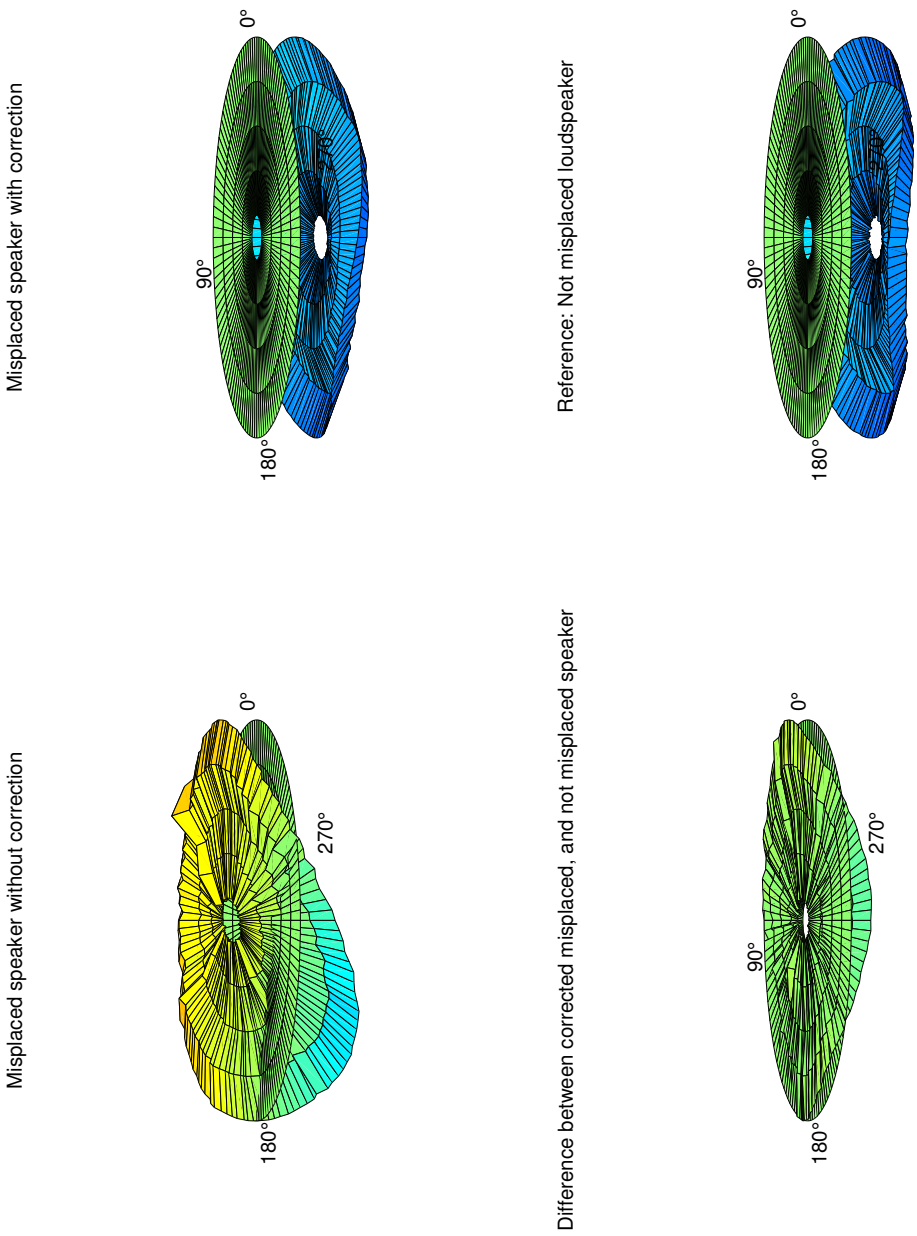


Figure 4.10: Layout example of the movie sequences.

4.4.1 Frontreflection

A wooden board was placed behind the center loudspeaker, see figure 3.10, and impulse responses were recorded through this loudspeaker. The received audio signal at the measuring points was a combination of direct audio from the center loudspeaker, and a reflection from the front. The results can be seen in figures 4.12 to 4.16. The video sequence that belongs to these measurements is called: "Frontreflection scenario BW300Hz 10fps.avi". BW300 means that the data is downsampled to 300Hz bandwidth. The video should be used to illustrate the findings in this chapter.

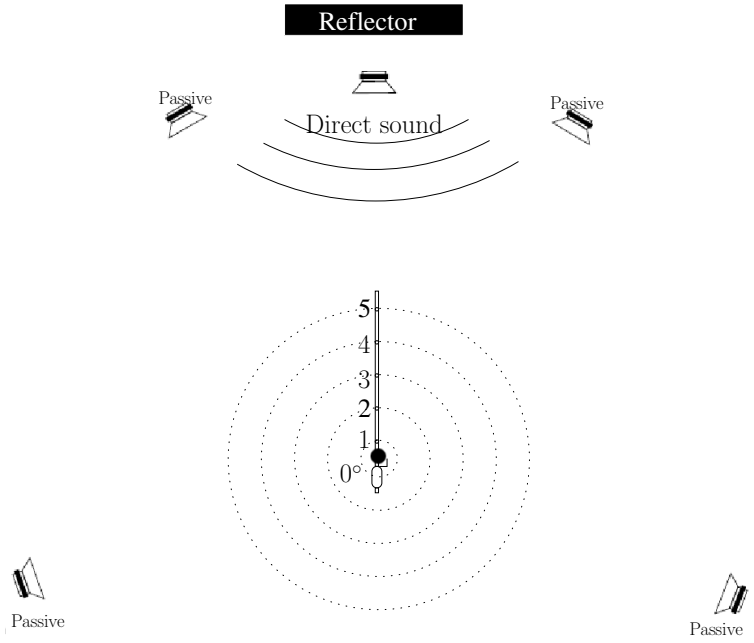


Figure 4.11: Situation for the plots on the following page.

4.4.2 Sidereflection

A wooden board was placed at one side of the loudspeaker setup. See figure 3.11. The received audio signal at the measuring points was a combination of direct audio from the center loudspeaker, and a reflection from the side. The results can be seen in figures 4.18 to 4.22. The measurement at 180° had to be replaced by the measurement at 177° because of a measurement error of mic 4 and 5. The video sequence that belongs to these measurements is called: "Sidereflection scenario.." BW300Hz or BW150Hz 5fps.avi. BW300 or BW150 means that the data is downsampled to 300Hz or 150Hz bandwidth. The video should be used to illustrate the findings in this chapter.

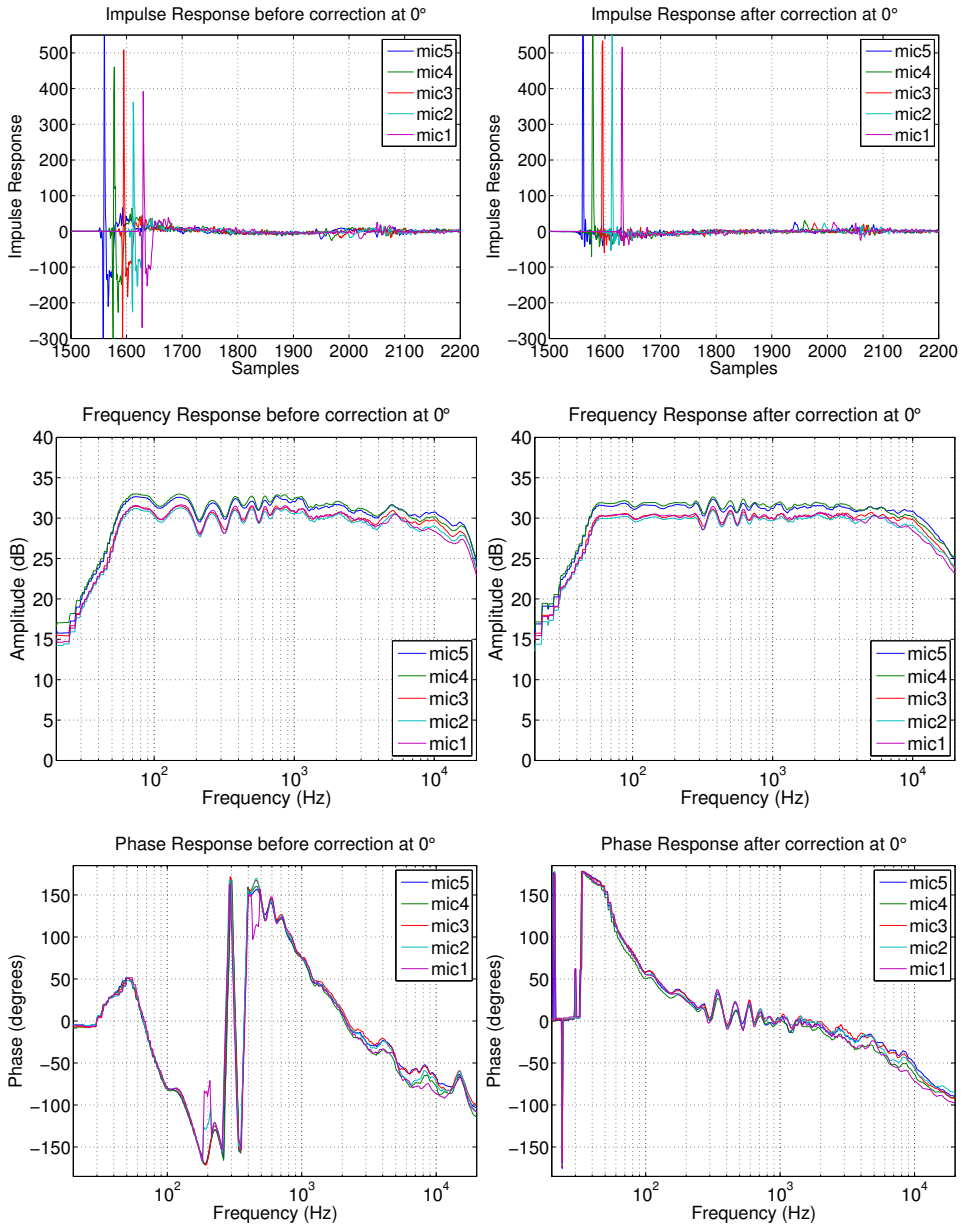


Figure 4.12: Before and after frontreflection correction at angle 0°. $\frac{1}{6}$ octave band smoothing have been used, and propagation delay has been removed in phase plots.

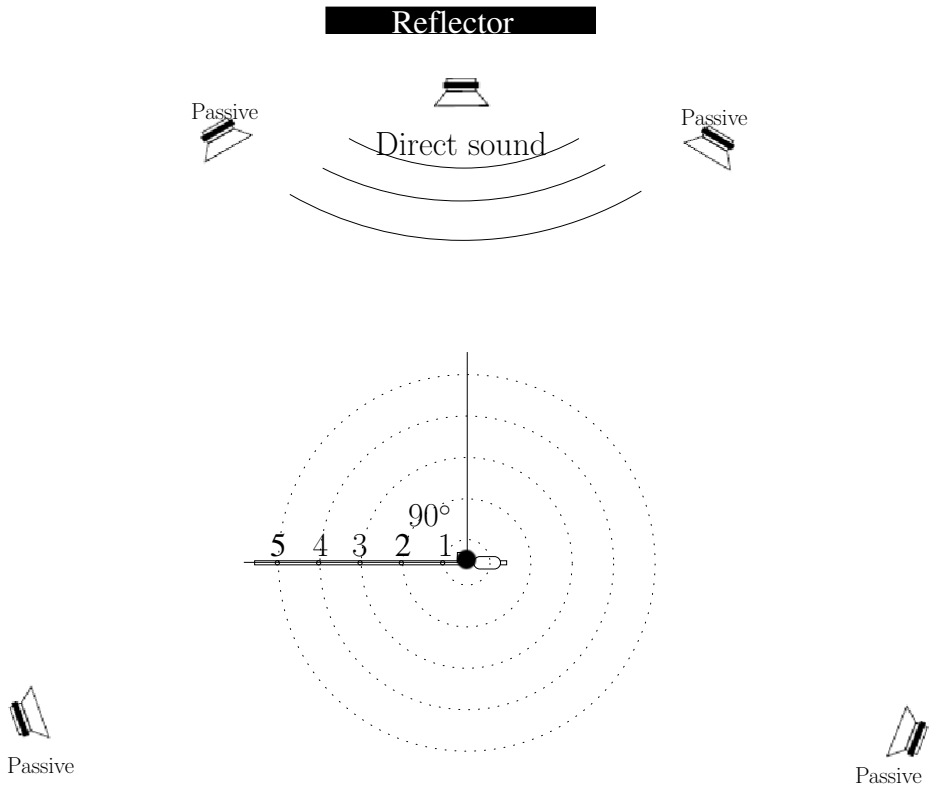


Figure 4.13: Situation for the plots on the following page.

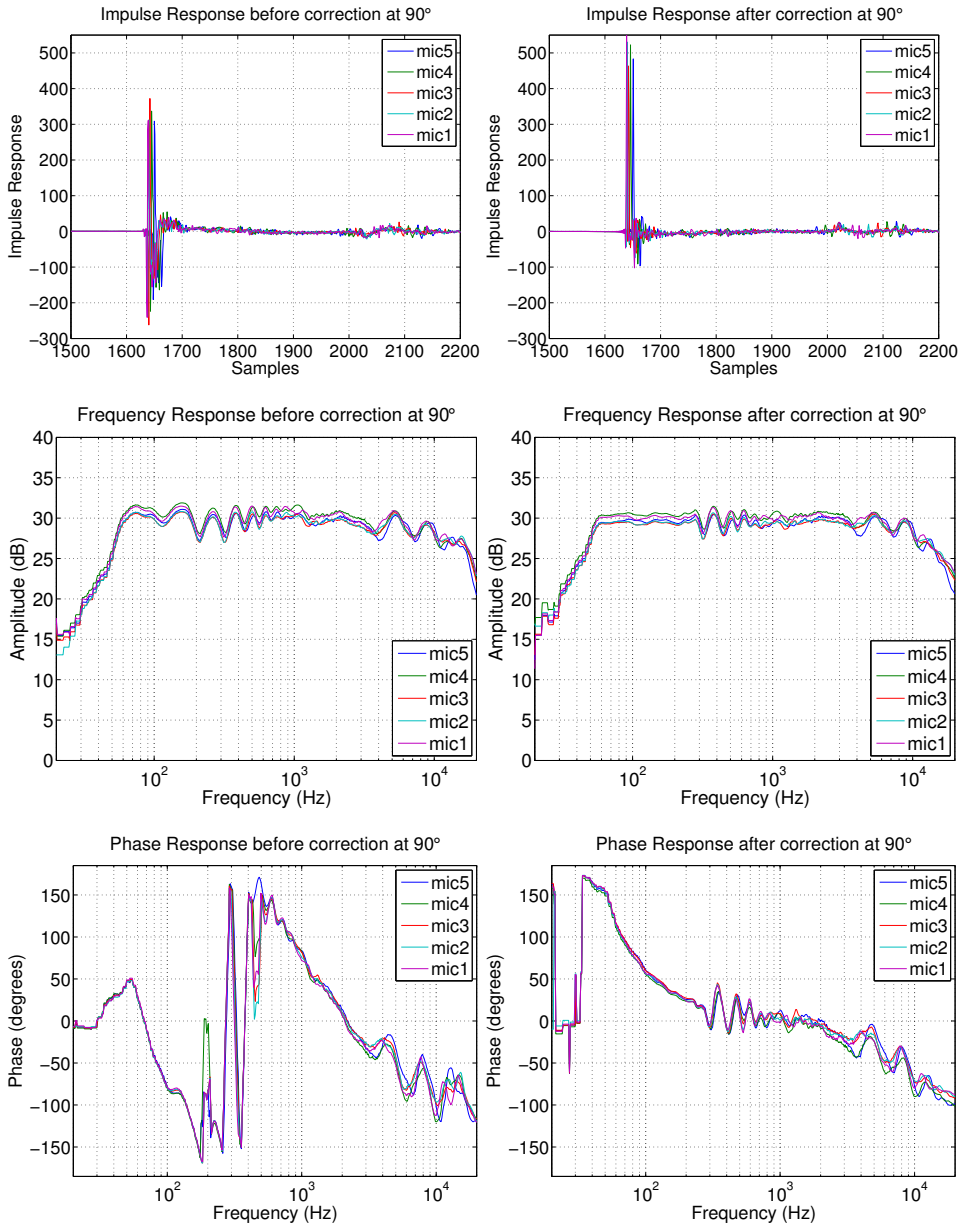


Figure 4.14: Before and after frontreflection correction at angle 90°. $\frac{1}{6}$ octave band smoothing have been used, and propagation delay has been removed in phase plots.

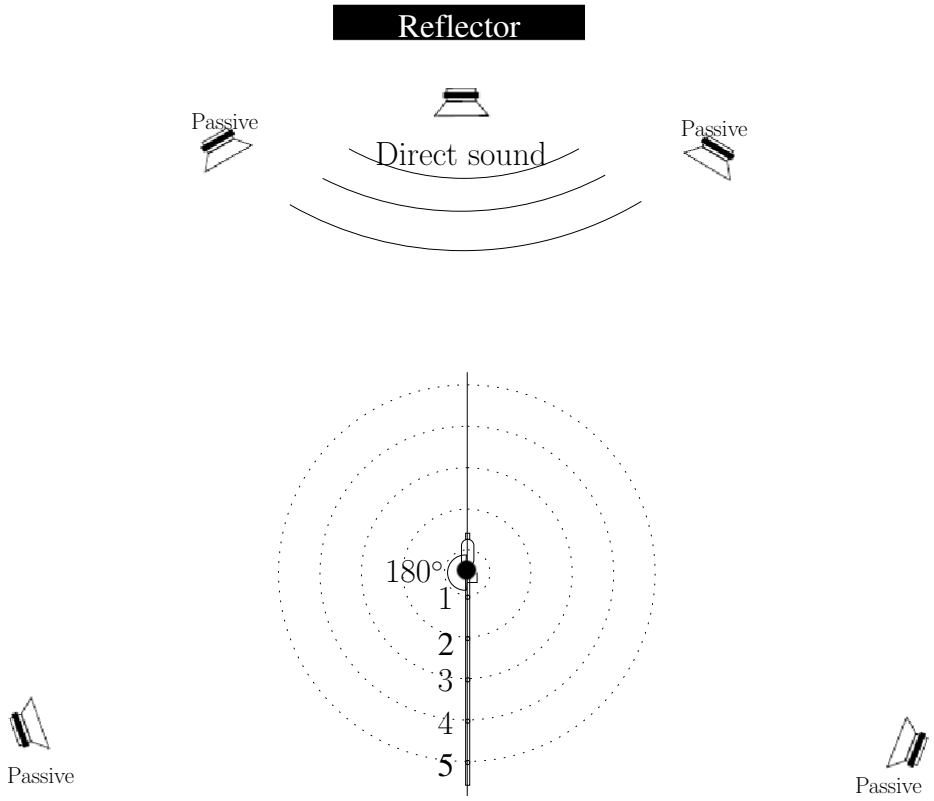


Figure 4.15: Situation for the plots on the following page.

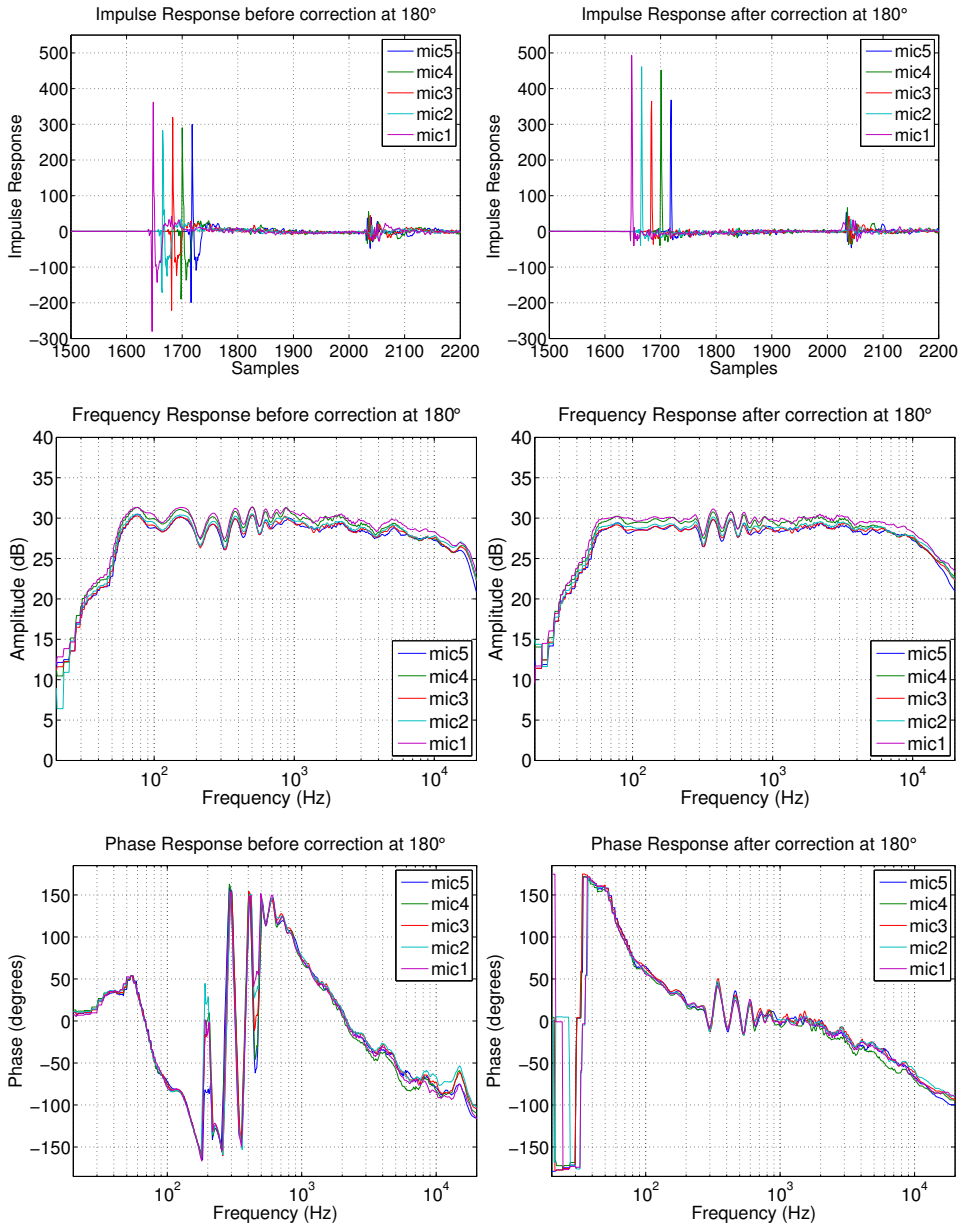


Figure 4.16: Before and after frontreflection correction at angle 180°. $\frac{1}{6}$ octave band smoothing have been used, and propagation delay has been removed in phase plots.

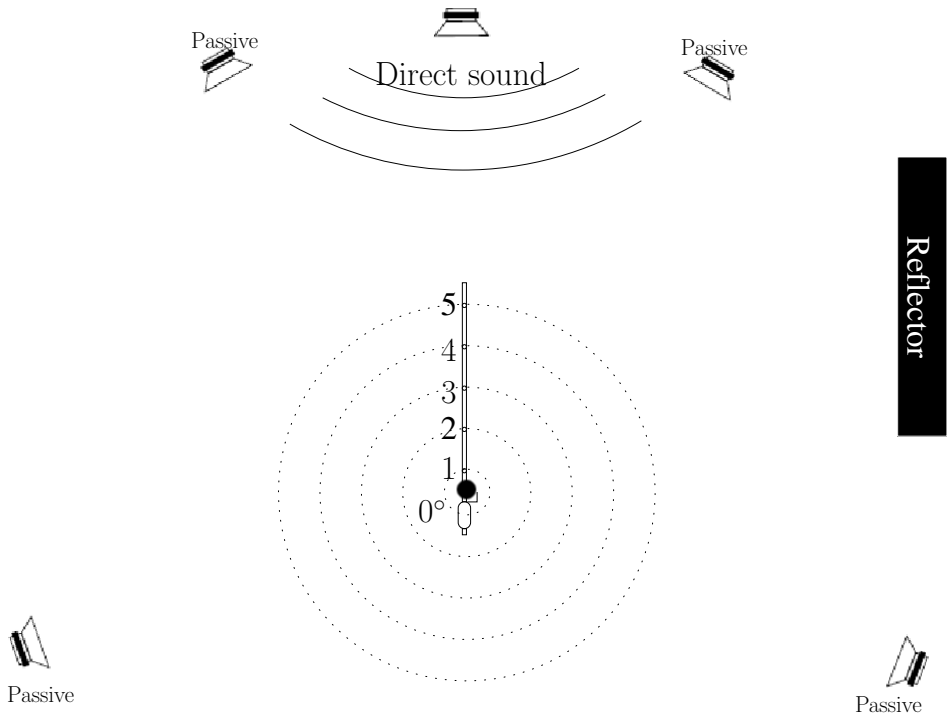


Figure 4.17: Situation for the plots on the following page.

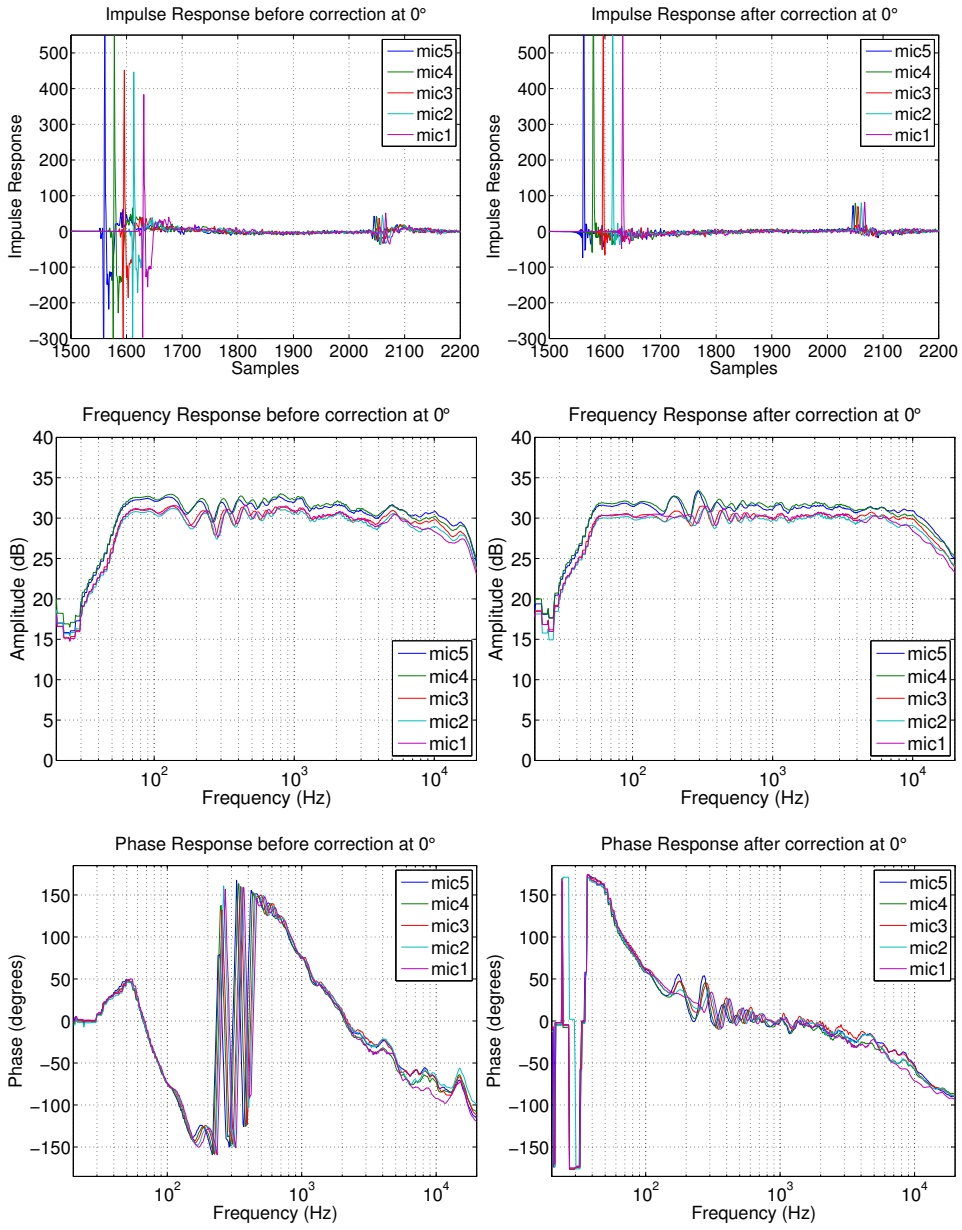


Figure 4.18: Before and after sidereflection correction at angle 0°. $\frac{1}{6}$ octave band smoothing have been used, and propagation delay has been removed in phase plots.

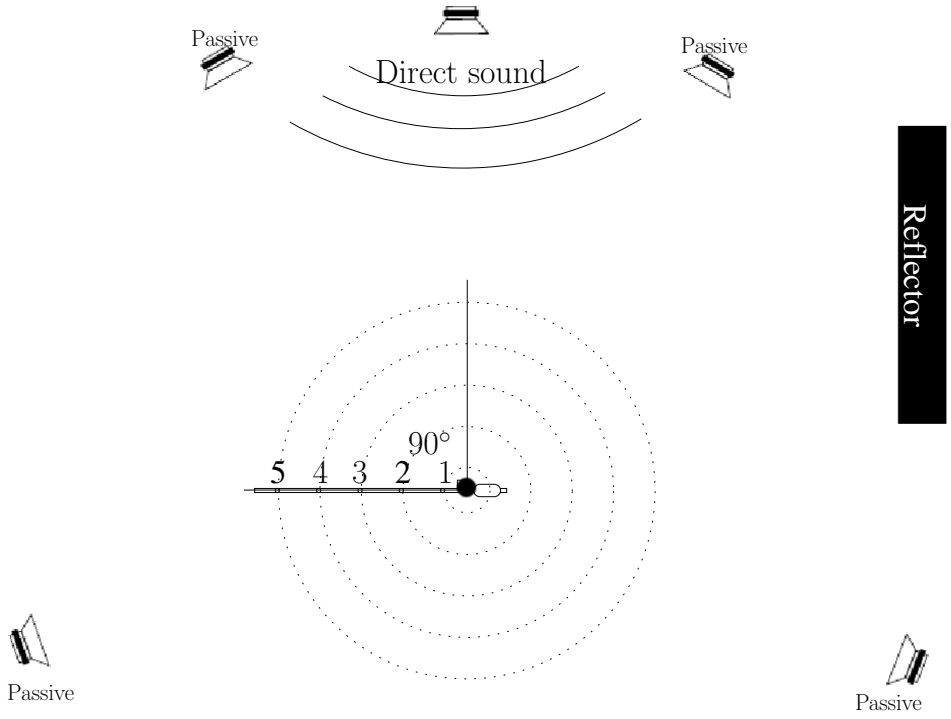


Figure 4.19: Situation for the plots on the following page.

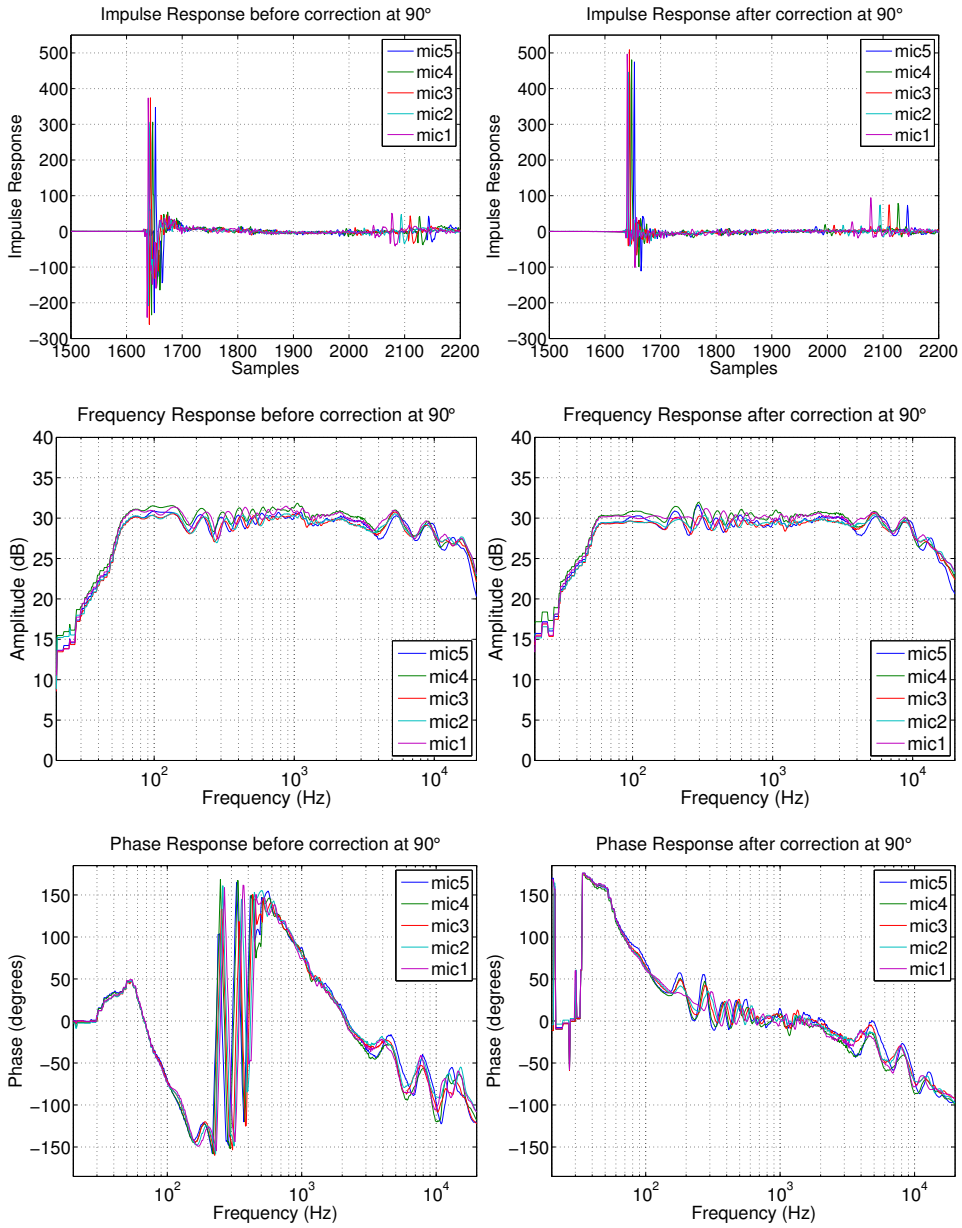


Figure 4.20: Before and after sidereflection correction at angle 90°. $\frac{1}{6}$ octave band smoothing have been used, and propagation delay has been removed in phase plots.

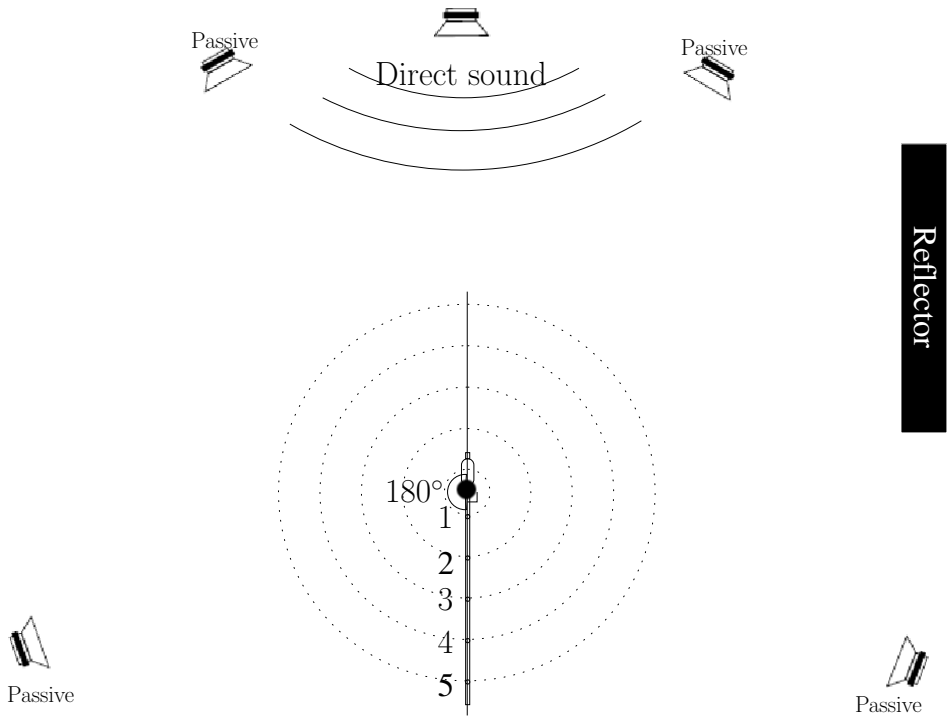


Figure 4.21: Situation for the plots on the following page.

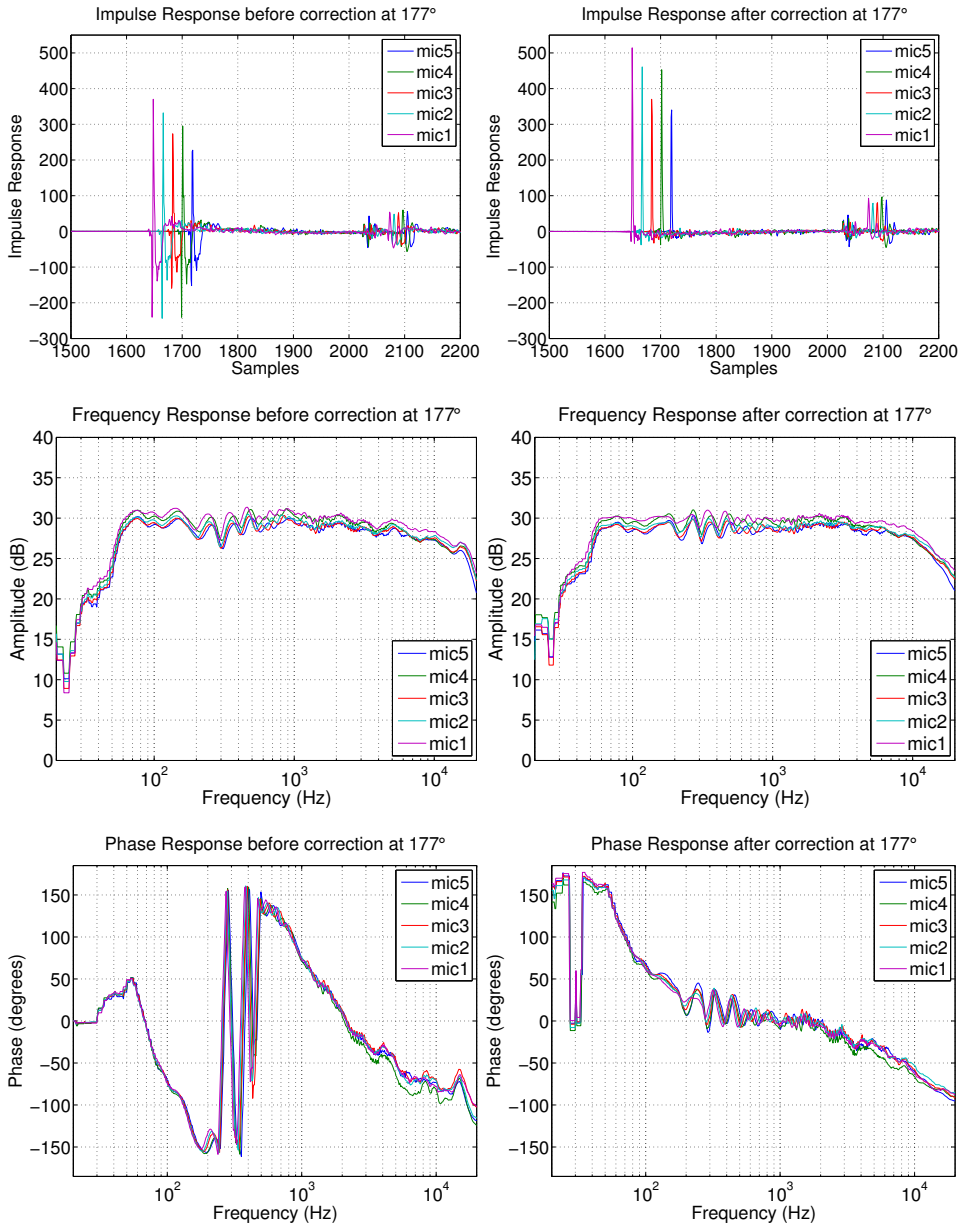


Figure 4.22: Before and after sidereflection correction at angle 177°. $\frac{1}{6}$ octave band smoothing have been used, and propagation delay has been removed in phase plots.

4.5 Virtual sources and perceived angle of acoustical origin

The Trinnov Optimizers ability to create virtual sources was tested by physically moving one of the five loudspeakers away from the ideal position. It was then set to remap the source back to its original position. See fig 3.12. This was done because it is often difficult, impossible or impractical to place loudspeakers in a studio or a home theater system according to the ITU-775 [3] standard. Functionality of a remapping device would be highly practical in every studio and home theater to enhance the performance of the system and still use practical loudspeaker placements. The video sequence that belongs to these measurements is called: "Virtual sources scenario.." BW300Hz or BW150Hz 5fps.avi. BW300 or BW150 means that the data is downsampled to 300Hz or 150Hz bandwidth. The video should be used to illustrate the findings in this chapter.

4.5.1 Finding perceived angle

The following figures were created in Matlab by an algorithm that was made to see at which angle each microphone perceived the acoustical origin. In the figures 4.23 to 4.26, each scenario is explained by abbreviations: comp - frequency/phase compensation, opt(2dRemap) - optimization to use remapping matrix to generate virtual sources. The different markers in the figures indicate where the different microphones "perceive" the acoustical origin.

To get results from these measurements it was necessary to lowpass the measured impulseresponses in order to get a bandwidth limited signal with reasonable performance expectancy from the Optimizer. By lowpassing the impulseresponses it became apparent that it was difficult to determine the details of the propagation delay using the method described in the phase measurements section. Knowing that a microphone moving in a circle will have a propagation delay following a sinus pattern, a sinusoidal curve fit was used to find the the point where the propagation delay was the smallest. This algorithm is based only on time delay, and finds at what angle the sound appears from first. The signal was resampled and the angle of the microphoneboom which had the lowest propagation delay was found. This angle was considered to be the angle of which the origin of the source would appear.

The full bandwidth measurements use the slope of the phase response to find the time delay. Details can be found in the script in appendix B.6.1. The lowpassed measurements used the sinusoidal curve fitting method described earlier. Details about this method can be investigated in the script in appendix B.6.2.

This has been measured for full bandwidth and lowpassed to 375Hz,

187Hz and 93Hz. The data from the 93Hz were discarded because of severe degradation due to heavy downsampling.

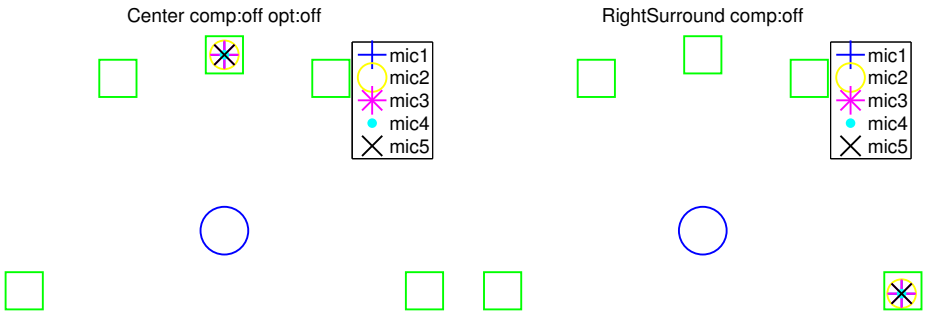


Figure 4.23: Perceived soundsource playing from the center speaker and right surround speaker.

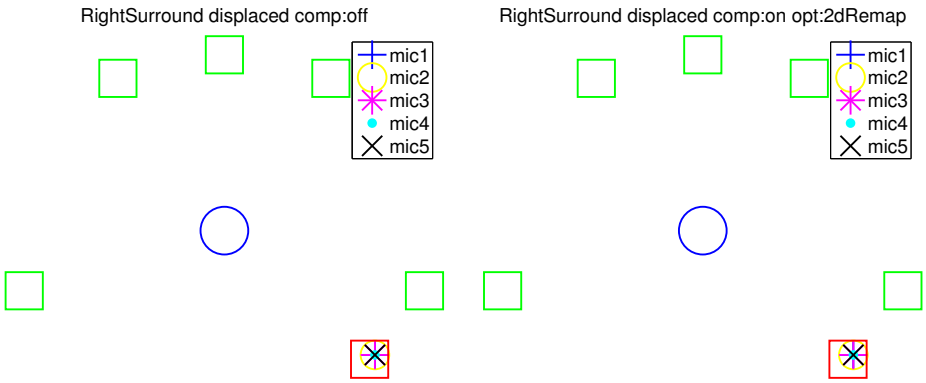


Figure 4.24: Perceived soundsource playing from the displaced right surround speaker. Full bandwidth.

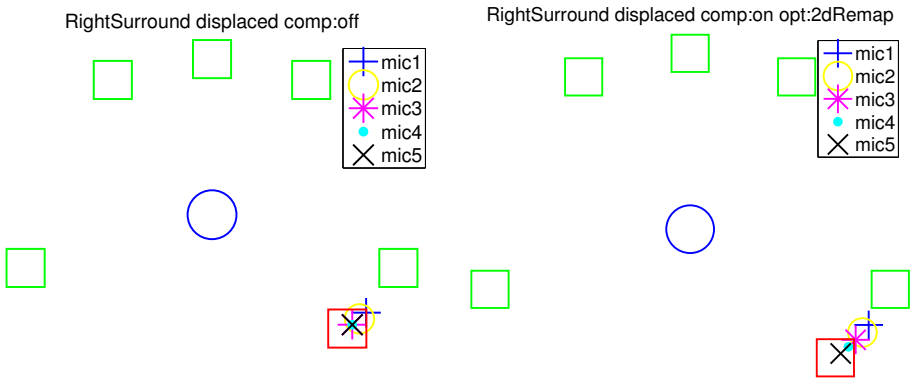


Figure 4.25: Perceived soundsource playing from the displaced right surround speaker. 375Hz bandwidth.

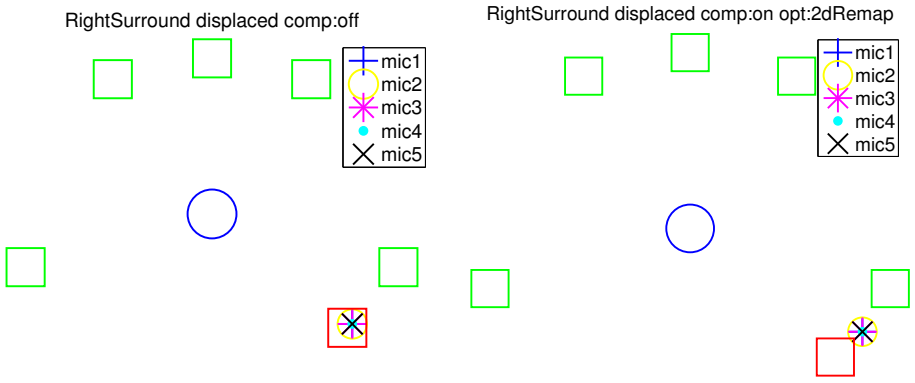


Figure 4.26: Perceived soundsource playing from the displaced right surround speaker. 187Hz bandwidth.

4.5.2 Frequency response error estimate

The plots 4.27, show the error in frequency response over the measured area around the listening position. The different figures show several different lowpassed scenarios to illustrate the performance of the reconstruction technique. The selected frequencies are natural instances of the sampling frequency(48kHz) being divided by two: 750Hz, 375Hz, 187Hz, and 93Hz. Expected theoretical performance is stated in table 4.1.
The error is calculated from:

$$\text{Freq Response Error} = 10log_{10}\left(\frac{\text{Frequency Response}}{\text{Frequency Response Reference}}\right) \quad (4.1)$$

"Frequency Response" is the frequency response of the system measured with the right surround loudspeaker out of position and optimized using 2dRemap optimization. The "Reference" is the system measured with the right surround loudspeaker in the correct position with frequency and phase correction on. This means that the plots should show a value of 0 if measurements were ideally equal. In the ideal scenario the reproduced soundfield around the listening position should be an exact copy of the physical soundfield from before the loudspeaker was moved.

The script for performing this is presented in appendix B.6.3.

Mic	*Radius*	*Theoretical frequency reconstruction range*
1	6.3cm	860Hz
2	18.75cm	288Hz
3	31.25cm	173Hz
4	43.75cm	124Hz
5	56.25cm	96Hz

Table 4.1: Theoretical frequency reconstruction limit of a 2D 2.order Ambisonics system.

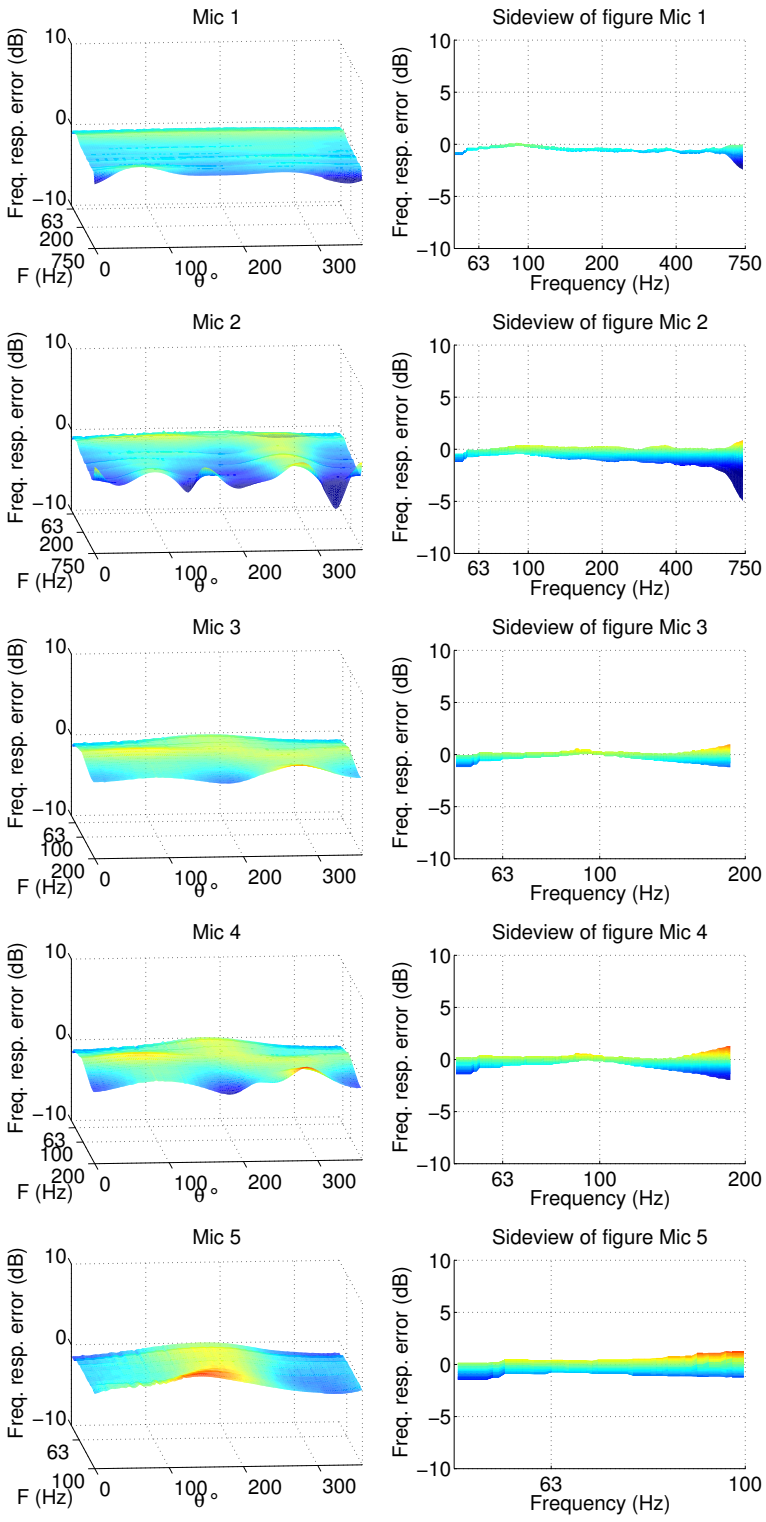


Figure 4.27: Error estimate. Note that the frequency axis are different from mic to mic.

5

Discussion

The Trinnov Optimizers performance has been tested on room correction issues outside of the sweetspot in four different cases. Correcting one loudspeaker, one loudspeaker with a reflection from the front, one loudspeaker with a reflection from the side, and a loudspeaker with the wrong position. The measurements have been performed earlier with the use of WinMLS. The results from these measurements will be discussed in this chapter.

5.1 Background measurements

It can easily be seen that the correction has a tremendous effect on the impulse response. The Optimizer performs well in cleaning up unwanted effects and imperfections in the loudspeaker. This implies the effects that was seen in the previous section of a pretty flat frequency and phase response. This will in theory mean that within certain limits it will not matter what kind of loudspeaker you own because the properties of the loudspeaker is determined by the impulse response. If the impulse response can be corrected to look the same regardless of what kind of loudspeaker you have it will mean they sound no different either.

The background measurements provide reference to prove validness of the WinMLS data. It is easy to see that the frequency responses are very similar. The measurements by the Optimizer and WinMLS are almost identical up to 4kHz (see figure 4.2). The difference above 4kHz is believed to be caused by a measurement error. The Optimizer microphone was not removed during WinMLS measurement, and the reflection from it may have caused some high frequency interferences. The low frequency has been cleaned up, and the lower frequency range of the loudspeaker has in fact improved greatly. The low end has been pushed a little bit down to give even better low frequency

reproduction. The errors caused by the anechoic room not being perfectly anechoic below 100Hz is handled very nicely by the Optimizer.

The phase measurements are now pretty identical (see figure 4.3), and most of the differences are believed to be caused by the fact that the Optimizer measurement is smoothed, and the WinMLS measurement is not. It was not considered a priority to develop a smoothing algorithm that takes phaseshifts into consideration. The differences around 300Hz, a couple of phaseshifts in the WinMLS measurements, are caused by the smoothing of the Optimizer. The plots now show a significant improvement of the measured phase response, and the Optimizer has compensated for a lot of the phase delay of the high frequencies of the loudspeaker and made the response flatter above about 100Hz. The overall phase delay has improved, and phase ripple has become smaller in the higher frequencies. It has been shown that the detection of the propagation delay worked perfectly and that all phaseresponses presented later in the results are purely the response of the loudspeaker and the room.

5.2 The soundfield around the listening position

The soundfield by the five microphones has been investigated at three angles: 0°, 90°, and 180°. At a first glance over the measurements at the three different angles (figure 4.5 - 4.9) one can see that the impulseresponses(IR) have been severely straightened. The immediate attention then goes to the small reflections later in the IR. The measurements have been done in an anechoic chamber, and reflections should be reduced to a minimum. By taking a look at IR at 0° and 180°, it can be observed that the reflections arrive at approximately the same time at all microphones. This would imply that whatever reflection that causes this effect must be symmetric over a line drawn from the center speaker and down through the center of the sweetspot. By looking at the IR at 90°(fig 4.7), it can be seen that the order of the pulses received imply a reflection from each side. The direct sound comes almost simultaneously to all five microphones, but then the reflections appear in a sequence: microphone 5-4-3-2-1-1-2-3-4-5. This confirms a theory of a reflection from both the rear loudspeakers. The effect is illustrated in figure 5.1.

$\frac{1}{6}$ octave band smoothing have been used on both frequency and phase responses. It can be seen that at 0° 4.5, the reflection is not as dominant as in the 90° 4.7 and 180° 4.9 case. The frequency response is more affected by these reflections at 90° and 180°. This can be explained by the strength of reflection at 180° is greater than at 0°Ålso at 90° the reflections are distributed over time and thus more significant interference effects occurs. The size of the loudspeakers prevents interferences in the low end of the spectrum because they are acoustically small compared to the low

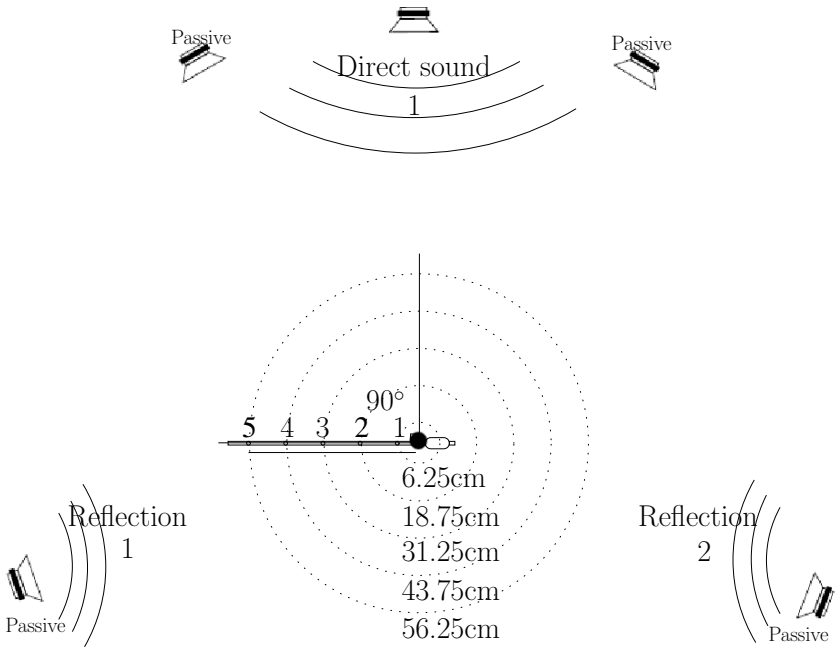


Figure 5.1: Reflections caused by the loudspeakers in the setup in anechoic room

frequencies.

The overall phase delay is improved in the speaker at all of the angles. Some high frequency phase ripple has been cancelled in the 0° and 180°cases. The phase ripple in the 90° case is still quite significant in the higher frequencies.

5.3 Investigation of reflection cancellation

Again the same three angles have been investigated.

5.3.1 Frontreflection

Reflector in the front: 4.12 - 4.16. It can be seen that the impulse responses have been straightened and cleaned.

0°: In figure 4.12, the effect of the reflector can be seen around samples 1900-2000. The reflections from the rear speakers appear right after the one from the reflector. The reflection is inverted and positive through the correction process. It can be seen on the frequency response that the Optimizer performs well below 300Hz and above 800Hz, and that it performs equally in all 5 microphone positions. The same can be said about the phase response. Significantly decreased phase delay from about 200Hz, and some successfully removed phase ripple above 2000Hz. The problems seen in the frequency response can also be observed in the phase response in the area between 300Hz and 800Hz, where there is some significant phase ripple.

90°In figure 4.14, the effect of the reflector can be seen around sample 2020. About the same sample that microphone 1 received the reflection in the 0°case. All microphones in this case receive the reflection at about the same time as for the direct sound. The reflections from the rear loudspeakers appear all around the received pulse from the reflector. The same problem area can be seen here as with the 0°case, that there is some ripple in both the frequency and phase response between 300 and 800Hz, and the problems observed without a reflector at all can be found in the higher frequencies here as well. This is believed to be caused by the reflection from the rear loudspeakers. The phase response has become flatter after the correction, and the performance of the Optimizer seems the same in all 5 measured positions.

180°: In figure 4.16, the effect of the reflector can be seen around samples 2250-2100. This produces about the same results as the 0°(fig 4.12). It has the same problem areas, and the same performance of the correction. The only difference is that the reflection from the reflector has a longer arrival time.

As it can be seen for all investigated angles, reflections are more or less cancelled under 300Hz. The video sequence also shows that this works quite

perfectly for all angles around the sweetspot at all the measured data points. The lower left video sequence in "Frontreflection scenario BW300" avi file

5.3.2 Sidereflection

Reflector on the side: 4.18 - 4.22. It can be seen that the impulse responses have been straightened and cleaned. The Optimizers algorithm for deconvolution will by default try to deconvolve an early reflection with an arrival time up to: 1000ms(up to 50Hz), 70ms(up to 150Hz), 10ms(up to 500Hz) and 1ms(up to 4kHz). The sidereflection time delay is at the limit of 10ms after the direct sound from the loudspeaker.

0°: In figure 4.18, the effect of the reflector can be seen around samples 2050. Since the reflector stood outside the loudspeaker setup, it seems that the time delay is pretty similar for the reflector and the reflection from the rear loudspeakers. Thus they can be hard to tell apart in the impulse response. Here there are some performance differences. The Optimizer does not deconvolve as well any more. It can be seen that the difference between the measured positions now is starting to become visible. Microphone 1,2 and 3 has a fairly flat frequency response up to 200Hz while microphone 4 and 5 only stays flat up to about 100Hz. The high frequencies seem to be taken care of, but the problem area has been moved down a couple of hundred Hz. The same counts for the phase response which actually shows more phase ripple when moving away from the sweetspot up to about 300Hz.

90°: In figure 4.20, the effect of the reflector can be seen around samples 2070-2140. The same tendencies can be seen here as for the 0°case. The frequency response seems to degrade when stepping further away from the sweetspot. High frequency interferences are present as in the other 90° measurements, most likely due to the different arrival time of the reflections from the rear loudspeakers and the frequency content of these. The same effects can be observed in the phase response.

177°: The measurement in figure 4.22 was supposed to be at a 180°angle. Because of a measurement error in microphone 4 and 5 in this data, it was replaced by the measurement at 177°. The effect of the reflector can here be seen between sample 2060 and 2110. From the frequency response it can be seen that microphone 1 has been corrected up to about 280Hz, but the other microphones have a rapid degrading whereas microphones 2 trough 5 has been corrected only up to 180Hz. In the phase response one can see a more gradual degradation from mic 1 to 5 between 200 and 300Hz. Again the problem area with ripple in both the frequency and phase response occurs from 200 to 700Hz.

The measurements investigated show that it is more difficult to cancel a reflection from the side, since it is does not have an incidence angle of a physical loudspeaker. The videos "Sidereflection scenario" BW300 and BW150 show an interesting effect. The BW300 (downsampled to 300Hz

bandwidth) video show that there are still reflections visible after correction, and the bottom right difference sequence shows the soundfield "wobble" around the sweetspot. The difference at microphone 1 is close to zero, hence it lies still in the center while the rest of the soundfield bends around it. This confirms that reflection cancellation works better the closer to the sweetspot you are. Looking at the BW150 video this shows that the effect of the reflections under 150Hz has been cancelled. This understates previous statements.

5.4 Virtual sources: Error estimation and perceived angle

The figures 4.23 to 4.26 show different markers for where the different microphones calculate the perceived incidence angle of the soundwave. 4.23 show the perceived source when full bandwidth is being used, and the sound is played from the center speaker, and the right surround speaker. All markers lie inside the speakers that produced the signal, so it can be said that all five microphones perceive the audio to be coming from that direction. The next figure 4.24 shows the right surround loudspeaker moved out of its original position, and it is illustrated that even with the remapping algorithm running, the source is still perceived to be inside the speaker when looking at the full bandwidth. The figure 4.25 shows that when the bandwidth is reduced to 375Hz and the remapping is turned on, the microphones closer to the sweetspot perceive the audio to come from outside the loudspeaker, closer to its original position. When the bandwidth is reduced further 4.26, the figures show that some of the outer microphones are moved closer, but no improvement is observed at the inner microphones. This is caused by the fact that the more the data is downsampled, the smaller are the differences between the levels at the different angles when the microphoneboom is rotated. This makes it much harder to detect a certain point where the propagation delay is the smallest. The inner microphones has such little movement in the first place, that it makes it very difficult to detect these small differences. Due to the possible error in these figures it could be considered to take a look at some frequency response measurements before comparing the results to theoretical data. The plots 4.27 show the error in frequency response for the different microphones over the different measured angles. They have been lowpassed to 750Hz, 375Hz, 187Hz, and 93Hz to be able to see details of how the frequency reproduction is reconstructed. The error is shown in dB difference between the response of the system with a displaced loudspeaker, and the reference ITU-775 system (see eq:4.1). Microphone 1 seems to have little error all the way up to 750Hz. Microphone 2 comes a little bit lower at about 600Hz. The errors for microphone 3 starts at approximately 180Hz, microphone 4 at 120Hz, and microphone 5

at about 80Hz. Compared to theoretical calculations on reconstruction of the wavefield in table 4.1, the results are close to the theoretical numbers except for microphone 2 which performs much better than expected. The theoretical numbers are based on perfect reconstruction. None of the results are perfect, but can give a good pointer that perfect reconstruction is very difficult to achieve, and that a perfectly reconstructed frequency response does not imply a perfectly reconstructed wavefield.

The videos "Virtual sources scenario" BW300 and BW150 show an interesting effect. Again the bending effect around the microphone closest to the center can be seen. This effect is smaller for BW150 than BW300, and this implies what was found in the errorestimate and the perceived angle scripts.

6

Conclusion

This report has investigated the evaluation of a multichannel audio reproduction system. Measurements have been done on frequency and phase response, reflection cancellation, and loudspeaker placement correction. The focus has been on measurements around the sweetspot. Working with the data has helped develop useful skills in processing and manipulating data through the use of Matlab. Skills in theoretical and practical understanding of measurements and important acoustical terms has increased together with the insight and understanding of a professional room correction unit.

The importance of being critical and thorough has been tested, and making only a small error in the phase plot script made the whole concept difficult until it was straightened out. After looking at these relatively simple constructed scenarios, and seen how difficult they are to interpret, it has given insight and respect for the complexity of a reverberant room. The anechoic measurements have been very useful, and the use of Matlab cannot be underestimated as a powerful tool for postprosessing and interpretation.

It has been shown that with a professional room correction device, the differences between loudspeakers and listening environments can be minimized. Though a flat frequency response is the most discussed topic when it comes to room correction, one must not forget the phase. The phase response of the loudspeaker and room affects the spacial image in a complex soundfield such as music or speech. It is important to remember that transient resolution and phase alignment is important in order to create source images between loudspeakers, and to recreate an harmonic signal correctly. The corrected phase plots presented in this report have common features; Midrange frequencies are more or less in phase while lower

frequencies (sub 200Hz) lies a bit ahead and frequencies above 2kHz lie a little bit behind. The overall phase distance between the high and low frequencies have improved significantly. Phase ripple is also a problem, and the Optimizer deals with this to a certain extent.

There have been reoccurring problem areas in the measurements throughout the whole report, and these are determined by the time of arrival of the different reflections in the room. The Optimizer has a way of finding out if its correction algorithms will help or not, and if they will not, they will only make matters worse. Therefore in these kind of situations they are not applied. This leads to the reflection cancellation which is basically determined by the arrival time of the reflection. The Optimizers algorithm for deconvolution will by default try to deconvolve an early reflection with an arrival time up to: 1000ms(up to 50Hz), 70ms(up to 150Hz), 10ms(up to 500Hz) and 1ms(up to 4kHz). But it is shown that the Optimizer has more trouble trying to cancel a reflection which has an incidence angle that does not correspond with a loudspeaker placement. It could also be that for some of the measurements the arrival time of the reflection exceeded 10ms and therefore the Optimizer would not try to deconvolve. Hence within the 10ms it was shown that the reflection from the front was deconvolved as well behind the sweetspot as in the front of the sweetspot, which implies different arrival times of both the reflector and the other loudspeakers reflections.

The fact that a significant impact could be observed from the reflections coming from the loudspeakers in the setup themselves reminds us that nothing will be easy in a livingroom or other reverberant environment.

Allthough it was difficult to obtain data to trust while looking at the virtual sources scenario, the three methods that was used point in the same direction as the theory of the subject. The remapping algorithm works to a certain extent based on frequency and distance from the sweetspot as one should expect from a spherical harmonics based remapping algorithm using only five loudspeakers.

The video sequences that were developed give an interesting new angle on the problems that were investigated. Other than looking at plots of different angles which is difficult and time consuming, the videos showed an intuitive perspective that enlightened the same issues as the common presented data of frequency and phase response measurements. The way the "pancakes" bends around the middle seems very easy to understand, and shows the effect that was wanted.

For the specific cases that were investigated in this report, the Optimizer showed the following:

- Frequency and phase response will in every situation be optimized to the extent of the Optimizers algorithms.
- Every case shows improvement in the frequency and phase response over the whole measured area.
- Direct frontal reflections was deconvolved up to 300Hz over the whole measured area with a radius of 56cm.
- The side reflection was deconvolved roughly up to 200Hz for microphones 1 through 3, up to a radius of 31.25cm, and up to 100Hz for microphones 4 and 5.
- The ability to create virtual sources corresponds fairly to the theoretical expectations.

Possible sources of errors in the measurements can be: imperfections in the measuring equipment i.e microphones and microphone amplifiers, reflections from objects in the anechoic lab such as loudspeakers and microphones along with their stands, the anechoic lab is only anechoic down to about 100Hz, bad wiring and microphone connectors.

6.1 Possible improvements and continuation of the work

The frequency and phase response of several different loudspeakers could be measured. It would be interesting to see how low the quality of the speaker can be before the Optimizer no longer can compensate for its imperfections.

New measurements could be performed where the reflectors position is investigated a bit more. How to get the best possible deconvolution from the Optimizer? The next step would be to take the system out of the anechoic room and into a reverberant room to look at behavior with multiple reflections and reverberation.

Since all the impulse responses of the system are recorded it would be possible to convolve different signals into the room and make new video sequences of how they propagate. Sinusoids and simple music or speech signals can be used.

More loudspeakers could be investigated with a 7, 16 or 24 channel Optimizer.

Bibliography

[1] Ambisonic.net. Ambisonics. http://www.ambisonic.net/, 2007.

[2] Thushara D. Abhayapala Darren B. Ward. Reproduction of a plane-wave sound field using an array of loudspeakers. *IEEE TRANSACTIONS ON SPEECH AND AUDIO PROCESSING*, 9(6), September 2001.

[3] Sweetwater Sound Inc. Itu-775. http://www.sweetwater.com/expert-center/glossary/t--ITU775Surround, 2007.

[4] Sebastien Moreau Jerome Daniel, Rozenn Nicol. Further investigations of high order ambisonics and wavefield synthesis for holophonic sound imaging. *Audio Eng.Soc. 114TH CONVENTION Convention Paper 5788*, March 2003.

[5] Asbjørn Krokstad. *Akustikk for ingeniører*. NTNU, 1999.

[6] Dave Malham. Second and third order ambisonics. http://www.york.ac.uk/inst/mustech/3d_audio/secondor.html, 2005.

[7] M.A. Poletti. A unified theory of horizontal holographic sound systems. *Audio Eng.Soc.*, 48(12), December 2000.

[8] M.A. Poletti. Three-dimentional surround sound systems based on spherical harmonics. *Audio Eng.Soc.*, 53(11), November 2005.

[9] Volker SchÃ¶nefeld. Spherical harmonics. http://heim.c-otto.de/~volker/prosem_paper.pdf, 2005.

[10] John Vanderkooy Stanly P. Lipshitz, Mark Popcock. On the audibility of midrange phase distortion in audio systems. *Audio Eng.Soc.*, 30(9), September 1982.

[11] Trinnov. Trinnov optimizer. `http://www.trinnov.com/product_Optimizer.php`, 2007.

[12] Wikipedia. Spherical harmonics. `http://en.wikipedia.org/wiki/Spherical_harmonics`, 2007.

[13] Joachim Wille. Evaluation of trinnov optimizer audio reproduction system, December 2007.

Part I

Appendix

A

Equipment

Loudspeakers: Dynaudio Acoustics BM6A fig: 3.3 (b)

R: 00676249
C: 00487311
L: 00676253
RS: 00562178
LS: 00487300

Computer with WinMLS and 6ch soundcard

Microphone amp: Norsonic 366 fig: 3.4

Mic 1&2: 20615
Mic 3&4: 20626
Mic 5: 18508
All preamps; Gain 20dB, Filtersetting: F, 0V pol, OLfunct: inst

Turntable : Norsonic NOR265 SER: 29300 fig: 3.3 (b)

Microphones fig: 3.3 (a)

Mic 1: Norsonic 1201 SER: 23890 Caps: Norsonic UC:53N 01411
Mic 2: Norsonic 1201 SER: 23823 Caps: Norsonic UC:53N 13574
Mic 3: Norsonic 1201 SER: 20976 Caps: Norsonic UC:53N 01404
Mic 4: Norsonic 1201 SER: 22038 Caps: Norsonic UC:53N 41759
Mic 5: Norsonic 1201 SER: 30517 Caps: Norsonic UC:53N 13558

Microphone boom fig:3.3 (b)

Trinnov Optimizer, ID 815117, Soundcard 9632 fig: 3.1

Mic V5-11 fig: 3.5

B

Matlab code

The presented Matlab code is a selection of the developed code to give more in depth information of how the data was obtained. These extractions of the code may and may not work depending of surrounding parameters (for instance names of measured data, directory structure etc.). This code was written and run in Matlab 7.4.0 R2007a on a Linux based OS.

B.1 Frequency, phase and impulse response

```
%gives impulse, phase and frequency response of measurements. All
    5
%microphones and angles 0, 90 and 180 degrees

clear all

for zz=1:2 %1;before 2;after correction

    if zz==2
        close all
    else
    end

for jj=[0 90 180] %

    for ii=5:-1:1
        before = (['optmeas_phys_C_compoff_optoff_deg_' num2str(jj
            ) 'Ch' num2str(ii) '.wmb']); %name before correction
            response
        after = (['optmeas_phys_C_compon_optoff_deg_' num2str(jj)
            'Ch' num2str(ii) '.wmb']);%name after correction
            response
        befaf = 'before'; %gives name to plots
        if zz==2 %gives before and after
            before=after; %changes impulsrespons
```

```
    befaf = 'after'; %changes name
else
end

%change attributes of plot area
set(0, 'DefaultAxesFontSize', 20);
scrnsize = get(0, 'ScreenSize');
set(0, 'DefaultFigurePosition', [0 150 scrnsize(3)/2
    scrnsize(4)]);
set(0, 'DefaultLineLineWidth', 2);

%load impulse responses before and after compensation
[imp1, fs] = loadimp(before);% fs = sample frequency
imp1 = -imp1; %inverts values because of polarity shift in
     measurements

c = 344; % speed of sound
t = [0:1/fs:((length(imp1)-1)/fs)]; %time vector

%plot impulse responses by samples
figure(1+jj)
plot(imp1);
grid on
hold all
xlabel('Samples')
ylabel('Impulse Response')
title(['Impulse Response ' befaf ' correction at ' num2str
    (jj) '\circ'])
legend('mic5','mic4','mic3','mic2','mic1')
axis([1500 2200 -300 550]);
saveas(gcf,(['../Bilder/master/loudspeaker/' befaf '
    _impulse_deg_' num2str(jj) '.eps']), 'psc2')

%fft of impulse responses to determine initial propagation
     delay
nfft = length(imp1)*2+1;                %Define fft length
freq1 = fft(imp1,nfft);                 %Take fft of
    impulse response
freq1 = freq1(1:nfft/2);                %Use first half of
     fft because of symmetry
f = fs/nfft*[0:nfft/2-1];               %Construct
    frequency vector
unw1 = unwrap(angle(freq1));            %Create unwrapped
    phase response
[A1,B1] = linfit(unw1, fs/2/length(f)); %Determining the
    slope of the phase response
propdelay1 = (abs(B1*fs/2/pi));         %Using slope to
    estimate propagation delay

%upsample by interpolation between samples by factor 10
tup = 0:max(t)/length(t)/10:(1-1/length(t)/10)*max(t);
fsup = fs*10;
imp1re = interp1(t, imp1, tup);
nfftup = length(imp1re)*2+1; %define fft length
```

```
        %Cut away propagtion delay
        imp1cut = imp1re(round(propdelay1*10)-0:length(imp1re));
               %Calculated from original impulse response

        %plot phase response of loudspeaker without propagation
            delay
        nfftnew1 = length(imp1cut)*2+1; %define fft length
        freq1lsp = fft(imp1cut,nfftnew1);
        freq1lsp = freq1lsp(1:nfftnew1/2);
        fnew1 = fsup/nfftnew1*[0:nfftnew1/2-1]; %Construct
            individual frequency vector

        ph1 = (angle(freq1lsp)*180/pi+10000); %phase response
            vector. adds a value to fool F2smospa
        [phase, fvecsmo] = F2smospa(fnew1,abs(ph1),6,20,20000,500)
            ; %smooths phase response
        figure(2+jj)
        semilogx(fvecsmo, phase-10000)
        grid on
        hold all
        axis([20 20000 -185 185]);
        title(['Phase Response ' befaf ' correction at ' num2str(
            jj) '\circ'])
        legend('mic5','mic4','mic3','mic2','mic1')
        xlabel('Frequency (Hz)')
        ylabel('Phase (degrees)')
        saveas(gcf,(['../Bilder/master/loudspeaker/' befaf '
            _phase_deg_' num2str(jj) '.eps']), 'psc2')

        [y1abs, fvecsmo] = F2smospa(f,abs(freq1),6,20,20000,500);
            %smooths frequency response

        %plot frequency responses
        figure(3+jj)
        semilogx(f,10*log10(abs(freq1)))
        axis([20 20000 0 40]);
        grid on
        hold all
        title(['Frequency Response ' befaf ' correction at '
            num2str(jj) '\circ'])
        legend('mic5','mic4','mic3','mic2','mic1','location', '
            SouthEast')
        xlabel('Frequency (Hz)')
        ylabel('Amplitude (dB)')
        saveas(gcf,(['../Bilder/master/loudspeaker/' befaf '
            _frequency_deg_' num2str(jj) '.eps']), 'psc2')

    end

end
end
```

B.2 Video sequences

B.2.1 Call

```
% Calls functions for creating video sequences

%names and directories of all datasets/measurement data
measnames = {'optmeas_10112007_1738/optmeas_phys_C_compoff_optoff'
     'optmeas_10112007_1917/optmeas_phys_C_compon_optoff'...
    etc...

    [p1,p2,pdiff,pref,phi,r] = videomatrix(cell2mat(measnames(8)),
         cell2mat(measnames(9)), cell2mat(measnames(2))); % loads
        two different datasets to be investigated and a dataset
        with single corrected speaker for reference
    %returns impulse response matrixes of the two investigated
    %measurements, the difference between them, a reference, an
        angle
    %vector and a radius vector


    [x,y,mov] = avifilemaker(phi, r, p1, p2, pdiff, pref, cell2mat
        (measnames(8))); % creates the moviesequence
    %returns x,y coordinates and a Matlab movie object
```

B.2.2 Build matrixes

```
function [bigmatrix1, bigmatrix2, bigmatrixdiff, bigmatrixref, phi
    , r] = mkbigmatrix2(filename1, filename2, filenameref)

down = 16*10; %defines no. of times downsampling
up = 10; %defines no. of times upsampling for smoother video
    playback

samples = (2^12)/down*up; %defines samples to shorten impulse
    response

bigmatrix1 = zeros(121, 5, samples); %makes a zero matrix for
    dataset 1
bigmatrix2 = zeros(121, 5, samples); %makes a zero matrix for
    dataset 2
bigmatrixref = zeros(121, 5, samples); %makes a zero matrix for
    reference dataset
bigmatrixdiff = zeros(121, 5, samples); %makes a zero matrix for
    difference between 1 or 2 and reference

phi = zeros(121, 5); %makes a zero angle matrix
r = zeros(121, 5); %makes a zero radius matrix

for ii=1:121
    %make suitable strings for importing files to matlab
    file1a = [filename1 '_deg_' num2str(ii*3-3) 'Ch1.wmb'];
    file2a = [filename1 '_deg_' num2str(ii*3-3) 'Ch2.wmb'];
    file3a = [filename1 '_deg_' num2str(ii*3-3) 'Ch3.wmb'];
    file4a = [filename1 '_deg_' num2str(ii*3-3) 'Ch4.wmb'];
    file5a = [filename1 '_deg_' num2str(ii*3-3) 'Ch5.wmb'];

    file1b = [filename2 '_deg_' num2str(ii*3-3) 'Ch1.wmb'];
    file2b = [filename2 '_deg_' num2str(ii*3-3) 'Ch2.wmb'];
    file3b = [filename2 '_deg_' num2str(ii*3-3) 'Ch3.wmb'];
    file4b = [filename2 '_deg_' num2str(ii*3-3) 'Ch4.wmb'];
    file5b = [filename2 '_deg_' num2str(ii*3-3) 'Ch5.wmb'];

    file1ref = [filenameref '_deg_' num2str(ii*3-3) 'Ch1.wmb'];
    file2ref = [filenameref '_deg_' num2str(ii*3-3) 'Ch2.wmb'];
    file3ref = [filenameref '_deg_' num2str(ii*3-3) 'Ch3.wmb'];
    file4ref = [filenameref '_deg_' num2str(ii*3-3) 'Ch4.wmb'];
    file5ref = [filenameref '_deg_' num2str(ii*3-3) 'Ch5.wmb'];

    %imports/reades files from WinMLS measurement files and
        resamples them
    file11a = interp(resample(loadimp(file1a), 1, down), up);
    file22a = interp(resample(loadimp(file2a), 1, down), up);
    file33a = interp(resample(loadimp(file3a), 1, down), up);
    file44a = interp(resample(loadimp(file4a), 1, down), up);
    file55a = interp(resample(loadimp(file5a), 1, down), up);
```

```
    file11b = interp(resample(loadimp(file1b), 1, down), up);
    file22b = interp(resample(loadimp(file2b), 1, down), up);
    file33b = interp(resample(loadimp(file3b), 1, down), up);
    file44b = interp(resample(loadimp(file4b), 1, down), up);
    file55b = interp(resample(loadimp(file5b), 1, down), up);

    file11ref = interp(resample(loadimp(file1ref), 1, down), up);
    file22ref = interp(resample(loadimp(file2ref), 1, down), up);
    file33ref = interp(resample(loadimp(file3ref), 1, down), up);
    file44ref = interp(resample(loadimp(file4ref), 1, down), up);
    file55ref = interp(resample(loadimp(file5ref), 1, down), up);


    r(ii,:) = [0.0625,0.1875,0.3125,0.4375,0.5625]; %creates
        radius vector

    for kk = 1:samples
        %shortens and puts into big matrix
        bigmatrix1(ii,1,kk) = file11a(kk+60);
        bigmatrix1(ii,2,kk) = file22a(kk+60);
        bigmatrix1(ii,3,kk) = file33a(kk+60);
        bigmatrix1(ii,4,kk) = file44a(kk+60);
        bigmatrix1(ii,5,kk) = file55a(kk+60);

        bigmatrix2(ii,1,kk) = file11b(kk+50);
        bigmatrix2(ii,2,kk) = file22b(kk+50);
        bigmatrix2(ii,3,kk) = file33b(kk+50);
        bigmatrix2(ii,4,kk) = file44b(kk+50);
        bigmatrix2(ii,5,kk) = file55b(kk+50);

        bigmatrixref(ii,1,kk) = file11ref(kk+50);
        bigmatrixref(ii,2,kk) = file22ref(kk+50);
        bigmatrixref(ii,3,kk) = file33ref(kk+50);
        bigmatrixref(ii,4,kk) = file44ref(kk+50);
        bigmatrixref(ii,5,kk) = file55ref(kk+50);

    end

    for jj = 1:5
        phi(ii,jj) = (ii*3-3);

    end
end

bigmatrixdiff = bigmatrix2 + bigmatrix1; %creates difference
    matrix between 1 & 2
```

B.2.3 Make AVI file

```
%Makes movie sequence of wave propagation from a single
    loudspeaker with and without optimalization by the Trinnov
    Optimizer. Resampled
%to see low frequency effects.

function [x,y,mov] = makemov(phi, r, p1, p2, pdiff, pref, name)

set(0, 'DefaultAxesFontSize', 20); % set axes fontsize
[x,y] = pol2cart(phi/180*pi,r); %convert coordinates

pdiff2 = p2-pref; %creates difference matrix between dataset 2 and
     reference

circle=zeros(121,5,204); % creates a "zero layer" for visual
    reference of 0

mov = avifile('name1', 'compression', 'none', 'fps', 5, 'quality',
     10); %creates movie object
a = get(0, 'ScreenSize'); %gets size of screen
h = figure('Position', [0 0 a(3)/2 a(4)]); %sets video size

for ii = 1:length(p1) %adds video frames sample by sample
    subplot(2,2,1)
    surf(x,y,circle(:,:,1)) %draw zero reference circle
    alpha(0.2)
    hold on
    surf(x,y,p1(:,:,ii))%plot coordinates impulse responses sample
         by sample
    text(0,0.7, '90\circ',  'FontSize', 20); %set axis angles
    text(0.6,0, '0\circ',  'FontSize', 20);
    text(-0.7,0, '180\circ',  'FontSize', 20);
    text(0,-0.9, '270\circ',  'FontSize', 20);
    set(gca, 'DataAspectRatioMode', 'auto','PlotBoxAspectRatioMode
        ', 'auto', 'CameraViewAngleMode', 'auto')
    colormap('default')
    axis([-0.6 0.6 -0.6 0.6 -5 7]);  %set axis size
    view(-12,18)%camera angle
    caxis([-2 2]); %color range
    title('Center speaker with reflector without correction')
    axis off
    hold off

    subplot(2,2,2)
    surf(x,y,circle(:,:,1))%draw zero reference circle
    alpha(0.2)
    hold on
    surf(x,y,p2(:,:,ii))
    text(0,0.7, '90\circ',  'FontSize', 20);
    text(0.6,0, '0\circ',  'FontSize', 20);
    text(-0.7,0, '180\circ',  'FontSize', 20);
```

```
    text(0,-0.9, '270\circ',  'FontSize', 20);
    set(gca, 'DataAspectRatioMode', 'auto','PlotBoxAspectRatioMode
        ', 'auto', 'CameraViewAngleMode', 'auto')
    colormap('default')
    axis([-0.6 0.6 -0.6 0.6 -5 7]);
    view(-12,18)
    caxis([-2 2]);
    title('Center speaker with reflector with correction')
    axis off
    hold off

    subplot(2,2,3)
    surf(x,y,circle(:,:,1))%draw zero reference circle
    alpha(0.2)
    hold on
    surf(x,y,pdiff2(:,:,ii))
    text(0,0.7, '90\circ',  'FontSize', 20);
    text(0.6,0, '0\circ',  'FontSize', 20);
    text(-0.7,0, '180\circ',  'FontSize', 20);
    text(0,-0.9, '270\circ',  'FontSize', 20);
    set(gca, 'DataAspectRatioMode', 'auto','PlotBoxAspectRatioMode
        ', 'auto', 'CameraViewAngleMode', 'auto')
    colormap('default')
    axis([-0.6 0.6 -0.6 0.6 -5 7]);
    view(-12,18)
    caxis([-2 2]);
    title('Difference between corrected reflection, and no
        reflection')
    axis off
    hold off

    subplot(2,2,4)
    surf(x,y,circle(:,:,1))%draw zero reference circle
    alpha(0.2)
    hold on
    surf(x,y,pref(:,:,ii))
    text(0,0.7, '90\circ',  'FontSize', 20);
    text(0.6,0, '0\circ',  'FontSize', 20);
    text(-0.7,0, '180\circ',  'FontSize', 20);
    text(0,-0.9, '270\circ',  'FontSize', 20);
    set(gca, 'DataAspectRatioMode', 'auto','PlotBoxAspectRatioMode
        ', 'auto', 'CameraViewAngleMode', 'auto')
    colormap('default')
    axis([-0.6 0.6 -0.6 0.6 -5 7]);
    view(-12,18)
    caxis([-2 2]);
    title('Reference: Center speaker without reflector')
    axis off
    hold off

    F = getframe(h); %"gets" movieframe
    mov = addframe(mov, F); %adds it to the movie object
    disp(['added frame no.' num2str(ii)])
end
```

```
mov = close(mov); %close movie object

% now encode

newdir = (['Videoer/' name '/']);
newfile = ([newdir 'film.avi']);

%use mencoder(linux SW) to encode avi
!mencoder name1.avi -fps 5 -o name1_comp.avi -ovc x264
s = movefile('name1_comp.avi', newfile, 'f') % s returns 0 if
    movefile fails to move file
if s==0
    mkdir(newdir);
    s = movefile('name1_comp.avi', newfile, 'f')
end

%delete original avi file
!rm name1.avi
```

B.3 FFT script

```
% function returns fft, fftabs, and frequency vector, and plots
    logarithmic
% frequency response.

function [y, yabs, f] = plotfft(x, Fs)
nfft = length(x)*2+1;
f = Fs/nfft*[0:nfft/2-1];
y = fft(x, nfft);
y = y(1:nfft/2);
yabs = abs(y);
% loglog(f, abs(y(1:nfft/2)), 'r')
```

B.4 Linear fit script (by Peter Svensson)

```
function [A,B,r2] = linfit(segment,Δx);
% linfit makes a line fit to a number of y-values that have
    equally
% spread x-values. The algorithm assumes  the first x-value to be
    0.
%
% Input parameters:
%   segment     A vector of y-values
%   Δx      The step size along the x-axis (e.g. 1/fs)
%
% Output parameters:
%   A,B         Line coefficients:   y = A + B*x
%   r2          The squared correlation coefficient
%
% Peter Svensson 981112 (svensson@tele.ntnu.no)
%
% [A,B,r2] = linfit(segment,Δx);
segment = segment(:);
l = length(segment);

sumx = (l-1)*l/2;
sumxx = (l-1)*l*(2*l-1)/6;
sumy = sum(segment);
sumxy = sum([0:l-1].'.*segment);
sumyy = sum(segment.^2);

B = (l*sumxy - sumx*sumy)/(l*sumxx - sumx*sumx);
A = (sumy-B*sumx)/l;
r2 = B*(l*sumxy-sumx*sumy)/(l*sumyy-sumy*sumy);
B = B/Δx;
```

B.5 Smoothing script (by Peter Svensson)

```
function [F2out,fvecout] = F2smospa(fvec,F2in,octfrac,fstart,fend,
    npoints);
% F2smo applies octave-band related smoothing to a squared
% TF magnitude function. A logarithmically spaced output vector
% is generated.
%
% Input parameters:
%   fvec    A vector with the frequencies that correspond to the
    TF.
%           They must be equally spaced.
%   F2in    A vector with the squared Tf magnitudes
%   octfrac The octave band fraction: 1 means octave band, 3 means
     third octaves etc.
%   fstart  The first frequency of the output vector
%   fend    The last frequency of the output vector
%   npoints The number of frequnecy values for the output vector
% Output parameters:
%   F2out   The output vector with smoothed squared TF magnitudes
%   fvecout The frequency values of the output vector
%
% Peter Svensson 981223 (peter@ta.chalmers.se)
%
% [F2out,fvecout] = F2smospa(fvec,F2in,octfrac,fstart,fend,npoints
    );

% Based on F2smo, by Johan Nielsen
df = fvec(2) - fvec(1);
fvecout = logspace(log10(fstart),log10(fend),npoints);
fvecout = fvecout.';
    nfreqsperdecade = npoints/( log10(fend) - log10(fstart));
    nfreqsperoctave = nfreqsperdecade*log10(2);
freqmultfac = 2^(1/(2*octfrac));
flosmo = fvecout/freqmultfac;   iv1 = round(flosmo/df)+1;
fhismo = fvecout*freqmultfac;   iv2 = round(fhismo/df)+1;

ivec = find(iv1<1);
if ~isempty(ivec),
    error(['fstart is too low (',num2str(fstart),')'])
end
ivec = find(iv2 > length(fvec));
if ~isempty(ivec),
    error(['fend is too high (',num2str(fend),')'])
end

F2out = zeros(npoints,1);
F2in = abs(F2in);
for ii=1:npoints,
   F2out(ii) = mean(F2in(iv1(ii):iv2(ii)));
end
```

B.6 Find angle of loudspeakers

B.6.1 Full bandwidth

```
clear all
warning off
measnames = {'optmeas_10112007_1738/optmeas_phys_C_compoff_optoff'
     'optmeas_10112007_1917/optmeas_phys_C_compon_optoff' ... etc
names = {'Center comp:off opt:off' 'Center comp:on opt:off' '
    Center comp:on opt:distance' ... etc

set(0, 'DefaultAxesFontSize', 20);
scrnsize = get(0, 'ScreenSize');
set(0, 'DefaultFigurePosition', [0 150 scrnsize(3)/2 scrnsize(4)])
    ;
set(0, 'DefaultLineLineWidth', 2);

number = 2^11; %defines samples to shorten impulse response
fs = 48000;
p1 = zeros(121, length(measnames));
p2 = zeros(121, length(measnames));
p3 = zeros(121, length(measnames));
p4 = zeros(121, length(measnames));
p5 = zeros(121, length(measnames));

for jj=17:17
    tic %time iterations to estimate completion time
    currentname = cell2mat(measnames(jj));

    ch1 = zeros(121,number); %makes a 2D zero matrix
    ch2 = zeros(121,number); %makes a 2D zero matrix
    ch3 = zeros(121,number); %makes a 2D zero matrix
    ch4 = zeros(121,number); %makes a 2D zero matrix
    ch5 = zeros(121,number); %makes a 2D zero matrix

    for ii=1:121
        %make suitable strings for importing files to matlab
        file1 = [currentname '_deg_' num2str(ii*3-3) 'Ch1.wmb'];
        file2 = [currentname '_deg_' num2str(ii*3-3) 'Ch2.wmb'];
        file3 = [currentname '_deg_' num2str(ii*3-3) 'Ch3.wmb'];
        file4 = [currentname '_deg_' num2str(ii*3-3) 'Ch4.wmb'];
        file5 = [currentname '_deg_' num2str(ii*3-3) 'Ch5.wmb'];

        %imports files
                file11 = LOADIMP(file1);
                file22 = LOADIMP(file2);
                file33 = LOADIMP(file3);
                file44 = LOADIMP(file4);
                file55 = LOADIMP(file5);

        %shortens and puts into big matrix
        ch1(ii,:) = -file11(1:number);
```

```
ch2(ii,:) = -file22(1:number);
ch3(ii,:) = -file33(1:number);
ch4(ii,:) = -file44(1:number);
ch5(ii,:) = -file55(1:number);
%         disp(['You have added angle no.' num2str(ii*3-3)
    ])

nfft = length(ch1)*2+1;                 %Define fft length
freq1 = fft(ch1(ii,:),nfft);                  %Take fft of
     impulse response
freq1 = freq1(1:nfft/2);                %Use first half of
     fft because of symmetry
freq2 = fft(ch2(ii,:),nfft);                  %Take fft of
     impulse response
freq2 = freq2(1:nfft/2);                %Use first half of
     fft because of symmetry
freq3 = fft(ch3(ii,:),nfft);                  %Take fft of
     impulse response
freq3 = freq3(1:nfft/2);                %Use first half of
     fft because of symmetry
freq4 = fft(ch4(ii,:),nfft);                  %Take fft of
     impulse response
freq4 = freq4(1:nfft/2);                %Use first half of
     fft because of symmetry
freq5 = fft(ch5(ii,:),nfft);                  %Take fft of
     impulse response
freq5 = freq5(1:nfft/2);                %Use first half of
     fft because of symmetry
f = fs/nfft*[0:nfft/2-1];               %Construct
    frequency vector
unw1 = unwrap(angle(freq1));            %Create unwrapped
    phase response
unw2 = unwrap(angle(freq2));            %Create unwrapped
    phase response
unw3 = unwrap(angle(freq3));            %Create unwrapped
    phase response
unw4 = unwrap(angle(freq4));            %Create unwrapped
    phase response
unw5 = unwrap(angle(freq5));
[A1,B1] = linfit(unw1, fs/2/length(f)); %Determining the
    slope of the phase response
[A2,B2] = linfit(unw2, fs/2/length(f)); %to estimate the
    initial delay
[A3,B3] = linfit(unw3, fs/2/length(f)); %Determining the
    slope of the phase response
[A4,B4] = linfit(unw4, fs/2/length(f)); %to estimate the
    initial delay
[A5,B5] = linfit(unw5, fs/2/length(f)); %Determining the
    slope of the phase response
p1(ii, jj) = (abs(B1*fs/2/pi));         %Using slope to
    estimate propagation delay
p2(ii, jj) = (abs(B2*fs/2/pi));
p3(ii, jj) = (abs(B3*fs/2/pi));         %Using slope to
    estimate propagation delay
```

```
        p4(ii, jj) = (abs(B4*fs/2/pi));
        p5(ii, jj) = (abs(B5*fs/2/pi));         %Using slope to
            estimate propagation delay

    end
    disp(['I have completed iteration ' num2str(jj) ' of  18....
        why?'])
    why %to make time go by
    time=toc;
    disp(['and I used ' num2str(time) ' seconds!'])
    disp(' ')
end

%find which index of each vector has the smallest propagation
    delay
[a, phisource1] = min(p1);
phisource1 = phisource1*3-3;
[b, phisource2] = min(p2);
phisource2 = phisource2*3-3;
[c, phisource3] = min(p3);
phisource3 = phisource3*3-3;
[d, phisource4] = min(p4);
phisource4 = phisource4*3-3;
[e, phisource5] = min(p5);
phisource5 = phisource5*3-3;

%sort data into matrix
phi = zeros(5,length(measnames));
for kk=1:length(measnames)
    phi(1,kk)= phisource1(kk);
    phi(2,kk)= phisource2(kk);
    phi(3,kk)= phisource3(kk);
    phi(4,kk)= phisource4(kk);
    phi(5,kk)= phisource5(kk);
end

phi = phi + 90; %rotate to fit loudspeaker setup
%plot perceived angles and loudspeaker setup
for zz = 1:length(measnames)
    figure(gcf+1)

    plot(cos((phi(1,zz))*2*pi/360), sin((phi(1,zz))*2*pi/360), '+b
        ', 'MarkerSize', 30)
    axis off
    hold on
    plot(cos((phi(2,zz))*2*pi/360), sin((phi(2,zz))*2*pi/360), 'oy
        ', 'MarkerSize', 30)
    plot(cos((phi(3,zz))*2*pi/360), sin((phi(3,zz))*2*pi/360), '*m
        ', 'MarkerSize', 30)
    plot(cos((phi(4,zz))*2*pi/360), sin((phi(4,zz))*2*pi/360), '.c
        ', 'MarkerSize', 30)
    plot(cos((phi(5,zz))*2*pi/360), sin((phi(5,zz))*2*pi/360), 'xk
        ', 'MarkerSize', 30)
    legend('mic1','mic2','mic3','mic4','mic5')
```

```
    title(cell2mat(names(zz)))
    xlim([-1.1 1.1])
    ylim([-1.1 1.1])
    plot(0,1, 's', 'MarkerSize', 50, 'MarkerEdgeColor', 'g')
    plot(cos((30+90)*2*pi/360), sin((30+90)*2*pi/360), 's', '
        MarkerSize', 50, 'MarkerEdgeColor', 'g')
    plot(cos((-30+90)*2*pi/360), sin((-30+90)*2*pi/360), 's', '
        MarkerSize', 50, 'MarkerEdgeColor', 'g')
    plot(cos((110+90)*2*pi/360), sin((110+90)*2*pi/360), 's', '
        MarkerSize', 50, 'MarkerEdgeColor', 'g')
    plot(cos((-110+90)*2*pi/360), sin((-110+90)*2*pi/360), 's', '
        MarkerSize', 50, 'MarkerEdgeColor', 'g')
    plot(0,0, 'o', 'MarkerSize', 50, 'MarkerEdgeColor', 'b')

    if zz>15
    plot(cos((-137+90)*2*pi/360), sin((-137+90)*2*pi/360), 's', '
        MarkerSize', 50, 'MarkerEdgeColor', 'r')
    else
    end
    saveas(gcf,(['../Bilder/master/oppfattetkilde/' cell2mat(names
        (zz)) '.eps']), 'psc2')
end
```

B.6.2 Lowpassed

```
clear all
warning off
measnames = {'optmeas_10112007_1738/optmeas_phys_C_compoff_optoff'
     'optmeas_10112007_1917/optmeas_phys_C_compon_optoff' ... etc
names = {'Center comp:off opt:off' 'Center comp:on opt:off' '
    Center comp:on opt:distance' ... etc

set(0, 'DefaultAxesFontSize', 20);
scrnsize = get(0, 'ScreenSize');
set(0, 'DefaultFigurePosition', [1920 150 scrnsize(3)/2 scrnsize
    (4)]);
set(0, 'DefaultLineLineWidth', 2);

number = 2^11; %defines samples to shorten impulse response
up=256;
down=128;
fs = 48000;
p1 = zeros(121, length(measnames));
p2 = zeros(121, length(measnames));
p3 = zeros(121, length(measnames));
p4 = zeros(121, length(measnames));
p5 = zeros(121, length(measnames));

for jj=1:length(measnames)
    tic
    currentname = cell2mat(measnames(jj));

    ch1 = zeros(121,number); %makes a 2D zero matrix
    ch2 = zeros(121,number); %makes a 2D zero matrix
    ch3 = zeros(121,number); %makes a 2D zero matrix
    ch4 = zeros(121,number); %makes a 2D zero matrix
    ch5 = zeros(121,number); %makes a 2D zero matrix


    for ii=1:121
        %make suitable strings for importing files to matlab
        file1 = [currentname '_deg_' num2str(ii*3-3) 'Ch1.wmb'];
        file2 = [currentname '_deg_' num2str(ii*3-3) 'Ch2.wmb'];
        file3 = [currentname '_deg_' num2str(ii*3-3) 'Ch3.wmb'];
        file4 = [currentname '_deg_' num2str(ii*3-3) 'Ch4.wmb'];
        file5 = [currentname '_deg_' num2str(ii*3-3) 'Ch5.wmb'];

        %import data and resample
        file11 = interp(resample(loadimp(file1), 1, down), up);
        file22 = interp(resample(loadimp(file2), 1, down), up);
        file33 = interp(resample(loadimp(file3), 1, down), up);
        file44 = interp(resample(loadimp(file4), 1, down), up);
        file55 = interp(resample(loadimp(file5), 1, down), up);

        %shortens and puts into big matrix
```

```
        ch1(ii,:) = -file11(1500:number+1499);
        ch2(ii,:) = -file22(1500:number+1499);
        ch3(ii,:) = -file33(1500:number+1499);
        ch4(ii,:) = -file44(1500:number+1499);
        ch5(ii,:) = -file55(1500:number+1499);

        [a, phisource1] = max(ch1(ii,:));
        [b, phisource2] = max(ch2(ii,:));
        [c, phisource3] = max(ch3(ii,:));
        [d, phisource4] = max(ch4(ii,:));
        [e, phisource5] = max(ch5(ii,:));

        p1(ii, jj) = phisource1;
        p2(ii, jj) = phisource2;
        p3(ii, jj) = phisource3;
        p4(ii, jj) = phisource4;
        p5(ii, jj) = phisource5;

    end
    disp(['I have completed iteration ' num2str(jj) ' of  18....
        why?'])
    why %to make time go by when matlab is working
    time=toc;
    disp(['and I used ' num2str(time) ' seconds!'])
    disp(' ')
end

x=1:121;

%define sinusoidal fit process
sinusfit = fittype('a0+a1*cos(2*pi*x/c1+phi)');
options=fitoptions(sinusfit);
options.startpoint=[1600 2 10 0];
options.lower=[1600 8 120 0];
options.upper=[2100 700 122 2*pi];

p1fitvec = zeros(121,18);
p2fitvec = zeros(121,18);
p3fitvec = zeros(121,18);
p4fitvec = zeros(121,18);
p5fitvec = zeros(121,18);

%fit curves
for ii=1:18

    p1fit =  fit(x',p1(:,ii), sinusfit, options);
    p2fit =  fit(x',p2(:,ii), sinusfit, options);
    p3fit =  fit(x',p3(:,ii), sinusfit, options);
    p4fit =  fit(x',p4(:,ii), sinusfit, options);
    p5fit =  fit(x',p5(:,ii), sinusfit, options);

    for jj=1:121
        p1fitvec(jj,ii) = p1fit(jj);
        p2fitvec(jj,ii) = p2fit(jj);
```

```
        p3fitvec(jj,ii) = p3fit(jj);
        p4fitvec(jj,ii) = p4fit(jj);
        p5fitvec(jj,ii) = p5fit(jj);
    end
end


close all
for ii=1:18
     figure
     plot(p3fitvec(:,ii),'g')
     hold on
     plot(p3(:,ii),'r')
end

[aa, phisource1] = min(p1fitvec);
phisource1 = phisource1*3-3;
[bb, phisource2] = min(p2fitvec);
phisource2 = phisource2*3-3;
[cc, phisource3] = min(p3fitvec);
phisource3 = phisource3*3-3;
[dd, phisource4] = min(p4fitvec);
phisource4 = phisource4*3-3;
[ee, phisource5] = min(p5fitvec);
phisource5 = phisource5*3-3;

phi = zeros(5,length(measnames));
for kk=1:length(measnames)
    phi(1,kk)= phisource1(kk);
    phi(2,kk)= phisource2(kk);
    phi(3,kk)= phisource3(kk);
    phi(4,kk)= phisource4(kk);
    phi(5,kk)= phisource5(kk);
end

phi = phi + 90; %rotate measurements to fit into lsp setup

%plot all perceived angles along with the ITU775 speaker setup
for zz = 1:length(measnames)
    figure(gcf+1)

    plot(cos((phi(1,zz))*2*pi/360), sin((phi(1,zz))*2*pi/360), '+b
        ', 'MarkerSize', 30)
    axis off
    hold on
    plot(cos((phi(2,zz))*2*pi/360), sin((phi(2,zz))*2*pi/360), 'oy
        ', 'MarkerSize', 30)
    plot(cos((phi(3,zz))*2*pi/360), sin((phi(3,zz))*2*pi/360), '*m
        ', 'MarkerSize', 30)
    plot(cos((phi(4,zz))*2*pi/360), sin((phi(4,zz))*2*pi/360), '.c
        ', 'MarkerSize', 30)
    plot(cos((phi(5,zz))*2*pi/360), sin((phi(5,zz))*2*pi/360), 'xk
        ', 'MarkerSize', 30)
    legend('mic1','mic2','mic3','mic4','mic5')
```

```
    title(cell2mat(names(zz)))
    xlim([-1.1 1.1])
    ylim([-1.1 1.1])
    plot(0,1, 's', 'MarkerSize', 50, 'MarkerEdgeColor', 'g')
    plot(cos((30+90)*2*pi/360), sin((30+90)*2*pi/360), 's', '
        MarkerSize', 50, 'MarkerEdgeColor', 'g')
    plot(cos((-30+90)*2*pi/360), sin((-30+90)*2*pi/360), 's', '
        MarkerSize', 50, 'MarkerEdgeColor', 'g')
    plot(cos((110+90)*2*pi/360), sin((110+90)*2*pi/360), 's', '
        MarkerSize', 50, 'MarkerEdgeColor', 'g')
    plot(cos((-110+90)*2*pi/360), sin((-110+90)*2*pi/360), 's', '
        MarkerSize', 50, 'MarkerEdgeColor', 'g')
    plot(0,0, 'o', 'MarkerSize', 50, 'MarkerEdgeColor', 'b')

    if zz>15
    plot(cos((-137+90)*2*pi/360), sin((-137+90)*2*pi/360), 's', '
        MarkerSize', 50, 'MarkerEdgeColor', 'r')
    else
    end
    axis([-1 1 -0.8 1.1])
    saveas(gcf,(['../Bilder/master/oppfattetkilde/lavpass/'
        cell2mat(names(zz)) '.eps']), 'psc2')
end
```

B.6.3 Errorestimate

```
% Makes error estimate of fft's. Must be run in folder containing
% measurement folders
warning off
% vector with names of all measurements and measurement folders
measnames = {'optmeas_10112007_1738/optmeas_phys_C_compoff_optoff'
     'optmeas_10112007_1917/optmeas_phys_C_compon_optoff' ... etc

set(0, 'DefaultAxesFontSize', 20);

l = length(measnames);

phi = [0:3:360];

down = 32; %downsample
up = 32;  %upsample

% smoothing
smoothingon = 1; %desides wether to use smooting or not. 1:
    smoothing on, 0: smooting off
octfrac = 3; % i.e. 3 gives smoothing over 1/3 octave bands
fstart = 20; % start frequency
fend = 20000; % end frequency
if smoothingon == 1
    npoints= 500; % new vector resolution if smooting on
else
    npoints= 4096; % vector resolution if smoothing off
end

% create zero matrixes
errormic1 = zeros(121, npoints);
errormic2 = zeros(121, npoints);
errormic3 = zeros(121, npoints);
errormic4 = zeros(121, npoints);
errormic5 = zeros(121, npoints);

errormic11 = zeros(121, npoints);
errormic22 = zeros(121, npoints);
errormic33 = zeros(121, npoints);
errormic44 = zeros(121, npoints);
errormic55 = zeros(121, npoints);

% extract data
[bigmat1, Fs] = makebig(cell2mat(measnames(17)), down, up);
bigmatref = makebig(cell2mat(measnames(15)), down, up); % sets
    reference matrix as physical source with compensation.

%get frequency response data and perform smoothing
for jj=1:121
    [y1, y1abs, fvec] = plotfft(bigmat1(1,:,jj), Fs);
    [y1ref, y1refabs] = plotfft(bigmatref(1,:,jj), Fs);
```

```
    if smoothingon == 1
        [y1abs, fvecsmo] = F2smospa(fvec,y1abs,octfrac,fstart,fend
            ,npoints);
        [y1refabs, fvecsmo] = F2smospa(fvec,y1refabs,octfrac,
            fstart,fend,npoints);
    end

    [y2, y2abs, fvec] = plotfft(bigmat1(2,:,jj), Fs);
    [y2ref, y2refabs] = plotfft(bigmatref(2,:,jj), Fs);
    if smoothingon == 1
        [y2abs, fvecsmo] = F2smospa(fvec,y2abs,octfrac,fstart,fend
            ,npoints);
        [y2refabs, fvecsmo] = F2smospa(fvec,y2refabs,octfrac,
            fstart,fend,npoints);
    end

    [y3, y3abs, fvec] = plotfft(bigmat1(3,:,jj), Fs);
    [y3ref, y3refabs] = plotfft(bigmatref(3,:,jj), Fs);
    if smoothingon == 1
        [y3abs, fvecsmo] = F2smospa(fvec,y3abs,octfrac,fstart,fend
            ,npoints);
        [y3refabs, fvecsmo] = F2smospa(fvec,y3refabs,octfrac,
            fstart,fend,npoints);
    end

    [y4, y4abs, fvec] = plotfft(bigmat1(4,:,jj), Fs);
    [y4ref, y4refabs] = plotfft(bigmatref(4,:,jj), Fs);
    if smoothingon == 1
        [y4abs, fvecsmo] = F2smospa(fvec,y4abs,octfrac,fstart,fend
            ,npoints);
        [y4refabs, fvecsmo] = F2smospa(fvec,y4refabs,octfrac,
            fstart,fend,npoints);
    end

    [y5, y5abs, fvec] = plotfft(bigmat1(5,:,jj), Fs);
    [y5ref, y5refabs] = plotfft(bigmatref(5,:,jj), Fs);
    if smoothingon == 1
        [y5abs, fvecsmo] = F2smospa(fvec,y5abs,octfrac,fstart,fend
            ,npoints);
        [y5refabs, fvecsmo] = F2smospa(fvec,y5refabs,octfrac,
            fstart,fend,npoints);
    end

    %make error matrix
    errormic1(jj, :) =  (y1abs)./y1refabs;
    errormic2(jj, :) =  (y2abs)./y2refabs;
    errormic3(jj, :) =  (y3abs)./y3refabs;
    errormic4(jj, :) =  (y4abs)./y4refabs;
    errormic5(jj, :) =  (y5abs)./y5refabs;
end

if smoothingon == 0
    fvecsmo = fvec;
end
```

```
fmax = Fs/down/2

savesize = [1,1,30,10];

%mesh all 5 microphones frequency response error over 360degrees
figure(gcf)
subplot(1,2,1)
surf(log10(fvecsmo), phi, 10*log10(errormic11))
axis([1.7 2.3 0 360 -10 10]);
caxis([-2 2]);
view(85, 15);
shading flat
set(gca,'xtick', [1.8 2 2.3 2.6 2.88 3 4])
set(gca,'xticklabel', [63 100 200 400 750 1000 10000])
title('Mic 1')
xlabel('F (Hz)');
ylabel('\theta \circ')
zlabel('Freq. resp. error (dB)')
subplot(1,2,2)
surf(log10(fvecsmo), phi, 10*log10(errormic11))
axis([1.7 2.3 0 360 -10 10]);
caxis([-2 2]);
view(0, 0);
shading flat
set(gca,'xtick', [1.8 2 2.3 2.6 2.88 3 4])
set(gca,'xticklabel', [63 100 200 400 750 1000 10000])
title('Sideview of figure Mic 1')
xlabel('Frequency (Hz)');
ylabel('\theta \circ')
zlabel('Freq. resp. error (dB)')
set(gcf, 'renderer', 'painter', 'paperposition', savesize)
% saveas(gcf,(['../Bilder/master/error/lsperror1' num2str(gcf) '
    .eps']), 'psc2')
print(gcf, '-depsc', '../Bilder/master/error/lsperror1.eps')

figure(gcf+1)
subplot(1,2,1)
surf(log10(fvecsmo), phi, 10*log10(errormic22))
axis([1.7 2.3 0 360 -10 10]);
caxis([-2 2]);
view(85, 15);
shading flat
set(gca,'xtick', [1.8 2 2.3 2.6 2.88 3 4])
set(gca,'xticklabel', [63 100 200 400 750 1000 10000])
title('Mic 2')
xlabel('F (Hz)');
ylabel('\theta \circ')
zlabel('Freq. resp. error (dB)')
subplot(1,2,2)
surf(log10(fvecsmo), phi, 10*log10(errormic22))
axis([1.7 2.3 0 360 -10 10]);
caxis([-2 2]);
view(0, 0);
```

```
shading flat
set(gca,'xtick', [1.8 2 2.3 2.6 2.88 3 4])
set(gca,'xticklabel', [63 100 200 400 750 1000 10000])
title('Sideview of figure Mic 2')
xlabel('Frequency (Hz)');
ylabel('\theta \circ')
zlabel('Freq. resp. error (dB)')
set(gcf, 'renderer', 'painter', 'paperposition', savesize)
% saveas(gcf,(['../Bilder/master/error/lsperror2' num2str(gcf) '
    .eps']), 'psc2')
print(gcf, '-depsc', '../Bilder/master/error/lsperror2.eps')


figure(gcf+1)
subplot(1,2,1)
surf(log10(fvecsmo), phi, 10*log10(errormic33))
axis([1.7 2.3 0 360 -10 10]);
caxis([-2 2]);
view(85, 15);
shading flat
set(gca,'xtick', [1.8 2 2.3 2.6 2.88 3 4])
set(gca,'xticklabel', [63 100 200 400 750 1000 10000])
title('Mic 3')
xlabel('F (Hz)');
ylabel('\theta \circ')
zlabel('Freq. resp. error (dB)')
subplot(1,2,2)
surf(log10(fvecsmo), phi, 10*log10(errormic33))
axis([1.7 2.3 0 360 -10 10]);
caxis([-2 2]);
view(0, 0);
shading flat
set(gca,'xtick', [1.8 2 2.3 2.6 2.88 3 4])
set(gca,'xticklabel', [63 100 200 400 750 1000 10000])
title('Sideview of figure Mic 3')
xlabel('Frequency (Hz)');
ylabel('\theta \circ')
zlabel('Freq. resp. error (dB)')
set(gcf, 'renderer', 'painter', 'paperposition', savesize)
% saveas(gcf,(['../Bilder/master/error/lsperror3' num2str(gcf) '
    .eps']), 'psc2')
print(gcf, '-depsc', '../Bilder/master/error/lsperror3.eps')


figure(gcf+1)
subplot(1,2,1)
surf(log10(fvecsmo), phi, 10*log10(errormic44))
axis([1.7 2.3 0 360 -10 10]);
caxis([-2 2]);
view(85, 15);
shading flat
set(gca,'xtick', [1.8 2 2.3 2.6 2.88 3 4])
set(gca,'xticklabel', [63 100 200 400 750 1000 10000])
title('Mic 4')
```

```
xlabel('F (Hz)');
ylabel('\theta \circ')
zlabel('Freq. resp. error (dB)')
subplot(1,2,2)
surf(log10(fvecsmo), phi, 10*log10(errormic44))
axis([1.7 2.3 0 360 -10 10]);
caxis([-2 2]);
view(0, 0);
shading flat
set(gca,'xtick', [1.8 2 2.3 2.6 2.88 3 4])
set(gca,'xticklabel', [63 100 200 400 750 1000 10000])
title('Sideview of figure Mic 4')
xlabel('Frequency (Hz)');
ylabel('\theta \circ')
zlabel('Freq. resp. error (dB)')
set(gcf, 'renderer', 'painter', 'paperposition', savesize)
% saveas(gcf,(['../Bilder/master/error/lsperror4' num2str(gcf) '
    .eps']), 'psc2')
print(gcf, '-depsc', '../Bilder/master/error/lsperror4.eps')


figure(gcf+1)
subplot(1,2,1)
surf(log10(fvecsmo), phi, 10*log10(errormic55))
axis([1.7 2.3 0 360 -10 10]);
caxis([-2 2]);
view(85, 15);
shading flat
set(gca,'xtick', [1.8 2 2.3 2.6 2.88 3 4])
set(gca,'xticklabel', [63 100 200 400 750 1000 10000])
title('Mic 5')
xlabel('F (Hz)');
ylabel('\theta \circ')
zlabel('Freq. resp. error (dB)')
subplot(1,2,2)
surf(log10(fvecsmo), phi, 10*log10(errormic55))
axis([1.7 2.3 0 360 -10 10]);
caxis([-2 2]);
view(0, 0);
shading flat
set(gca,'xtick', [1.8 2 2.3 2.6 2.88 3 4])
set(gca,'xticklabel', [63 100 200 400 750 1000 10000])
title('Sideview of figure Mic 5')
xlabel('Frequency (Hz)');
ylabel('\theta \circ')
zlabel('Freq. resp. error (dB)')
set(gcf, 'renderer', 'painter', 'paperposition', savesize)
% saveas(gcf,(['../Bilder/master/error/lsperror5' num2str(gcf) '
    .eps']), 'psc2')
print(gcf, '-depsc', '../Bilder/master/error/lsperror5.eps')
```