Master's Thesis

Increasing Sound Quality using Digital Signal Processing in a Surveillance System

Patrik Siljeholm
Gustav Johnsson Henningsson

Department of Electrical and Information Technology,
Faculty of Engineering, LTH, Lund University, 2016.

Increasing Sound Quality using Digital Signal Processing in a Surveillance System

Patrik Siljeholm
elt11psi@student.lu.se
Gustav Johnsson Henningsson
atn10gjo@student.lu.se

Department of Electrical and Information Technology
Lund University

Advisor: Mikael Swartling & Nedelko Grbic, Lund University
Henrik Duner, Simon Christensson & Anders Svensson, Axis Communications

June 7, 2016

Printed in Sweden
E-huset, Lund, 2016

Abstract

Hearing is our second most important sense. The main focus of a surveillance camera is obviously the video but the audio shouldn't be neglected. It is possible to hear what the camera can't see and what is said in a conversation can't be showed in a video. Therefore there are good reasons to always try and achieve the best audio possible. This is not an easy task since cameras are operating in a plethora of different environments with different background noises and are mounted with varying distances to the origin of the sound.

This thesis investigates which possibilities there are for improving the audio quality using digital signal processing. The thesis consist of three main parts: the choice of microphone, equalization and noise reduction. The microphone is the first link in the audio system and influences the conditions for digital signal processing. An equalizer was implemented to compensate for the acoustics in and around the microphone to make the audio sound more natural and closer to the original sound. The equalizer estimates the frequency response from the microphone in the frequency domain and inverts it in order to get a flat frequency response. Surveillance cameras are stationary and not seldom there is some kind of source of stationary noise nearby, e.g. ventilation or power switching from the camera itself. From the user's point of view it can be tiresome and hard to distinguish different sounds, why noise reduction is desired. Four different noise reduction algorithms were investigated and implemented and finally one algorithm was chosen that produced the best result, which was Multi-band Spectral Subtraction. Since all non-stationary sounds should be left untouched and only stationary noises removed the noise is estimated over a long time. Different ways of reducing the musical noise which often occurs after noise reduction are tested and the listening experience was in focus when tweaking the parameters. Everything is implemented in real-time Matlab.

Acknowledgements

We would like to thank Nedelko Grbic for his guidance, support and valuable input throughout the thesis. We thank Mikael Swartling for sharing his knowledge and experience, and for providing us with real-time Matlab. We also thank our supervisors at Axis; Henrik Duner, Simon Christensson and Anders Svensson. They made us feel welcome and as a part of the Axis team and supported us and shared their knowledge about audio during our time at Axis. Lastly we thank Axis for giving us the opportunity to do this master thesis and for providing us with space and the necessary equipment.

Table of Contents

List of Figures

List of Tables

List of Abbreviations

AOP Acoustic Overload Point

AP Audio Precision

AR Auto Regressive

DFT Discrete Fourier Transform

DSP Digital Signal Processor

DTFT Discrete Time Fourier Transform

EIN Equivalent Input Noise

FFT Fast Fourier Transform

FIR Finite Impulse Response

GUI Graphical User Interface

IDFT Inverse Discrete Fourier Transform

IFFT Inverse Fast Fourier Transform

IIR Infinite Impulse Response

IP Internet Protocol

LTI Linear Time Invariant

MEMS MicroElectrical-Mechanical System

PSR Power Supply Rejection

PSRR Power Supply Rejection Ratio

SNR Signal to Noise Ratio

SPL Sound Pressure Level

THD Total Harmonic Distortion

VAD Voice Activity Detection

WOLA Weighted Overlap and Add

Chapter 1
Introduction

This chapter consists of the background and goal of this thesis work. Some information about Axis is also presented.

1.1 Background

Hearing is our second most important sense. The ability to capture good audio and present it to the user is important in many areas and neither the less in surveillance applications. Good quality audio can have many advantages compared to video in surveillance applications. An obvious example is that audio can easily capture information from all directions while video often has a limited field of view. But more importantly audio can pick up information that is not possible with only video, such as conversations, and offer a more complete representation of the camera's surroundings compared to when only using video. The main goal of this thesis work is to investigate the possibilities to improve the quality of the sound using digital signal processing. Digital signal processing can enhance, modify and adjust the sound captured by the microphone before it's presented to the user. This thesis work has explored and implemented a calibration process which adjusts the incoming audio's frequency response to be flat for all frequencies using equalization. Also a noise reduction algorithm which suppresses unwanted noise while retaining the wanted sound has been implemented. The first part of this thesis work was spent choosing a microphone for surveillance applications with suitable characteristics that was used during the project.

1.2 About Axis

Axis is mainly a producer of network cameras with over 2000 employees and a revenue of over 6 billion SEK 2015. Axis is based in Sweden, founded in 1984 by Mikael Karlsson, Martin Gren and Keith Bloodworth. Axis started making print servers and in 1996 the first network camera was released. Today, cameras have become the focus and the main source of income but in recent years Axis's new business department has begun to investigate other products connected to the network in especially the security sector. Some examples of new products are Axis doorstation and IP audio.

1.3 Thesis Outline

Chapter 2 presents the theory which this thesis is based on, such as sampling, filter banks, signal processing theory and performance measurements. Also different properties of microphones are explained.

Chapter 3 describes the work performed when choosing the microphone and how the calibration process and noise reduction algorithms were implemented.

Chapter 4 presents the achieved results for this thesis work.

Chapter 5 discusses and summarizes the result from this thesis work and proposes improvements and future work.

Chapter 2
Theory

This chapter presents the theory on which this thesis is based on such as microphone properties, digital signal processing and noise reduction methods.

2.1 Microphone

The first part of this master thesis consisted of selecting a microphone. The different properties of a microphone that were considered in the selection process are explained below.

2.1.1 Signal-to-Noise Ratio

The SNR of a microphone is measured with 1kHz sine wave at 94dB SPL [1]. The value is calculated as the ratio between the output level of the microphone with the 1kHz reference signal and with silence. Often the SNR is specified in dBA which is the A-weighted value over a 20kHz bandwidth. The weighting corresponds to the human ear's sensitivity to different frequencies [1].

2.1.2 Sensitivity

The sensitivity of a microphone gives information about the level of the output from the microphone for a given signal which is typically a 1kHz sine wave at 94dB SPL [1]. For an analog microphone the sensitivity is calculated with

$$\text{Sensitivity}_{dBV} = 20 \cdot \log_{10}\left(\frac{\text{Sensitivity}_{mV/Pa}}{\text{Output}_{\text{REF}}}\right) \tag{2.1}$$

where $\text{Output}_{\text{REF}}$ is the 1000mV/Pa level of the reference signal.

2.1.3 Dynamic Range

It is desired to have a linear response from the microphone at different sound levels. The dynamic range is the difference between the highest and lowest SPL where the microphone behaves linearly.

2.1.4 Frequency Response

The frequency response describes how the microphone responds to different frequencies, it can be visualized as a plot where the output level of the microphone is a function of the frequency over a frequency spectrum.

2.1.5 Total Harmonic Distortion

THD is measured using a pure tone signal and measuring the distortion on the output. THD is measured in percent as the ratio between the sum of the powers of the harmonic frequencies above the fundamental frequency and the power of the tone at the fundamental frequency [1]. The standard level of the input signal is 105dB SPL, the high level is because THD usually increases with the input level.

2.1.6 Directionality

Depending on usage, microphones with different directionalities can be used. Omnidirectional microphones picks up sound equally independent of the location of the sound, these are used in most Axis cameras. There are also unidirectional microphones which attenuates all sounds coming from another direction than the specified. These types of microphones can be used when the source of the sound is known and other sounds are undesirable.

2.1.7 Acoustic Overload Point

The AOP is the point where the THD reaches 10% and the value is specified as the SPL where 10% occurs [1]. This value equals the highest value in the dynamic range.

2.1.8 Equivalent Input Noise

EIN describes how quiet sounds the microphone can pick up. Sounds lower than EIN are below the noise floor of the microphone, EIN is the lower limit of the dynamic range. EIN can be derived from the other specifications like this

$$\text{EIN} = \text{AOP} - \text{dynamic range} \tag{2.2}$$

and

$$\text{EIN} = 94dB - \text{SNR}. \tag{2.3}$$

2.1.9 Power Supply Rejection and Power Supply Rejection Ratio

PSR and PSRR give indications on how well the microphone rejects noise present in the supply voltage. PSR is measured with a 217Hz, 100mV_{pp} square wave added to the supply voltage, the output is integrated and A-weighted over the audible frequency range [1]. PSRR is measured similarly but with a sine wave and instead of a single frequency it is measured over a frequency spectrum.

2.2 Sampling

Sampling reduces a continuous signal to a discrete signal consisting of a series of values corresponding to the amplitude of the continuous signal at certain points in time. The sampling frequency f_s determines the period time between samples to be

$$T = \frac{1}{f_s}. \tag{2.4}$$

To avoid aliasing, i.e. high frequencies in the continuous signal being seen as lower frequencies when sampling to a discrete signal, the sampling frequency needs to be at least twice of the higher limit of the bandwidth according to Nyquist's criterion

$$f_{\max} = \frac{f_s}{2}. \tag{2.5}$$

2.3 Discrete Fourier Transform

The sampled signal can, given that the samples are sampled equidistantly in time and that the signal is finite, be converted into a set of coefficients of complex sinusoids via the DFT. A coefficient represent the presence of a frequency with a phase in the signal. The transform of a discrete signal in the time domain into the frequency domain is done by

$$X_k = \sum_{n=0}^{N-1} x_n \cdot e^{-2\pi ikn/N}, \qquad 0 \leq k \leq N-1 \tag{2.6}$$

and the transformation back to time domain, the IDFT is

$$x_n = \frac{1}{N} \sum_{n=0}^{N-1} X_k \cdot e^{2\pi ikn/N}, \qquad 0 \leq k \leq N-1. \tag{2.7}$$

The FFT and IFFT are common and efficient ways of implementing DFT and IDFT respectively. FFT and IFFT reduces the complexity from $O(n^2)$ to $O(n \log n)$ using the Divide-and-Conquer design approach. In this master thesis, the Matlab functions *FFT* and *IFFT* are used.

2.4 Weighted-Overlap-and-Add Filter Bank

During real time signal processing the whole signal is not available and therefore it must be divided into frames. Each frame is processed individually and all the processing is done in the frequency domain. When the processing is done the frame is transferred back to the time domain. These two transformation steps are implemented by a filter bank. The transformation from time to frequency domain is done by the analysis part of the filter bank and the transformation back to the time domain is done by the synthesis part of the filter bank. In Figure 2.1 the analysis and synthesis can be seen to the left respectively right part of the figure.

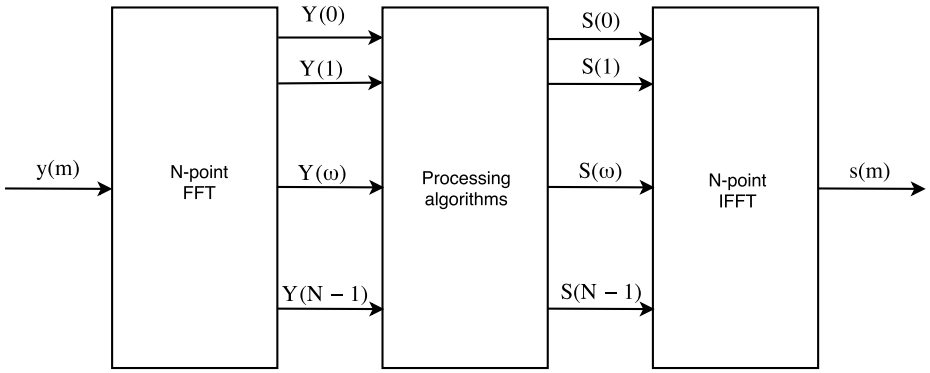

Figure 2.1: Filter bank with analysis and synthesis filter.

Since the frames are processed individually they have have to be fitted together again to form the enhanced time signal. To avoid audible clipping sounds during the reconstruction of the signal a processed called WOLA, is used [6]. The WOLA method uses overlap and windowing to smooth the ends of each frame and thus removing the unwanted clipping sounds. The WOLA consists of two parts that can be seen in Figure 2.2 and 2.3. In Figure 2.2 each frame is put into a buffer where a window is added before the FFT function is used on the whole buffer. In Figure 2.3 the IFFT function is used on the whole buffer after the desired processing is done and then added together with buffer from the previous iteration where padded zeros has been added and replaced the oldest frame. The delay of the system depends on the length of the frames, too small and the overhead will make it inefficient, too large and the delay will be too long. The size of the FFT should be a power of two for efficiency and be at least the length of the frame but can also be larger to increase resolution but for the cost of more computation. The size of the FFT determines the bandwidth of each sub-band to be f_s/N where N is the size if the FFT.

Depending on the window used in the WOLA method there is a limit of how much overlap that can be used. For a hanning window the overlap needs to be about 50%. In Figure 2.4 it is visible that 45% is not enough to get an even summation of the windows. In Figure 2.5 it is clear that 75% is enough which also is what is used for this master thesis. It can also be seen that the windows adds up to a sum greater than one, this needs to be corrected with a constant factor to preserve the level of the signal. With an overlap of 75%, four frames are windowed together and in each iteration a new frame is brought in which is windowed with the last three frames. Higher overlap results in more frames that should be windowed together and the delay will increase. When there are no clipping sounds there is no reason to increase the overlap further.

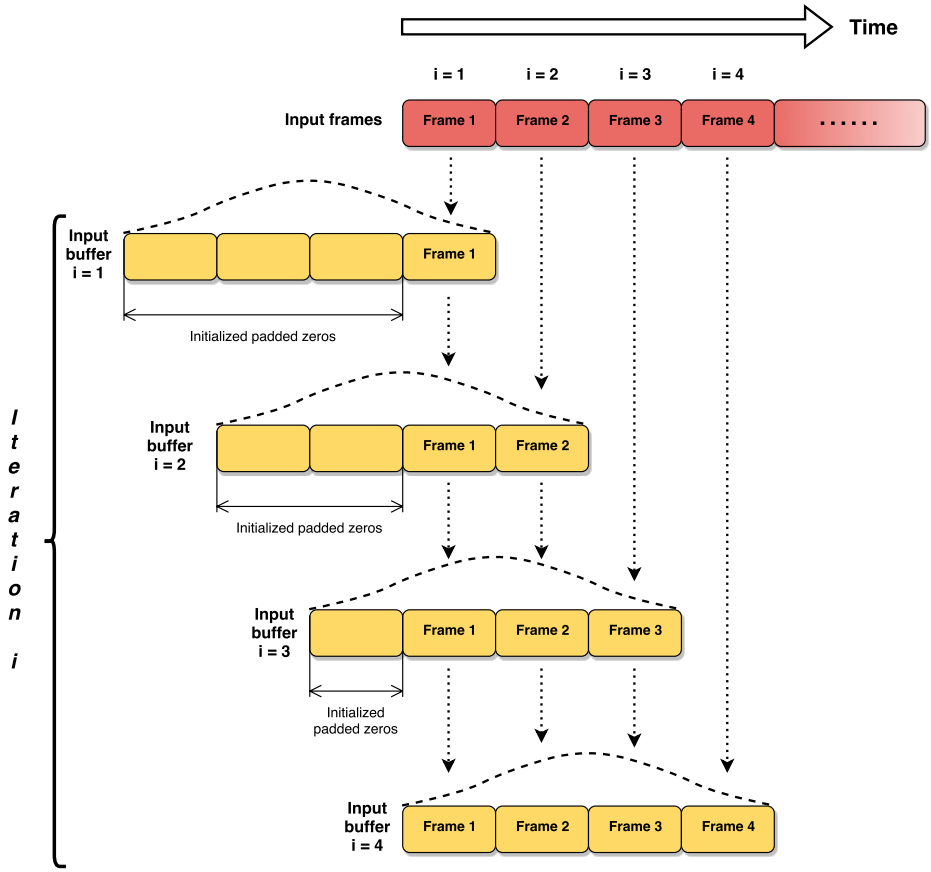

Figure 2.2: WOLA with 75% overlap during transformation to frequency domain.

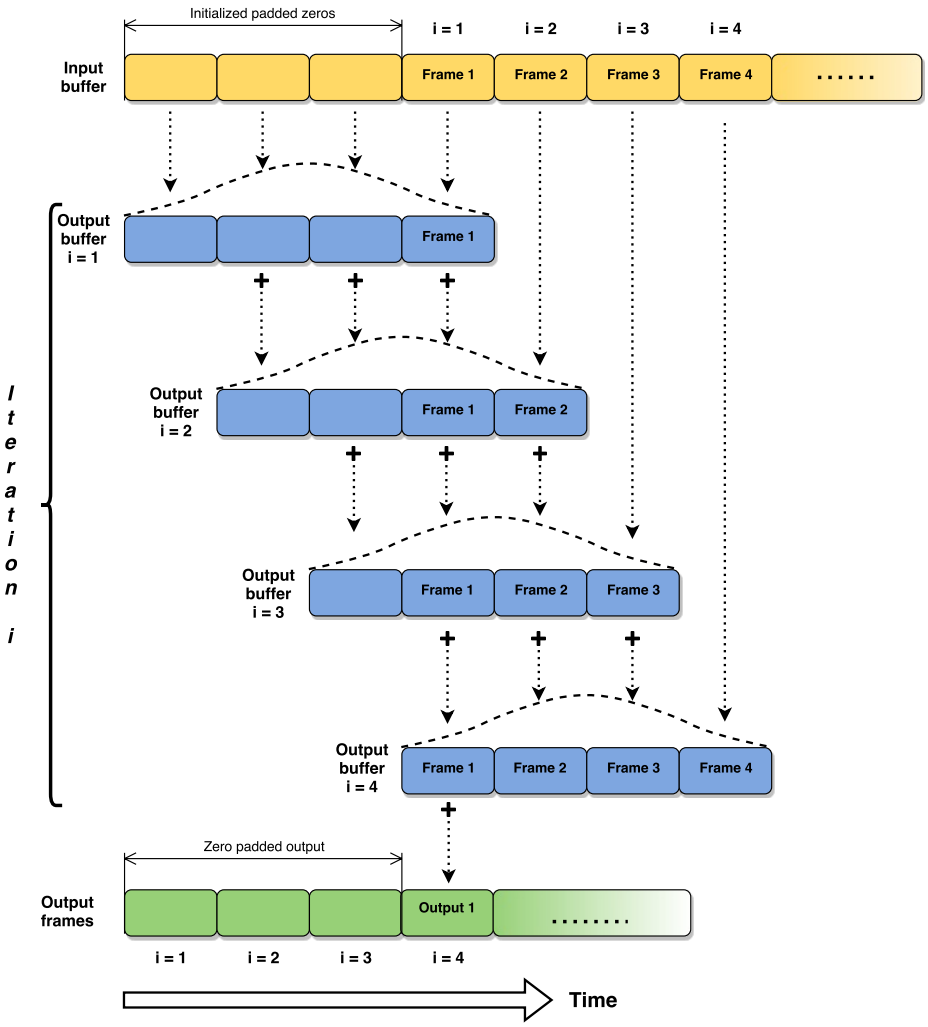

Figure 2.3: WOLA with 75% overlap during transformation back to time domain.

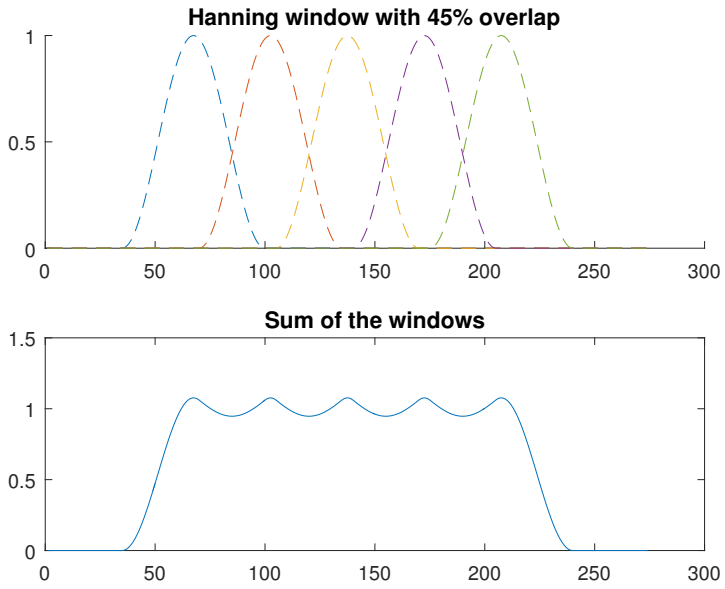

Figure 2.4: Hanning window of length 64 with 45% overlap.

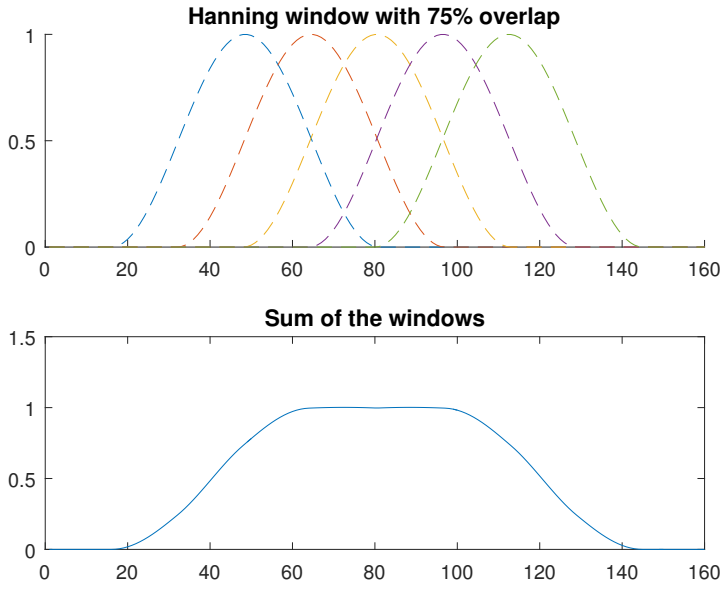

Figure 2.5: Hanning window of length 64 with 75% overlap.

2.5 Signal-to-Noise Ratio

SNR is a measurement used when comparing the level of the desired signal to the level of background noise. The SNR value is defined as

$$\mathrm{SNR} = \frac{P_{\mathrm{signal}}}{P_{\mathrm{noise}}} \tag{2.8}$$

where P_{signal} and P_{noise} is the averaged power for the signal and noise respectively. Due to the wide dynamic range of audio signals the SNR value is often shown in decibel which is defined as

$$\mathrm{SNR}_{dB} = 10 \log_{10} \left(\frac{P_{\mathrm{signal}}}{P_{\mathrm{noise}}} \right). \tag{2.9}$$

2.6 Voice Activity Detection

The purpose of VAD is to determine when speech is present in a noisy signal. The use of a VAD is important in many speech and audio processing applications. A good VAD should have a high accuracy for different SNR levels and work independently of the noisy environment [7]. Low complexity is also vital for real-time applications. Many different features can be used when implementing a VAD; short-time energy, zero-crossing rate, autocorrelation function based, spectrum based and higher order statistics based. Short-time energy and zero-crossing rate based are simple and therefore also commonly used. To get a more robust VAD, other features can be used, and even a combination of features can be used with maintained complexity to increase the robustness in different environments[7].

2.7 Spectrum Estimation

An AR model can be used to estimate the spectrum over a period of time. How much the estimation should be affected by a change in the input can be chosen depending on over how long time the spectrum should be estimated and how much smoothing that is desired.

The AR model has the form of a stochastic difference equation, depending linearly on the previous outputs and a stochastic term. The AR model is defined as

$$X(t) = c + \sum_{i=1}^{p} \varphi_i X(t-i) + \varepsilon(t) \tag{2.10}$$

where p is the order of the model, c is a constant, φ_i are the parameters which determine how much the output depends of previous outputs, and $\varepsilon(t)$ is the stochastic term. For an AR model of order p to be stable it requires that the roots of the polynomial

$$z^p - \sum_{i=1}^{p} \varphi_i z^{p-i} \tag{2.11}$$

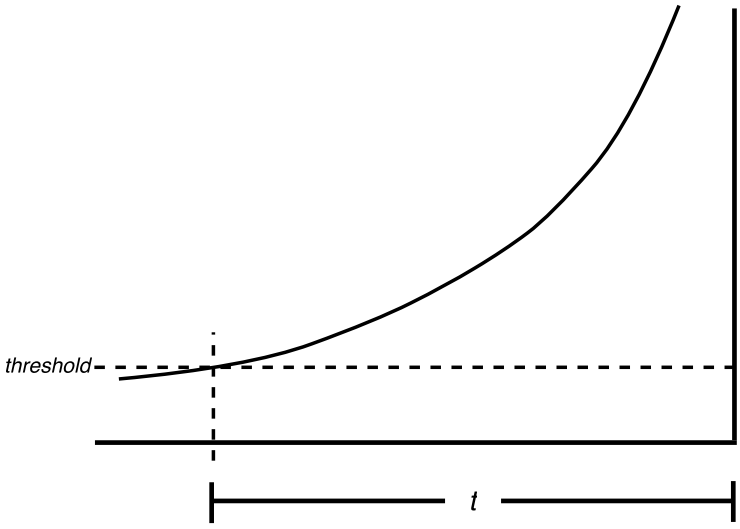

Figure 2.6: The figure shows the level of contribution of previous output values which decreases with time until it is below the chosen threshold.

lie within the unit circle. It can be seen from the definition that the previous outputs affect the current output for an infinite amount of time but the contribution will go towards zero (if the model is stable).

In this thesis a first order AR model is used which looks like

$$X(t) = \varphi X(t-1) + (\varphi - 1)\varepsilon(t). \tag{2.12}$$

The current output, $X(t)$ is dependent on a previous output with a factor $\varphi^{\text{offset in time}}$, using this and selecting a threshold ($-60dB$ for audio) where it is seen as an old output value no longer contributes to the current output, φ can be calculated to give a desired estimation time according to Figure 2.6. This can be seen as a low-pass filter of the first order. The formula for the calculation is

$$\varphi = \text{threshold}^{\text{blocksize}/t \cdot f_s} \tag{2.13}$$

where blocksize is the size of each input frame to the algorithm.

2.8 Performance Measurements

To be able to compare the different solutions other than just listening to the sound, a few different units of measurement were used. Besides the methods below, SNR improvement was also used to quantify performance of the solutions.

2.8.1 Signal Distortion

The distortion D is calculated by estimating the spectral power of the clean signal, $\hat{P}_{x_s}(\omega)$ and the processed clean signal, $\hat{P}_{y_s}(\omega)$ and integrate over the difference

$$D = \frac{1}{2\pi}\int_{-\pi}^{\pi} |C_d \hat{P}_{y_s}(\omega) - \hat{P}_{x_s}(\omega)| d\omega \tag{2.14}$$

where $\omega = 2\pi f$, and f is the normalized frequency [9]. Ideally, the distortion is zero. C_d is a normalizing constant to prevent amplification or attenuation to affect the calculation and is defined as

$$C_d = \frac{\int_{-\pi}^{\pi} \hat{P}_{x_s}(\omega) d\omega}{\int_{-\pi}^{\pi} \hat{P}_{y_s}(\omega) d\omega}. \tag{2.15}$$

2.8.2 Noise Suppression

The normalized noise suppression S_N is calculated using the spectral power estimate of the noise, $\hat{P}_{x_N}(\omega)$ and the processed noise, $\hat{P}_{y_N}(\omega)$ [9]

$$S_N = C_s \frac{\int_{-\pi}^{\pi} \hat{P}_{y_N}(\omega) d\omega}{\int_{-\pi}^{\pi} \hat{P}_{x_N}(\omega) d\omega} \tag{2.16}$$

where

$$C_s = \frac{1}{C_d}. \tag{2.17}$$

2.9 Noise Reduction

Noise reduction is a process which reduces the noise that is corrupting a clean signal. There are several noise reduction methods and in this report spectral subtraction and Wiener filter are implemented and tested.

2.9.1 Spectral Subtraction

Assume that the discrete noisy signal $y(m)$ is composed by the clean signal $s(m)$ and the disturbing noise $n(m)$ according to

$$y(m) = s(m) + n(m). \tag{2.18}$$

The magnitude spectrum of the noisy signal using the FFT can then be represented as

$$|Y(\omega)| = |S(\omega)| + |N(\omega)| \tag{2.19}$$

where $Y(\omega)$ is the spectrum of the noisy signal, $S(\omega)$ the spectrum of the clean signal and $N(\omega)$ the spectrum of the noise signal. Since $N(\omega)$ in most cases cannot be directly obtained, a time-averaged estimation of the noise spectrum

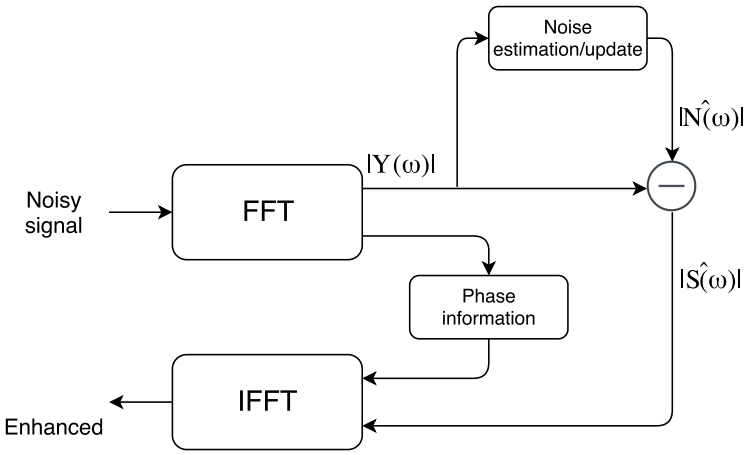

Figure 2.7: Block diagram of Spectral Subtraction.

$\hat{N}(\omega)$ is calculated over time. The estimated clean signal spectrum $\hat{S}(\omega)$ can be defined as:

$$|\hat{S}(\omega)| = |Y(\omega)| - |\hat{N}(\omega)| \tag{2.20}$$

[8]. Equation 2.20 can also be written in the following form:

$$|\hat{S}(\omega)| = H(\omega)|Y(\omega)| \tag{2.21}$$

where

$$H(\omega) = 1 - \frac{|\hat{N}(\omega)|}{|Y(\omega)|}. \tag{2.22}$$

$H(\omega)$ is often referred to as the gain function. $H(\omega)$ is always positive and assuming values between $0 \leq H(\omega) \leq 1$. It decides how much attenuation that should be done for each frequency bin in the noisy spectrum $|Y(\omega)|$.

A block diagram for spectral subtraction can be seen in Figure 2.7. As seen from Equation 2.20 the estimated noise is subtracted from the noisy signal spectrum. Due to the fact that the noise is an estimate there might be some errors after the subtraction which can lead to negative values in the spectrum of the enhanced signal. These negative values can be set to zero which is called half-wave rectification. With half-wave rectification the enhanced signal spectrum can be written as:

$$|\hat{S}(\omega)| = \begin{cases} |\hat{S}(\omega)| & \text{if} \quad |\hat{S}(\omega)| > 0 \\ 0 & \text{else} \end{cases}. \tag{2.23}$$

To finally obtain the improved time signal the IFFT is used according to:

$$\hat{s}(m) = \text{IFFT}\Big(|\hat{S}(\omega)|e^{j\theta(\omega)}\Big) \tag{2.24}$$

where $\theta(\omega)$ is the phase of the noisy signal $Y(\omega)$ and $|\hat{S}(\omega)|$ is the enhanced signal spectrum from 2.23.

Although the spectral subtraction method can be easy and straightforward to implement it comes with a few shortcomings. After the subtraction there might be small isolated peaks in the spectrum occurring at certain frequencies. However these frequencies will change over time which will create tones at different frequencies that produces an audible warble sound when converted back to the time domain. This sound is commonly referred to as musical noise. There are several factors in the spectral subtraction process that can lead to musical noise. Some of these factors are:

- Half-wave rectification 2.23. This nonlinear processing of the negative values can lead to isolated peaks in the spectrum which will distort the signal.
- Incorrect estimate of the disturbing noise which will lead to a bad estimation of the clean signal.
- Using the noisy signal's phase together with the modified spectrum to create the enhanced time signal. Since there is no enhancement of the phase of the noisy signal this can lead to some added distortions of the clean signal. However estimating the phase of the clean signal is a difficult process and greatly increases the complexity of the algorithm. Therefore using the phase of the noisy signal is considered an acceptable practice when creating the clean time signal. Furthermore the distortion of the noisy phase is small compared to other factors, especially at high SNR's. This can be seen in Figure 2.8 where the enhanced signal's, $S(\omega)$, phase is almost the same as the phase of the noisy signal $Y(\omega)$ at high SNR values, compared to low SNR values where the difference is bigger.
- Large and fast variations in the system's gain function in Equation 2.21.

There has been a lot of research throughout the years for different ways to reduce the musical noise. However it is very difficult to reduce the musical noise without affecting the clean signal and generally there is a trade-off between how much noise to suppress and how much distortions introduced by the algorithm. One approach introduced to reduce the musical noise was proposed by Berouti et al. [10]. This approach included a combination of an over-subtraction factor and a spectral flooring with the following form:

$$|\hat{S}(\omega)| = \begin{cases} |Y(\omega)| - \alpha|\hat{N}(\omega)| & \text{if } |Y(\omega)| > (\alpha+\beta)|\hat{N}(\omega)| \\ \beta|\hat{N}(\omega)| & \text{else} \end{cases} \tag{2.25}$$

where α is the over-subtraction factor ($\alpha \geq 1$), and $\beta(0 < \beta \ll 1)$ is the spectral floor parameter. The reason behind using over-subtraction factor and spectral flooring is that after subtracting the estimated noise there will still remain peaks in the spectrum. These peaks will cause musical noise as mentioned above. With over-subtraction, when $\alpha > 1$, more than the actual estimated noise will be subtracted and thus reduce the amplitude of the remaining peaks. However there will still remain deep valleys between the peaks in the spectrum and therefore

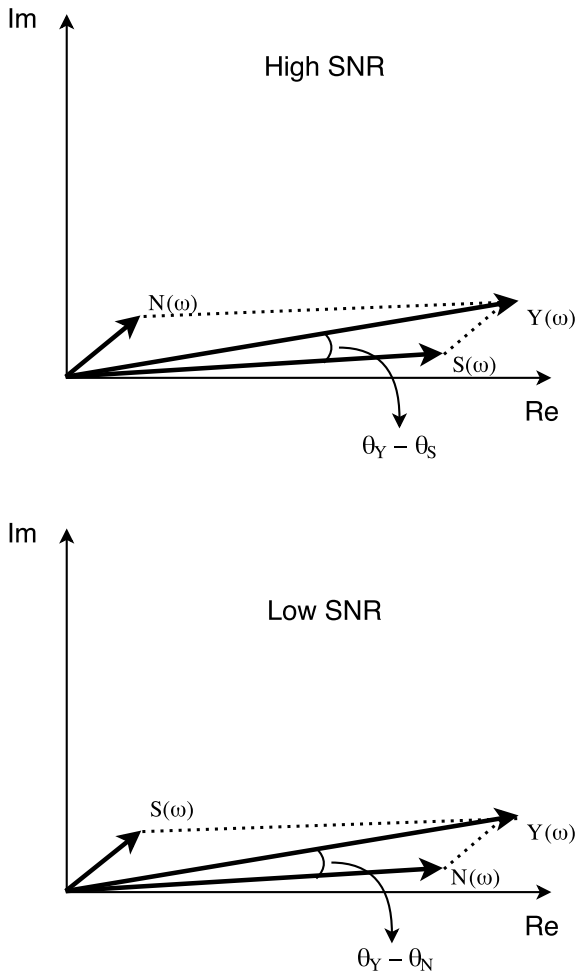

Figure 2.8: Geometric view of high and low SNR conditions.

spectral flooring is used to fill in these gaps. By doing this the depth of the valleys will decrease and hence possibly mask the remaining peaks. With these two parameters β and α it is possible to customize the spectral subtraction algorithm to fit the wanted needs. With a smaller β there will be less residual noise but the musical noise will be more audible. And vice versa a larger β will have more residual noise but the musical noise won't be audible. α affects the amount of distortion introduced to the enhanced signal. If α is too big the resulting signal will have severe distortion and with a smaller α there will be less distortion but more musical noise. Berouti et al. [10] suggested that α should vary depending on the SNR value. When the noisy signal $y(m)$ in 2.18 has a high SNR, i.e. less noise and more clean signal present, α should be smaller thus removing less noise. And when $y(m)$ has a low SNR value, i.e. lots of noise present, the α parameter should be bigger.

2.9.2 Nonlinear Spectral Subtraction

The nonlinear spectral subtraction method proposed in [11] has a few differences compared to the basic spectral subtraction algorithm described above. Instead of calculating the SNR value and α parameter for the whole frame and thus use the same α for all frequencies in that frame, the nonlinear spectral subtraction method calculates the SNR value and α parameter for each frequency bin. The reason behind this is that a lot of real world noises are colored, e.q. a fan noise or a car interior noise, and therefore affecting some frequencies more than other. Because of this there may be more noise at certain frequencies and therefore wanting to subtract more noise at those frequencies.

2.9.3 Multiband Spectral Subtraction

The motivation behind multiband spectral subtraction is similar to the one for nonlinear spectral subtraction that says that most real world noise does not have a flat frequency response but instead affecting some frequencies more than others. One disadvantage of using nonlinear spectral subtraction is that since it calculates one subtraction factor for each frequency bin it can introduce both distortion and musical noise if the SNR for some frequency bins changes radically from frame to frame. To avoid this from happening multiband spectral subtraction groups numerous frequency bins together to form one frequency band. This introduces the possibility to choose the the number of frequency bands and the size of each band. The method proposed in [12] evaluates the performance differences between one to eight bands. So the estimate of the clean signal spectrum in the ith band is obtained by:

$$|\hat{S}_i(k)| = |Y_i(k)| - \alpha_i \delta_i |\hat{N}_i(k)| \quad b_i \leq k \leq e_i \tag{2.26}$$

where b_i and e_i are the beginning and ending frequency bins of the ith frequency band, α_i is the over-subtraction factor of the ith frequency band and δ_i is a tweaking factor for the ith frequency band that can be individually set to customize the noise removal process.

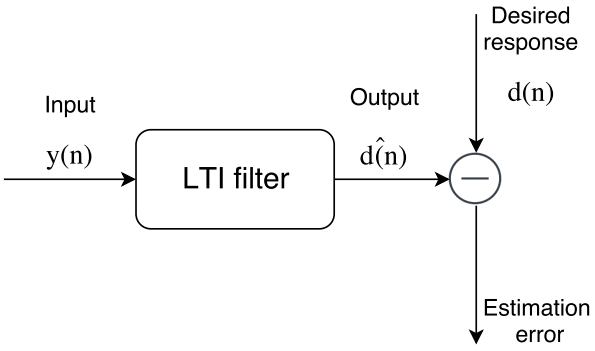

Figure 2.9: Block diagram of Wiener Filter.

2.9.4 Wiener Filter

Unlike spectral subtraction, the Wiener filter is designed to mathematically optimize the enhanced signal using the mean-square error. The estimation error $e(n)$ is the difference between the enhanced signal $\hat{d}(n)$ and the desired signal $d(n)$

$$e(n) = d(n) - \hat{d}(n). \tag{2.27}$$

The input signal goes through a LTI system which is designed in a way such that the output $\hat{d}(n)$ is as close to $d(n)$ as possible, minimizing the error. A block diagram of this can be seen in Figure 2.9. The filter that minimizes the error is the Wiener filter [8], named after the mathematician Norbert Wiener. A constraint on the filter is that it should be linear. The filter can be either FIR or IIR but FIR is the most common since it has some desired properties, such as that FIR filters are inherently stable and the solutions are linear and computationally easy to evaluate. Using a FIR filter the enhanced signal is derived by

$$\hat{d}(n) = \sum_{k=0}^{M-1} h_k y(n-k) \qquad n = 0, 1, 2, ... \tag{2.28}$$

where M is the length of the filter, h_k the filter coefficients and $y(n-k)$ the noisy input signal. The filter coefficients h_k are computed so that the estimation error $e(n)$ is minimized, they can be derived in either the time- or the frequency domain. All processing in this master thesis is done in the frequency domain so this report only covers derivation in the frequency domain.

Consider a two-sided, infinite duration filter

$$\hat{d}(n) = \sum_{k=-\infty}^{\infty} h_k y(n-k) \qquad -\infty < n < \infty \tag{2.29}$$

called the Wiener smoother filter. The output signal can be obtained by convolu-

tion in the time domain

$$\hat{d}(n) = h(n) * y(n) \tag{2.30}$$

which, transformed to the frequency domain becomes

$$\hat{D}(\omega) = H(\omega)Y(\omega) \tag{2.31}$$

where $H(\omega)$ and $Y(\omega)$ are the DTFT of $h(n)$ and $y(n)$ respectively. The estimated error for a specific frequency bin can be defined as

$$E(\omega_k) = D(\omega_k) - \hat{D}(\omega_k) = D(\omega_k) - H(\omega_k)Y(\omega_k). \tag{2.32}$$

The mean-square error is minimized in order to retrieve the optimal transfer function $H(\omega)$. The mean-square error is written as $E\big[|E(\omega_k)|^2\big]$ where $E[\cdot]$ is the expectation operator, and given by

$$\begin{aligned} E\big[|E(\omega_k)|^2\big] &= E\{[D(\omega_k) - H(\omega_k)Y(\omega_k)]^*[D(\omega_k) - H(\omega_k)Y(\omega_k)]\} \\ &= E\big[|D(\omega_k)|^2\big] - H(\omega_k)E\big[D^*(\omega_k)Y(\omega_k)\big] \\ &\quad -H^*(\omega_k)E\big[Y^*(\omega_k)D(\omega_k)\big] + |H(\omega_k)|^2E\big[|Y(\omega_k)|^2\big] \end{aligned} \tag{2.33}$$

where $[\cdot]^*$ is the conjugate operator. Replacing $E\big[|Y(\omega_k)|^2\big]$ with $P_{yy}(\omega_k)$ i.e. the power spectrum of the input $y(n)$ and $E\big[Y(\omega_k)D^*(\omega_k)\big]$ with $P_{yd}(\omega_k)$ i.e. the cross-power spectrum of $y(n)$ and $d(n)$, the mean-square error can be rewritten as

$$\begin{aligned} J_2 = E\big[|E(\omega_k)|^2\big] &= E\big[|D(\omega_k)|^2\big] - H(\omega_k)P_{yd}(\omega_k) - H^*(\omega_k)P_{dy}(\omega_k) \\ &\quad +|H(\omega_k)|^2P_{yy}(\omega_k). \end{aligned} \tag{2.34}$$

Taking the complex derivative of the mean-square error J_2 with respect to $H(\omega_k)$, setting it to zero and solving the equation yields the optimal filter $H(\omega_k)$.

$$\begin{aligned} \tfrac{\partial J_2}{\partial H(\omega_k)} &= H^*P_{yy}(\omega_k) - P_{yd}(\omega_k) \\ &= \big[H(\omega_k)P_{yy}(\omega_k) - P_{dy}(\omega_k)\big]^* = 0. \end{aligned} \tag{2.35}$$

The general form of the Wiener filter in the frequency domain is obtained by solving for $H(\omega_k)$ as follows

$$H(\omega_k) = \frac{P_{dy}(\omega_k)}{P_{yy}(\omega_k)} \tag{2.36}$$

where the cross-power spectrum $P_{dy}(\omega_k)$ is generally complex which also makes $H(\omega_k)$ complex.

The derivation of the Wiener filter $H(\omega_k)$ for a noise reduction application is described below. The noisy input signal

$$y(n) = x(n) + n(n) \tag{2.37}$$

consists of the noise signal $n(n)$ and the clean signal $x(n)$ which equals $d(n)$ compared with Equation 2.27. The signals in Equation 2.37 transformed to the frequency domain becomes

$$Y(\omega_k) = X(\omega_k) + N(\omega_k). \tag{2.38}$$

Looking back at the general form of the Wiener filter in Equation 2.36, $P_{dy}(\omega_k)$ and $P_{yy}(\omega_k)$ needs to be computed. Using Equation 2.38 and the fact that $D(\omega_k) = X(\omega_k)$, the unknown power spectra can be written as

$$\begin{aligned} P_{dy}(\omega_k) &= E\big[X(\omega_k)\big(X(\omega_k) + N(\omega_k)\big)^*\big] \\ &= E\big[X(\omega_k)X^*(\omega_k)\big] + E\big[X(\omega_k)N^*(\omega_k)\big] = P_{xx}(\omega_k) \end{aligned} \tag{2.39}$$

and

$$\begin{aligned} P_{yy}(\omega_k) &= E\big[\big(X(\omega_k) + N(\omega_k)\big)\big(X(\omega_k) + N(\omega_k)\big)^*\big] \\ &= E\big[X(\omega_k)X^*(\omega_k)\big] + E\big[N(\omega_k)N^*(\omega_k)\big] + E\big[X(\omega_k)N^*(\omega_k)\big] \\ &\quad + E\big[N(\omega_k)X^*(\omega_k)\big] = P_{xx}(\omega_k) + P_{nn}(\omega_k). \end{aligned} \tag{2.40}$$

Using these results, Equation 2.36 can be written as

$$H(\omega_k) = \frac{P_{xx}(\omega_k)}{P_{xx}(\omega_k) + P_{nn}(\omega_k)}. \tag{2.41}$$

Both power spectra above are positive and have even symmetry which means that the Wiener filter $H(\omega_k)$ is also positive, even and real. The impulse response h_k must also be even and therefore not causal why the Wiener filter is not realizable. Instead the *a priori* SNR at frequency ω_k is defined as

$$\xi_k \triangleq \frac{P_{xx}(\omega_k)}{P_{nn}(\omega_k)} \tag{2.42}$$

inserted in Equation 2.41, the Wiener filter can be expressed as

$$H(\omega_k) = \frac{\xi_k}{\xi_k + 1} \tag{2.43}$$

where $H(\omega_k)$ is in the interval [0 1], going towards zero for low SNR and towards one for high SNR. This means that the Wiener filter will perform almost no attenuation for the frequency ω_k when SNR is high and the other way around, it will attenuate heavily when SNR is low. The attenuation depending on the SNR for the Wiener filter can be seen in Figure 2.10. Two parameters α and β can be added to Equation 2.41 to create a parametric Wiener filter with possibility to control the attenuation characteristics.

$$H(\omega_k) = \left(\frac{P_{xx}(\omega_k)}{P_{xx}(\omega_k) + \alpha P_{nn}(\omega_k)}\right)^{\beta}. \tag{2.44}$$

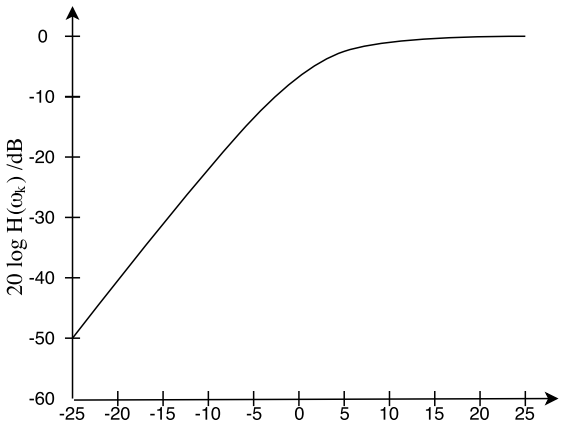

Figure 2.10: Attenuation curve of the Wiener filter as a function of the *a priori* SNR.

Substituting the power spectra with the *a priori* SNR gives

$$H(\omega_k) = \left(\frac{\xi_k}{\alpha + \xi_k} \right)^{\beta}. \tag{2.45}$$

How α and β affects the attenuation curve can be seen in Figure 2.11 and 2.12 respectively. α affects the attenuation for both low and high SNR levels whilst β affects the attenuation heavily for low SNR levels and scarcely anything for high SNR levels.

2.10 Musical Noise Reduction

Two black-box methods were implemented to reduce the musical noise which can be used together with both the different variants of spectral subtraction and Wiener filter. The first method is called Adaptive Averaging and is proposed by H. Gustafsson et al. in [13].

2.10.1 Adaptive Averaging

As mentioned above, variations in the gain function can result in musical noise. The purpose of the adaptive averaging is to reduce the variations adaptively so that the gain function is allowed to vary more when a signal is active and less during noise-only periods. The adaptive smoothing of the gain function is described as

$$\bar{H}(\omega, i) = \alpha_1(i) \cdot \bar{H}(\omega, i-1) + (1 - \alpha_1(i)) \cdot H(\omega, i) \tag{2.46}$$

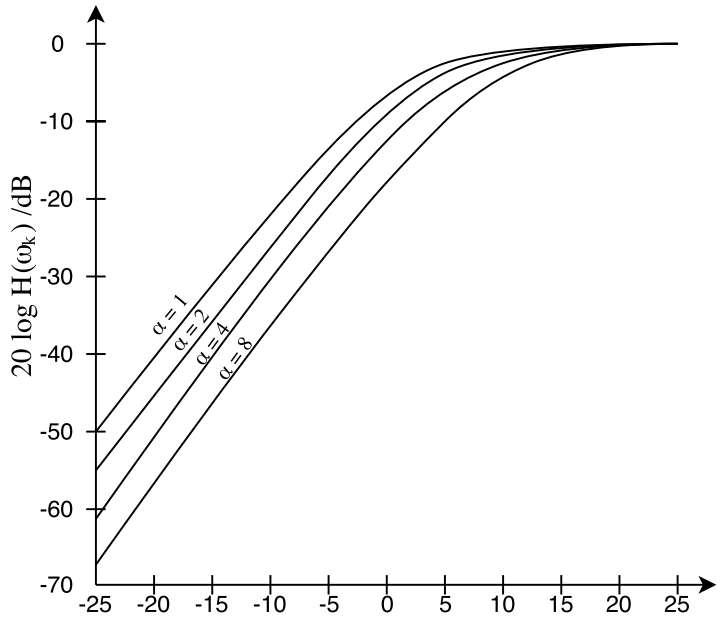

Figure 2.11: Attenuation curves of the parametric Wiener Filter depending on different values of α with fixed $\beta = 1$.

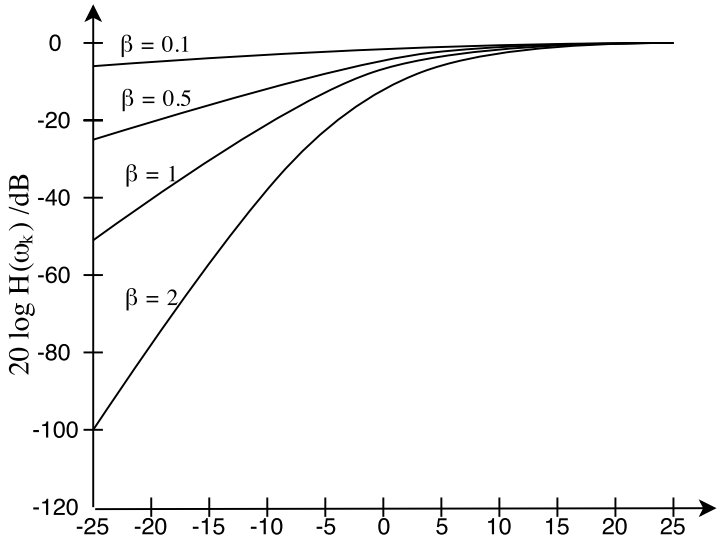

Figure 2.12: Attenuation curves of the parametric Wiener Filter depending on different values of β with fixed $\alpha = 1$.

where $\bar{H}(\omega, i)$ is the smoothed gain function, i the current frame and $\alpha_1(i)$ is the adaptive smoothing factor changing from frame to frame. $\alpha_1(i)$ is derived from

$$\alpha_1(i) = 1 - \beta(i) \tag{2.47}$$

where $\beta(i)$ is a spectral discrepancy measure. $\beta(i)$ depends on the spectrum of the current noisy frame and the noise estimation.

$$\beta(i) = \min \left\{ \frac{\sum\limits_{\omega} |P_Y(\omega, i) - \bar{P}_N(\omega, i)|}{\sum\limits_{\omega} \bar{P}_N(\omega, i)}, 1 \right\} \tag{2.48}$$

where $P_Y(\omega, i)$ is the power spectrum for the current frame and $\bar{P}_N(\omega, i)$ is the averaged estimation of the noise. When $\beta(i)$ is small, i.e. when the noise signal consists of mostly noise, $\alpha_1(i)$ will go towards one and the gain function will be averaged over a longer time. When a non-stationary signal is active, $\beta(i)$ will be big and $\alpha_1(i)$ will come close to zero and almost no averaging is applied. At the end of a non-stationary signal, the amount of averaging will increase fast which means that the previous gain functions suited for the non-stationary signal will linger and affect the background noise so that a shadow of the signal will be audible even after the signal is present. To avoid this the increase of averaging is limited

according to

$$\alpha_2(i) = \begin{cases} \gamma_c\alpha_2(i-1) + (1-\gamma_c)\alpha_1(i) & \text{if} \quad \alpha_2(i-1) < \alpha_1(i) \\ \alpha_1(i) & \text{else} \end{cases} \tag{2.49}$$

where $\alpha_2(i)$ becomes the new variable that determines the averaging of the gain function. If $\alpha_1(i)$ for the current frame is larger than $\alpha_2(i-1)$ used to smooth the previous frame its increase is limited by the constant γ_c. Now, the increase of averaging is limited but it can decrease at an unlimited rate. Equation 2.46 is consequently replaced by

$$\bar{H}(\omega, i) = \alpha_2(i) \cdot \bar{H}(\omega, i-1) + (1-\alpha_2(i)) \cdot H(\omega, i). \tag{2.50}$$

2.10.2 Postfilter

The second method is a postfilter proposed by T. Esch and P. Vary in [14]. In comparison with the previous method, Adaptive averaging which smoothes the gain function over time, this method smoothes the gain function over the spectrum. The input to the algorithm is the noisy signal and the gain function from the existing system. The power ratio of the enhanced and noisy signal, $\zeta(i)$ is calculated with

$$\zeta(i) = \frac{\sum\limits_{\omega} |G(\omega, i) \cdot Y(\omega, i)|^2}{\sum\limits_{\omega} |Y(\omega, i)|^2} = \frac{\sum\limits_{\omega} |\hat{S}(\omega, i)|^2}{\sum\limits_{\omega} |Y(\omega, i)|^2} \tag{2.51}$$

to serve as an indicator of the presence of a non-stationary signal. $G(\omega, i)$ is the gain function calculated with the existing system, $\hat{S}(\omega, i)$ is the enhanced signal according to the existing system and $Y(\omega, i)$ is the spectrum of the noisy signal. If there is a lot of non-stationary signals active, $G(\omega, i)$ will be close to one for most frequency bins and not alter the signal very much. For this case $\zeta(i)$ will also be close to one. For the other case when the noisy signal consists of mostly noise, $\zeta(i)$ will be close to zero. A threshold is introduced to decide if a signal is present or not:

$$\zeta_T(i) = \begin{cases} 1 & \text{if} \quad \zeta(i) \geq \zeta_{\text{thr}} \\ \zeta(i) & \text{else} \end{cases} . \tag{2.52}$$

When $\zeta(i) \geq \zeta_{thr}$, $\zeta_T(i)$ is set to one to indicate that a non-stationary signal is present. $\zeta_T(i)$ will later decide when and how much smoothing that is performed on the spectrum, since $\zeta_T(i)$ is dependent on the threshold, the threshold affects the trade-off between reduction of musical noise and the amount of signal distortion.

The gain function is smoothed by a moving average window, the odd length of the window $N(i)$ is calculated as

$$N(i) = \begin{cases} 1 & \text{if} \quad \zeta_T(i) = 1 \\ 2 \cdot \text{round}\left(\left(1 - \frac{\zeta_T(i)}{\zeta_{\text{thr}}}\right) \cdot \Psi\right) + 1 & \text{else} \end{cases} \tag{2.53}$$

where Ψ is a scaling factor that sets the maximum amount of smoothing. The quota in equation 2.53 provides a soft-decision for how much smoothing that is applied. Smoothing with the moving average window is equivalent to linear filtering with the impulse response

$$H_i(\mu) = \begin{cases} \frac{1}{N(i)} & \text{if} \quad \mu < N(i) \\ 0 & \text{else} \end{cases}, \ \mu \in [0, M-1] \tag{2.54}$$

where μ is the index of the frequency bin and M is the size of the FFT. The new postfiltered gain function is the original gain function convoluted with the impulse response as

$$G_{\text{PF}}(i,\omega) = |G(i,\omega)| * H_i(\mu) \tag{2.55}$$

and the new enhanced signal in the frequency domain $\hat{S}_{\text{PF}}(i,\omega)$ thus becomes

$$\hat{S}_{\text{PF}}(i,\omega) = G_{\text{PF}}(i,\omega) \cdot Y(i,\omega). \tag{2.56}$$

2.11 Filtering

In digital signal processing filtering can be used to remove unwanted frequencies of a signal. There are two types of digital filters: IIR and FIR.

2.11.1 FIR Filters

The main characteristics of FIR filters are that they have a finite length impulse response and settles to zero in finite time. This means that FIR filters are inherently stable. FIR filters can also be designed to have linear phase which is often desired in audio applications. For a casual FIR filter the output $y(m)$ can be calculated with

$$y(m) = \sum_{k=0}^{M-1} h(k)x(m-k) \tag{2.57}$$

where $h(k)$ is the impulse response of the system and M is the length of the filter. By entering the frequency domain the output can also be calculated according to

$$Y(\omega) = X(\omega)H(\omega) \tag{2.58}$$

where $H(\omega)$ is the transfer function and $X(\omega)$ and $Y(\omega)$ are the filter's input respectively output. The computational complexity for filtering in the time domain is

$$(N_h + N_x - 1)(2N_h - 1) \tag{2.59}$$

where N_h is the length of the filter and N_x the length of the input [15]. In the frequency domain the computational complexity is

$$(N_x + N_h - 1)\log(N_x + N_h - 1) + 6(N_x + N_h - 1) \tag{2.60}$$

which is less than for time domain filtering except for small filter lengths.

Chapter 3
Method

This chapter describes the the microphone selection process and how the calibration of the equalizer and noise reduction algorithms were implemented.

3.1 Selection of the Microphone

When selecting the microphone there were a few properties that had to be fulfilled. The microphone should have a frequency range from 20Hz to 20kHz, i.e. the whole audible range. The microphone should also be omnidirectional to be able to capture sound from all directions and analog to be able to be implemented in the camera, see Figure 3.4. Furthermore, the microphone should not cost more than a few dollars, but no exact cost limit was decided. After these properties were fulfilled the most important property was the SNR value for the microphone. Since surveillance cameras are often placed far from the origin of the sound the SNR value must be high to be able to capture quiet sounds from far away. The frequency response should be relatively flat but this was not the most crucial property since part of the task was to implement an equalizer. Sensitivity, AOP, PSR and PSRR are other factors to get good quality sound that also had to be considered.

To be able to find a suitable microphone many microphones and their specifications were examined, only analog and omnidirectional microphones with a frequency range from 20Hz to 20kHz were considered. When these requirements were fulfilled, the two microphones with the highest SNR values were chosen. The chosen microphones were PUI Audio POM-3535L-3-R electret microphone [4] and InvenSense ICS-40720 MEMS microphone [5]. Their specifications can be seen in table 3.1. Some of the specified values were measured for verification. The results from the measurements can be seen in the Result section. The properties that were measured were SNR, sensitivity and frequency response. These were all important factors when deciding which microphone to use and the available equipment limited the measurements to include these.

The equipment used for microphone measurements were: sound proofed chamber with built in speaker, see Figure 3.1 and 3.2, reference microphone Earthworks M30 with known flat frequency response, see Figure 3.3, computer with SpectraPlus installed [2], AP audio analyzer [3], dB-meter, external sound card and amplifier.

	ICS-40720	**POM-3535L-3-R**
SNR	70dBA	> 68dB
Sensitivity (dBV	-32 ± 2	-34 ± 4
AOP (dB SPL)	124	N/A
THD (%)	0.6	N/A
EIN (dBA SPL)	24	N/A
PSR (dBV A-weighted)	-77	0
PSRR (dB)	-45	0

Table 3.1: Specifications for the InvenSense ICS-40720 MEMS and PUI Audio POM-3535L-3-R electret microphone.

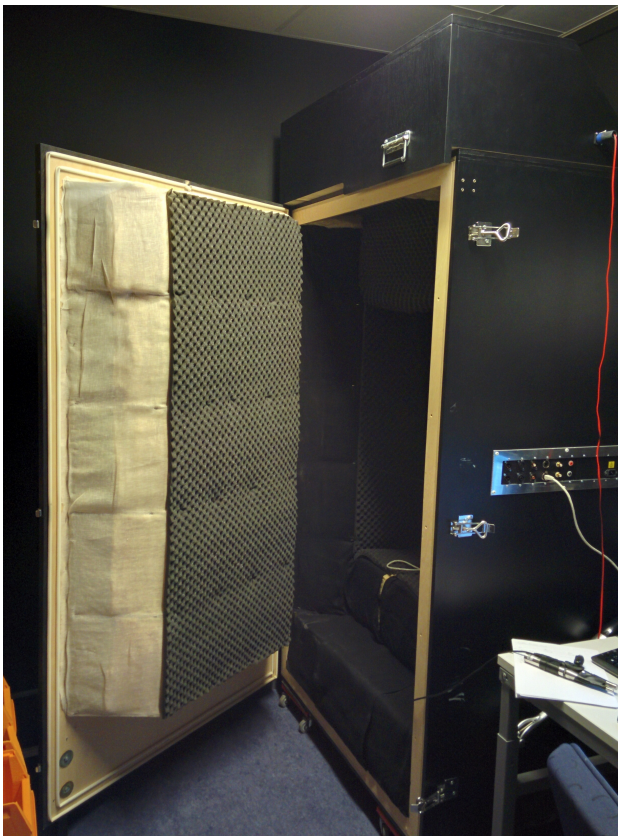

Figure 3.1: Sound proofed chamber with build in speaker.

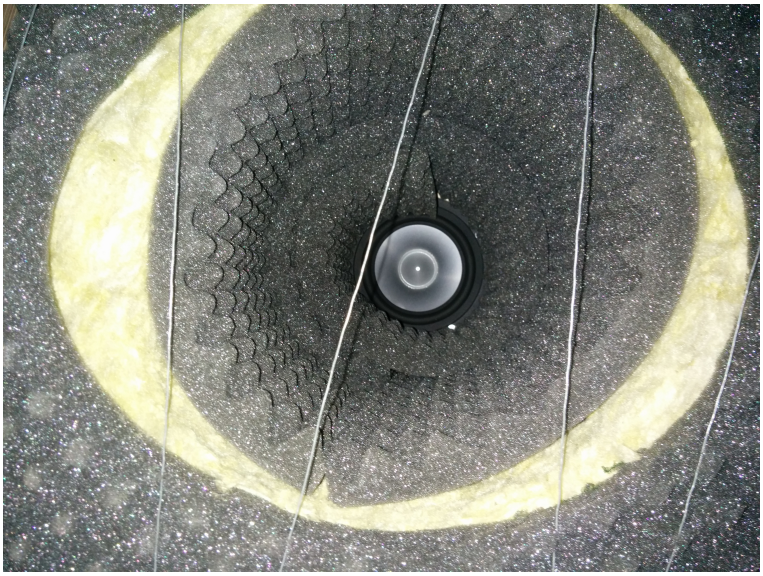

Figure 3.2: Speaker element in the sound chamber.

Figure 3.3: The reference microphone Earthworks M30 used for compensating for acoustics.

To measure the sensitivity the speaker was calibrated to 94dB when playing 1kHz sine tone with the dB-meter at the same placement as the microphone would be [1]. The microphones were powered by batteries to get clean supply voltages. The microphone was placed in the sound chamber and connected to the AP and the output voltage was measured.

SNR was achieved by measuring the output voltage during silence and calculated as

$$\mathrm{SNR} = 20 \cdot \log_{10}\left(\frac{V_{\mathrm{signal}}}{V_{\mathrm{silence}}}\right). \tag{3.1}$$

To measure the frequency response of the microphone, the acoustics in the sound chamber had to be compensated for. The frequency response of the sound chamber was measured with the reference microphone and saved in SpectraPlus to be used as a calibration for later when using the actual microphone. White noise was played through the speaker and the response was averaged to get a better result. This procedure was then repeated with the microphone using the calibration for the sound chamber.

3.2 Equalizer

Equalizers are used in audio applications to amplify or attenuate certain frequencies. There are two scenarios when equalization is helpful, to compensate for acoustics and flatten the frequency response, or to modify the frequency response to a desired result, e.g. amplifying low frequencies in order to hear speech better.

3.2.1 Problem Identification

Since the microphones in Axis cameras are inside a case, the sound that reaches the microphone has been affected by both the acoustics of the case and the channel that leads the sound to the microphone. The camera used in this master thesis is Q1775 which can be seen in Figure 3.4. The microphone itself is also unlikely to have a flat frequency response. This means that the sound from the source is different compared to the sound that is presented to the costumer. To compensate for this an equalizer is used to shape the signal to be as close to the original source as possible.

A simple GUI has also been implemented in Matlab to let the user shape the frequency response to his or hers preference.

3.2.2 Calibration

To calibrate for the acoustics and the microphone's characteristics, the camera is placed in a sound chamber where white noise is played through a speaker and the frequency spectrum is recursively averaged according to equation 2.12. By doing this it is possible to compare the source spectrum played through the speaker with the spectrum received from the microphone, according to:

$$Y(\omega) = X(\omega) \cdot H(\omega) \tag{3.2}$$

Figure 3.4: Axis camera Q1775.

where $X(\omega)$ is the source spectrum fed to the speaker, $H(\omega)$ the camera, speaker and surroundings total transfer function and $Y(\omega)$ the spectrum of the sound presented to the user. To compensate for the fact that the sound chamber and speaker have their own acoustic properties which affects the measurement, a first base calibration is measured using a reference microphone which has a flat frequency response, according to:

$$X_{\text{calibrated}}(\omega) = X(\omega) \cdot H_{\text{base calibration}}(\omega) \tag{3.3}$$

where $\text{H}_{\text{base calibration}}(\omega)$ is the inverse of the speaker and surroundings transfer function. This base calibration will eliminate the external factors from the result. After this the same white noise is played for the camera using the base calibration and this time another calibration is measured that will represent the frequency response for the camera alone, according to:

$$Y_{\text{camera}}(\omega) = X_{\text{calibrated}}(\omega) \cdot H_{\text{camera}}(\omega) \tag{3.4}$$

where $H_{\text{camera}}(\omega)$ is the transfer function for the camera and $Y_{\text{camera}}(\omega)$ is the output spectra from the camera. To compensate for the camera's transfer function its inverse is calculated according to:

$$H_{\text{equalizer}}(\omega) = H_{\text{camera}}(\omega)^{-1} \tag{3.5}$$

where $H_{\text{equalizer}}(\omega)$ is the final filter used to equalize the incoming signals according to:

$$Y_{\text{enhanced}}(\omega) = H_{\text{equalizer}}(\omega) \cdot Y(\omega) \tag{3.6}$$

The setup for this can be seen in Figure 3.5, but for simplicity the base calibration is not included.

3.3 Noise Reduction

A noise reduction algorithm is created to reduce the noise. What is considered noise can differ from case to case. A noise reduction algorithm created for a phone

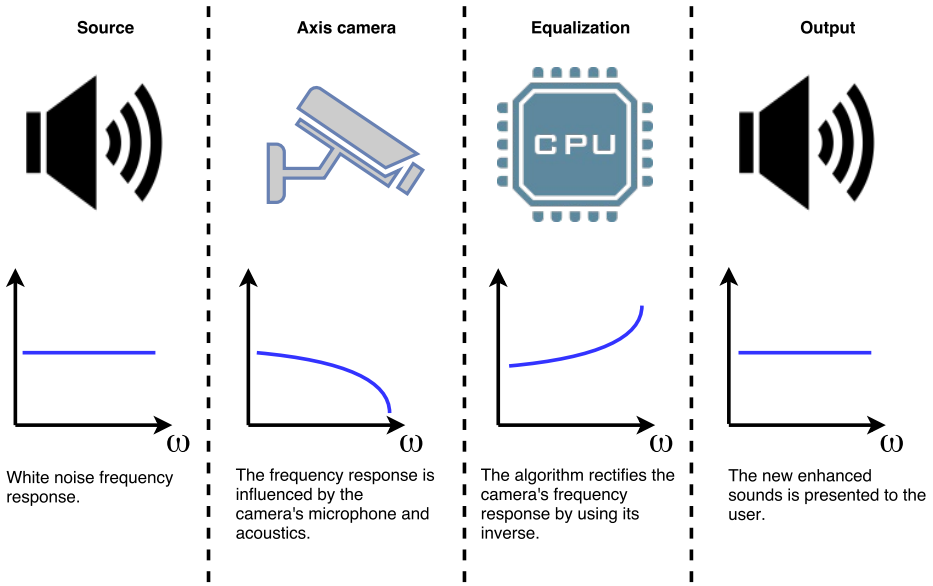

Figure 3.5: The camera's calibration process.

could for example mean that everything that is not speech should be reduced. This section describes the purpose and implementation of the noise reduction.

3.3.1 Problem Identification

Axis cameras are placed in a variety of environments and many have poor sound conditions. A camera could for example be placed near a road with traffic, high up in the ceiling close to ventilation or at a train station. Background noise can mask sounds of interest but also be tough to listen to for long periods of time, why they are good reasons to try and remove the noise.

The implemented noise reduction algorithm should remove stationary noise and still trying to preserve the wanted signal as close to the original as possible. Short disturbing noises, such as a car passing by, should not be removed or attenuated. However disturbing noise that lasts for a longer period of time should be attenuated.

3.3.2 Spectral Subtraction

Three different variants of spectral subtraction were implemented and compared. These were basic-, nonlinear- and multi-band spectral subtraction, the basic concept of these are described in the Theory section of this report. A block-diagram over the algorithms can be seen in Figure 3.6

The amount of noise that is removed is based on the SNR which means that less noise will be removed while non-stationary signals are active. If too much noise is removed during silent periods with only stationary noise active it will give

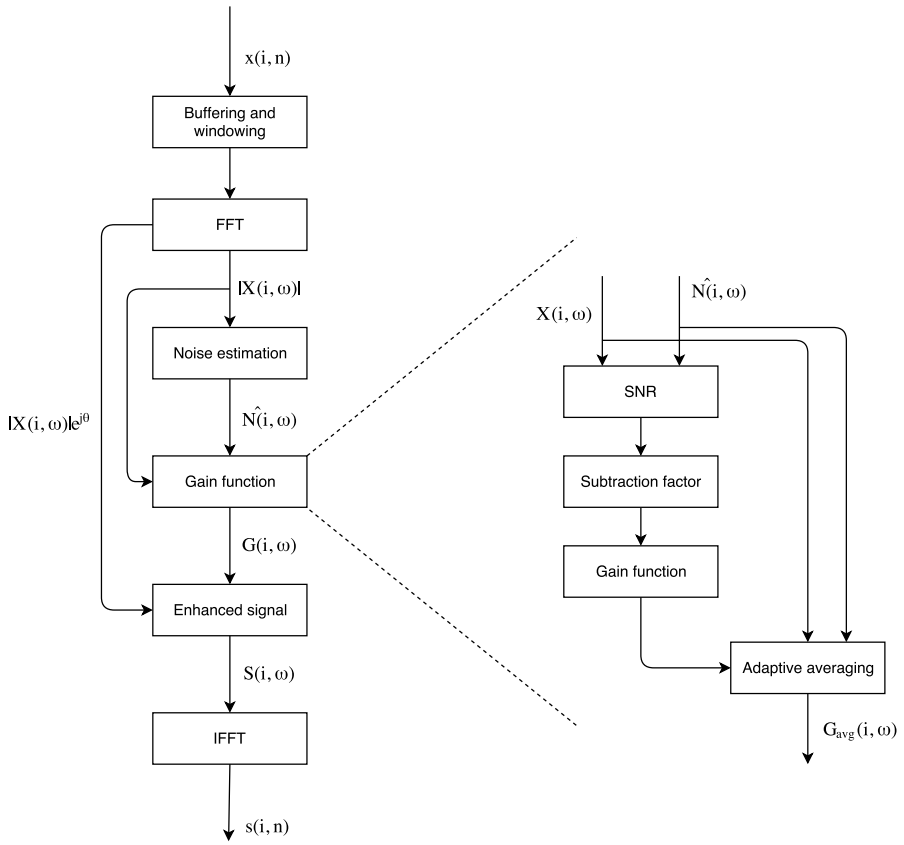

Figure 3.6: Block-diagram over the Spectral Subtraction algorithms.

the impression of signal distortion when a signal becomes active. The reason is that the noise removed will then be less and therefore become audible again. For this reason the attenuation is limited to 15dB according to

$$H(\omega) = \max(H(\omega), 10^{-15/20}) \tag{3.7}$$

where $H(\omega)$ is the gain function. This will also slightly flatten the gain function and set a floor at the maximum attenuation level, see Figure 3.7 and 3.8. Both non-linear- and multi-band spectral subtraction have better potential of reducing more noise when a non-stationary signal is active since the amount of noise subtracted is dependent on the frequency which means that if the spectra doesn't overlap, maximal reduction can still be performed on the frequencies where there is only noise.

The subtraction factor which controls how much of the noise that should be removed is calculated the same way for all three variants of spectral subtraction but with the difference that it takes different frequency bands as input. The

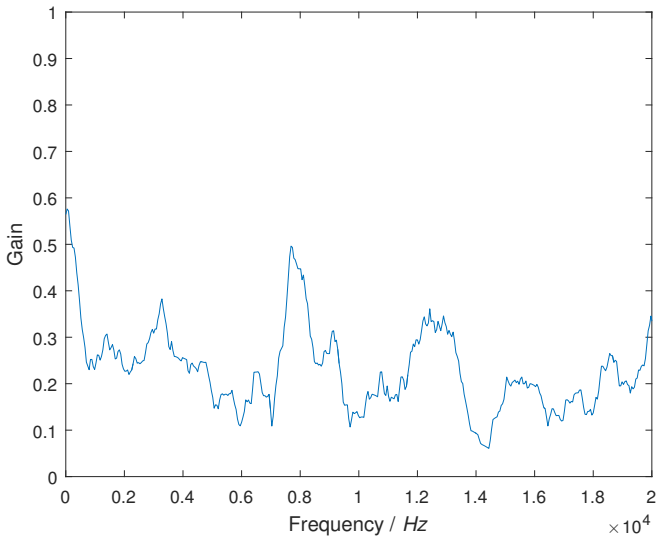

Figure 3.7: The gain function at a specific frame for a noisy signal without any limitations on the attenuation.

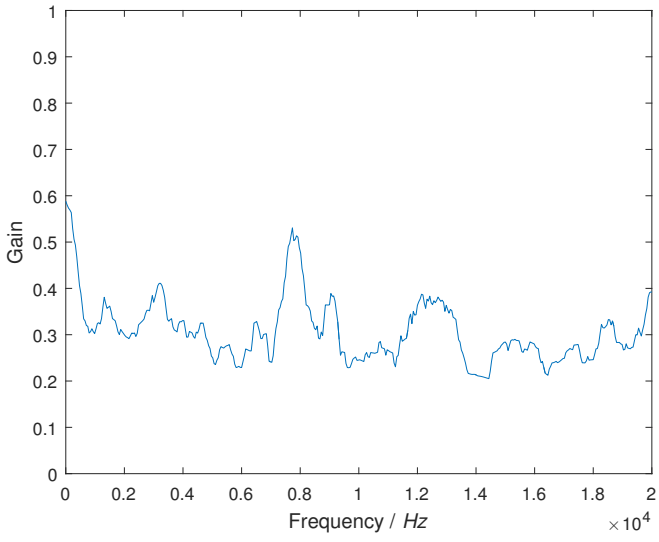

Figure 3.8: The gain function at a specific frame for a noisy signal with attenuation limited to -15dB.

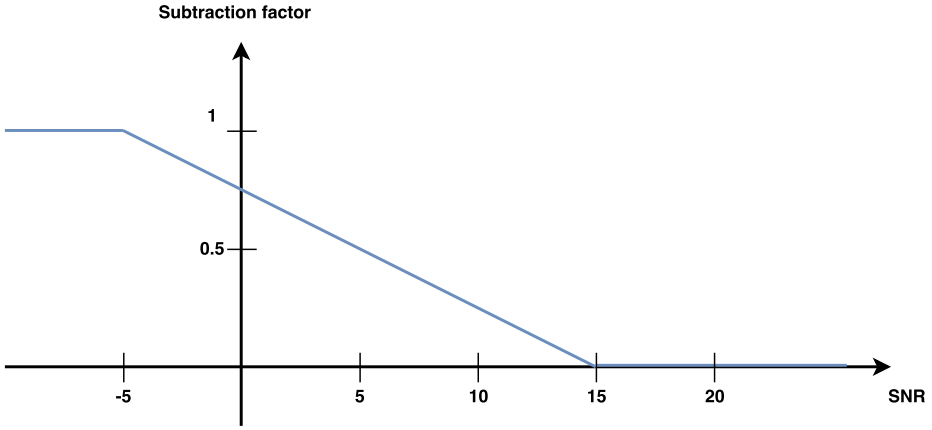

Figure 3.9: The variation of the subtraction factor as a function of SNR.

Frequency Band	Frequencies (Hz)
1	1 - 256
2	256 - 1024
3	1024 - 3064
4	3064 - 6024
5	6024 - 12000
6	12000 - 24000

Table 3.2: The frequency bands placement for the multi-band spectral subtraction.

subtraction factor α varies with SNR for the current band as follows

$$\alpha = \begin{cases} 1 & \text{if} \quad \text{SNR} < -5 \\ \frac{\text{SNR}-(-5)}{20} & \text{if} \quad -5 \leq \text{SNR} \leq 15 \\ 0 & \text{if} \quad 15 < \text{SNR} \end{cases} \tag{3.8}$$

which also is visualized in Figure 3.9. α varies between zero and one. When α is zero no noise is attenuated and when α is one the full value of the noise estimated is subtracted. Since the attenuation is limited to 15dB it was found that oversubtraction that is used in [10] i.e. subtracting more than the noise was unsuitable. Both the number of bands and their placement for the multi-band spectral subtraction method was evaluated and empirically set according to Table 3.2.

To reduce the amount the musical noise, adaptive smoothing is applied on the gain function with the method from [13]. This method was originally created for

speech enhancement and since it is, in this case used for stationary noise reduction, small modifications were by experiment found to give a better result. Because the focus only lies in reducing the stationary noise, more smoothing can be applied during noise-only periods. Equation 2.48 were modified to

$$\beta(i) = \min\left\{\left(\frac{\sum_{\omega}|Y(\omega,i) - \hat{N}(\omega,i)|}{\sum_{\omega}\hat{N}(\omega,i)}\right)^{3}, 1\right\} \tag{3.9}$$

where $Y(\omega, i)$ is the spectrum of the noisy signal and $\hat{N}(\omega, i)$ is the estimation of the noise. The addition of the exponent boost the amount of averaging for low SNR periods but doesn't affect high SNR periods that much. This was found to yield less musical noise during noise-only periods and still have little influence where non-stationary signals were active. The use of magnitude spectrum instead of power spectrum also increased the amount of smoothing which was desired.

The postfilter from [14] was also implemented to reduce the musical noise. The reduction was noticeable but the signal distortion increased so much that it was decided to not use the postfilter in the final implementations.

3.3.3 Wiener Filter

The Wiener filter method is quite similar to the spectral subtraction method. Wiener filter uses the *a priori* SNR to calculate the filter, since this is not available in a real-time system it is estimated. To estimate the *a priori* SNR, first the *a posteriori* SNR is estimated. The *a posteriori* SNR is simply calculated using Equation 2.8 where the powers are calculated as the sum of all squared samples. The *a posteriori* SNR is subtracted with one and half-wave rectified. Then the *a priori* SNR ξ is calculated as

$$\xi(i) = \varphi H(\omega, i-1)^2 \mathrm{SNR}_{\mathrm{posteriori}}(i-1) + (1-\varphi)\mathrm{SNR}_{\mathrm{posteriori}}(i) \tag{3.10}$$

where φ is a smoothing factor and $H(\omega, i-1)$ is the previous Wiener filter. The current Wiener filter that is going to be used is finally calculated with Equation 2.45. The parameters α and β were chosen by experiment to achieve the desired attenuation, the values were set to 1 and 0.47 respectively. The Wiener filter is smoothed in two steps, first with recursive averaging where the amount of smoothing is constant and independent. Then adaptive smoothing is also applied to the Wiener filter in the same way with the same parameters as for the spectral subtraction methods.

3.3.4 Performance Measurements

To be able to measure the performance for the various algorithms three identical systems were implemented but with different input signals, which can be seen in Figure 3.10. System 1 in Figure 3.10 was the normal noise reduction algorithm with a noisy input signal and the enhanced signal as output. System 2 and 3 had only the clean respective noise as input signals but with the exact same gain function

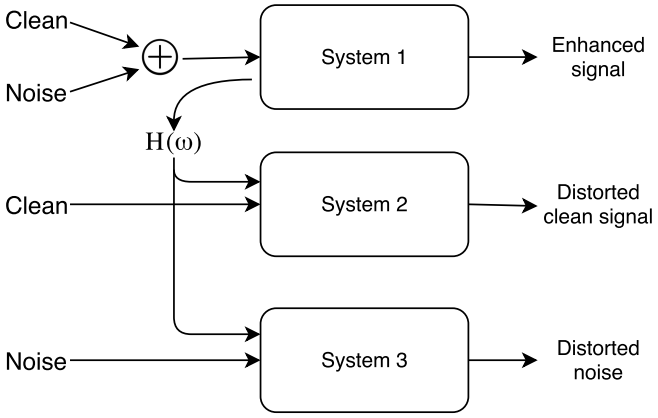

Figure 3.10: Three identical systems with different input signals.

$H(\omega)$ as System 1. By using the equations described in Section 2.8 together with the systems and their output signals in Figure 3.10 values were calculated that described how well the algorithms performed. Before the signals were sent to the systems the amplitude of the signals were adjusted to a predetermined SNR value. To make this easier the power of the noise signals P_{noise} were first normalized to one by adjusting the noise signals $n(m)$ according to

$$n_{\text{normalized}}(m) = \frac{n(m)}{\sigma} \tag{3.11}$$

where σ is the standard deviation of $n(m)$. After this the gain factor γ was calculated using

$$\gamma = \frac{P_{\text{signal}}}{10^{\frac{\rho}{10}}} \tag{3.12}$$

where P_{signal} is the power of the clean signal, ρ the predetermined SNR value in dB and γ the gain factor. Then the amplitudes of the noise signals were adjusted with γ for each clean signal to obtain the predetermined SNR value. When calculating the SNR improvement for each noise reduction algorithm the silent parts of the clean signal had to be removed. Otherwise P_{signal} in Equation 2.9 would be equal to zero during the silent parts which would lower the SNR value. To decide where the silent parts were the amplitude of the signal had to be under a threshold for a given number of samples consecutively.

Chapter 4
Results

In this chapter the measured results for the microphone, equalizer and noise reduction methods are presented.

4.1 Microphone

The two microphones that were selected, the PUI Audio POM-3535L-3-R electret microphone [4] and InvenSense ICS-40720 MEMS microphone [5] were measured to determine that their given specifications were consistent with their actual values. The measured values can be seen in table 4.1. The measured frequency responses for the MEMS and electret microphone can bee seen in Figure 4.1 and 4.2 respectively.

4.2 Equalizer

The frequency response of the camera before and after equalization can be seen in Figure 4.3 and 4.4 respectively. In both figures the frequency response is calibrated using the frequency response recorded with the reference microphone so that it is only the camera's acoustics which affects the frequency response, see Figure 4.5.

	ICS-40720	**POM-3535L-3-R**
Signal (mV_{RMS})	27.1	18.1
Noise (μV_{RMS})	9.7	6.30
SNR (dB)	68.92	69.17
Sensitivity (dBV)	−31.34	−34.85

Table 4.1: Sensitivity and SNR for the InvenSense ICS-40720 MEMS and PUI Audio POM-3535L-3-R electret microphone.

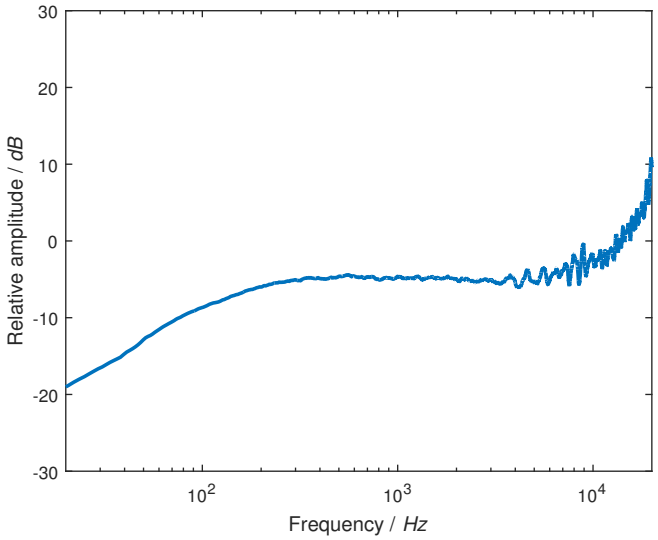

Figure 4.1: Frequency response for the Invensense ICS-40720 MEMS microphone.

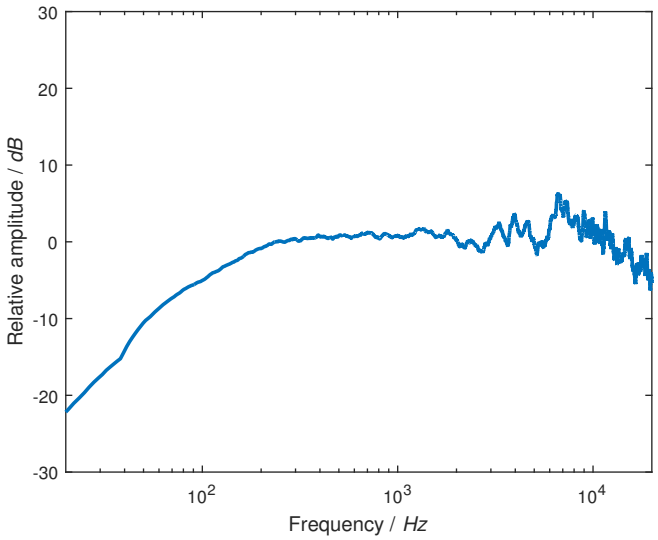

Figure 4.2: Frequency response for the PUI Audio POM-3535L-3-R electret microphone.

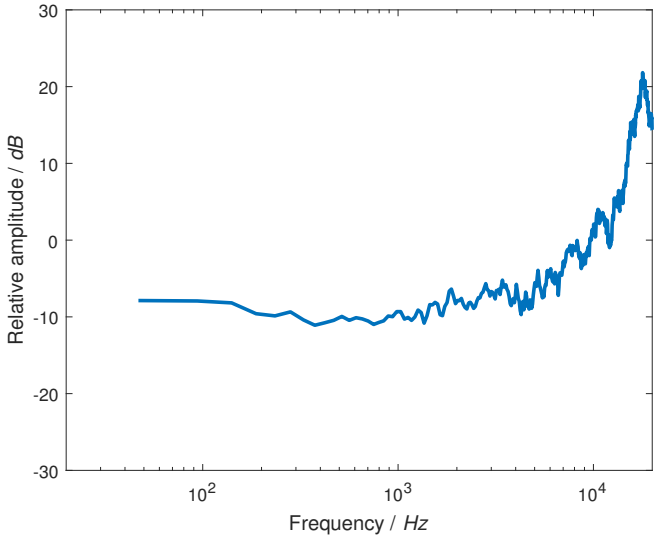

Figure 4.3: Frequency response for the camera calibrated with the frequency response from the reference microphone, the blocksize is 1024.

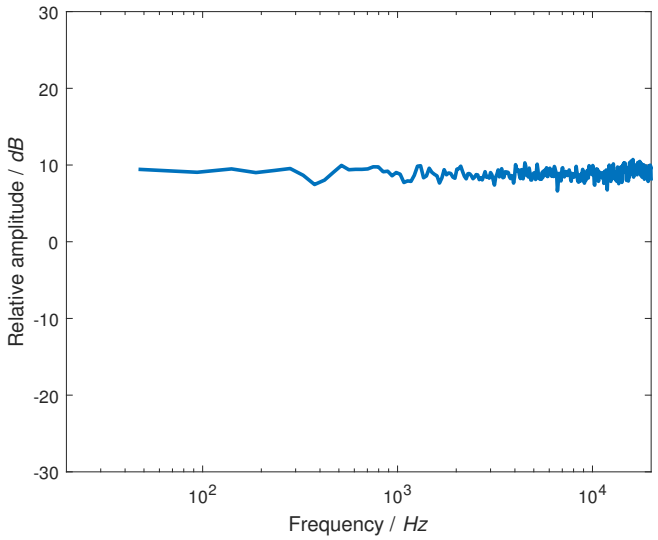

Figure 4.4: Frequency response for the camera equalized and calibrated with the frequency response from the reference microphone, the blocksize is 1024.

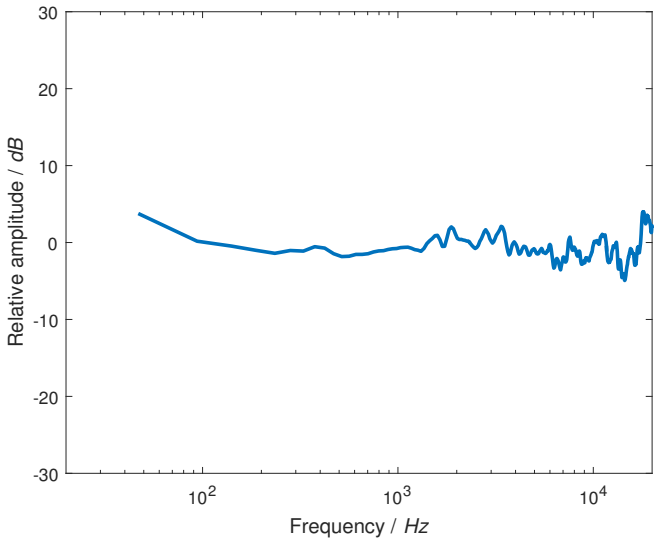

Figure 4.5: Frequency response for the reference microphone, the blocksize is 1024.

4.3 Noise Reduction

To calculate the performance for each noise reduction algorithm different noises and clean sounds were used. The performance and memory usage for the multi-band spectral subtraction method is also presented.

4.3.1 Achieved result for the noise reduction methods

The selected noise sounds can be seen in Appendix, section A.1.2, and the selected clean sounds can be seen in section A.1.1. In total 22 different clean sounds and 12 different noise sounds were used as test inputs. Both the clean and noise sounds were sounds that are usually picked up by a camera, e.g. speech, car alarm, traffic noise and fan noise. First a single clean signal with a single noise signal was tested. The result from this can be seen in Figure 4.6 - 4.10. Then several clean sounds were chosen where each clean sound was tested with all the different noise signals. The mean and the confidence interval was plotted and the results were similar for all clean signals. In Figure 4.11 - 4.14 the results be seen for Speech (Clean sound 2 from A.1.1) together with all noises and in Figure 4.15 - 4.18 the results can be seen for car alarm together with all noises. Finally all clean signals were tested together with all noise signals in one plot which can be seen in Figure 4.19 - 4.22 for all different noise reduction methods, and a summary of this can be seen in Table 4.2.

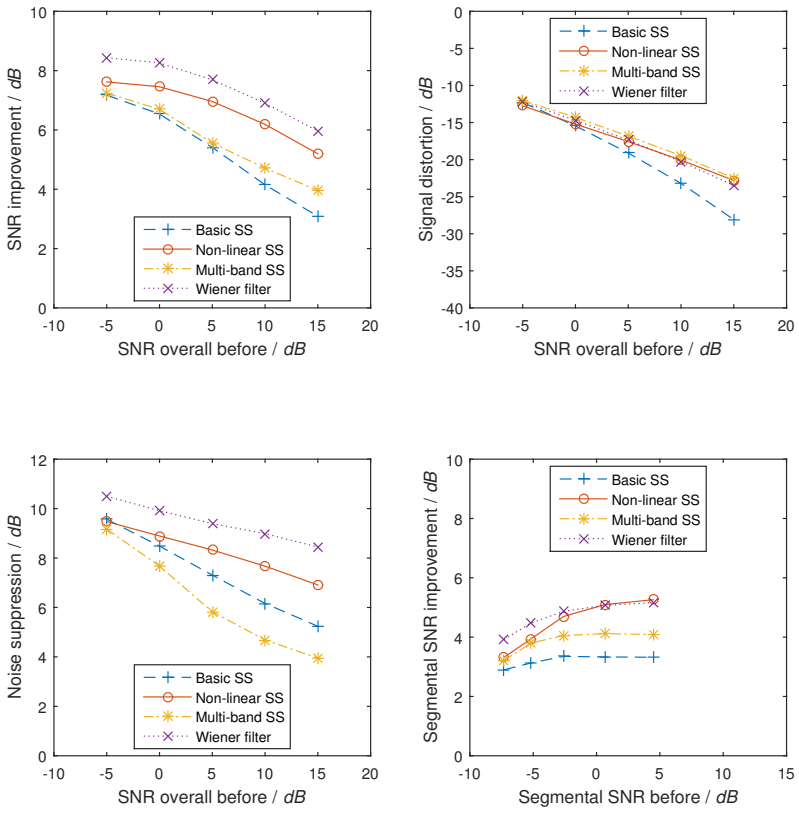

Figure 4.6: Gun shots with electric fan noise.

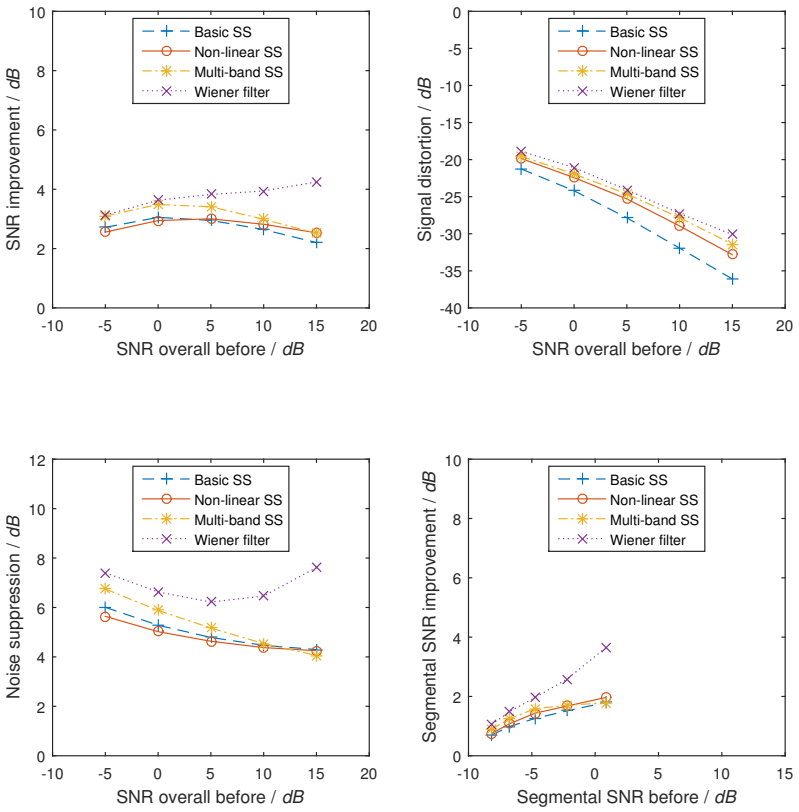

Figure 4.7: Hammering sound with highway noise.

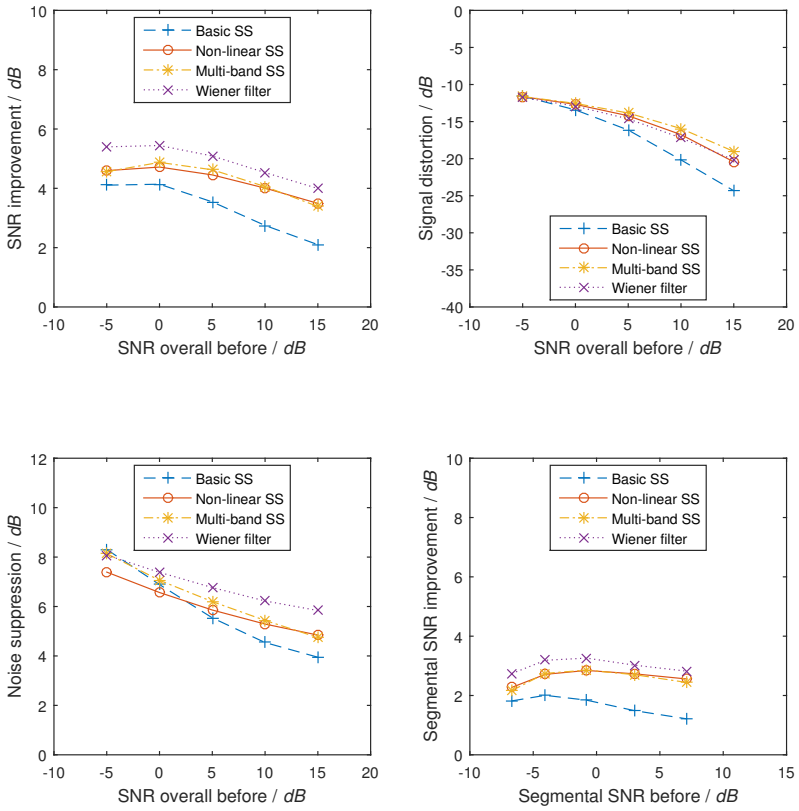

Figure 4.8: Speech (Clean sound 2 from A.1.1) with PSU switching noise.

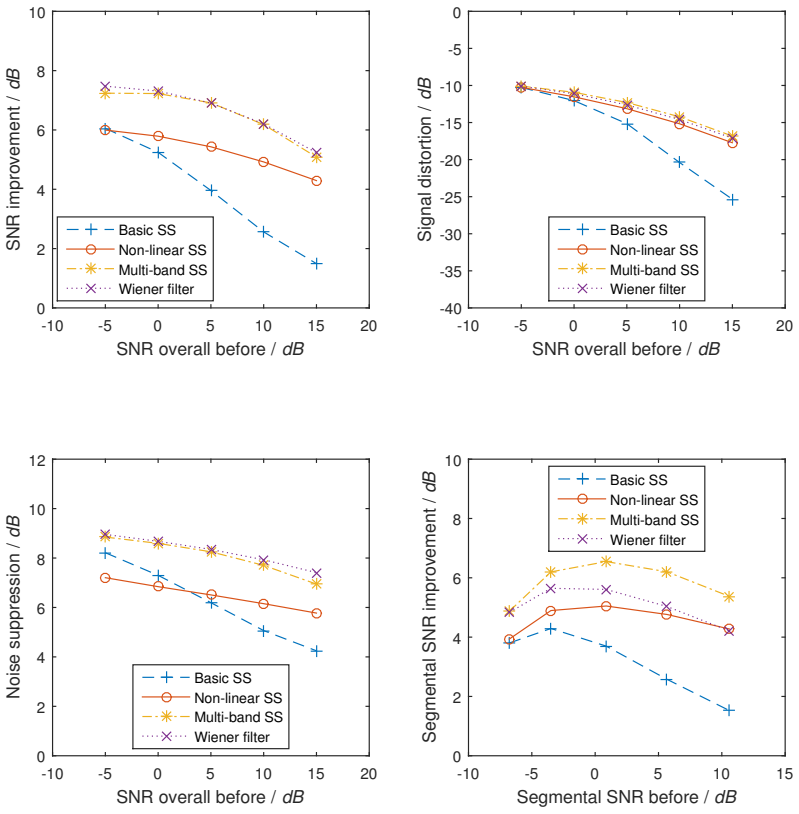

Figure 4.9: Explosion with vacuum noise.

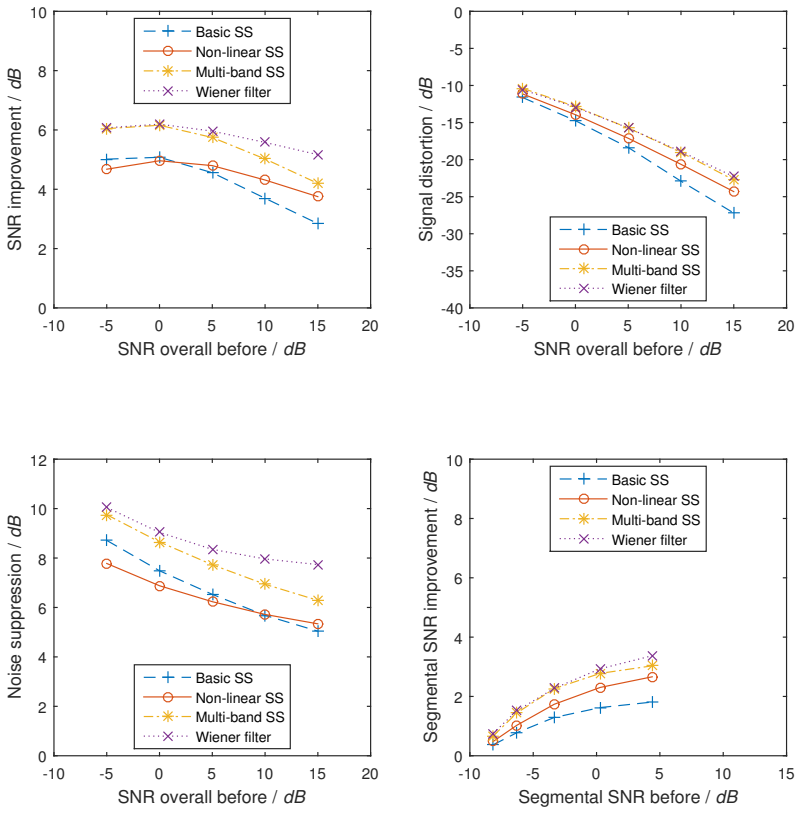

Figure 4.10: Window breaking with white noise.

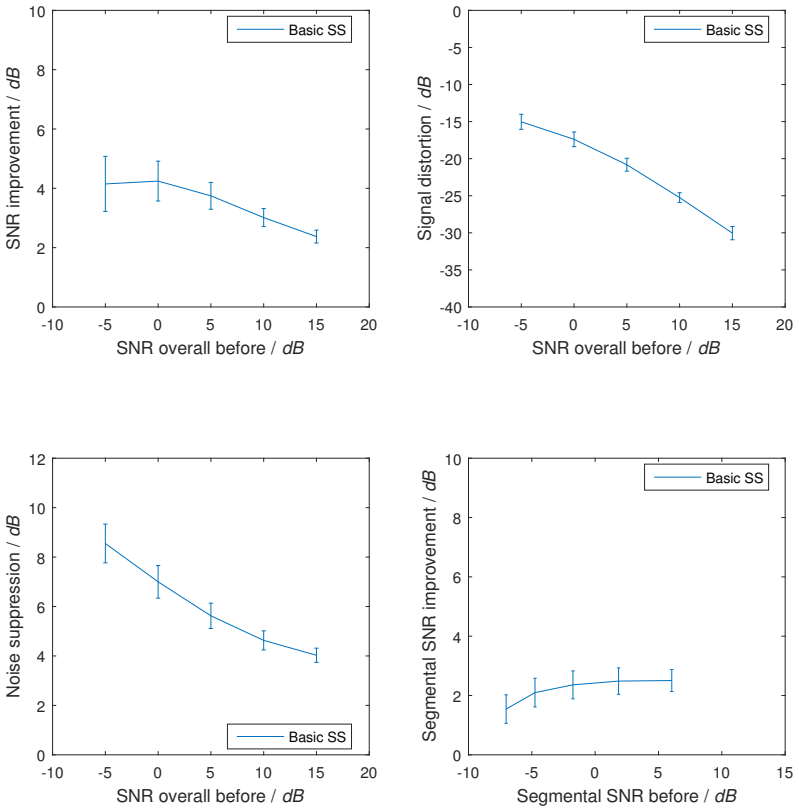

Figure 4.11: Mean and confidence interval for basic spectral subtraction with speech (Clean sound 3 from A.1.1) with all noises.

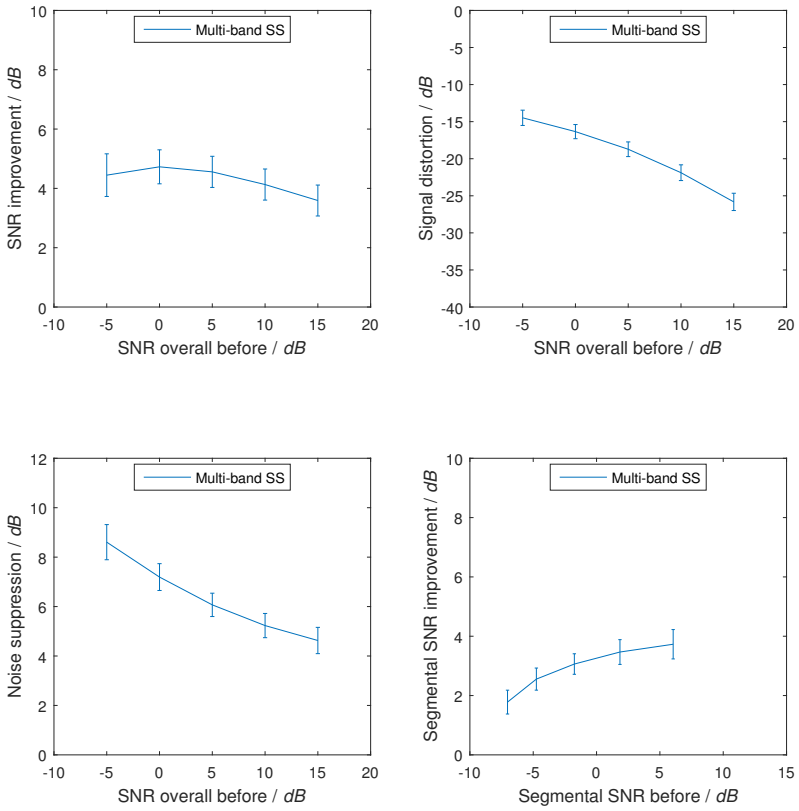

Figure 4.12: Mean and confidence interval for multi-band spectral subtraction with speech (Clean sound 3 from A.1.1) with all noises.

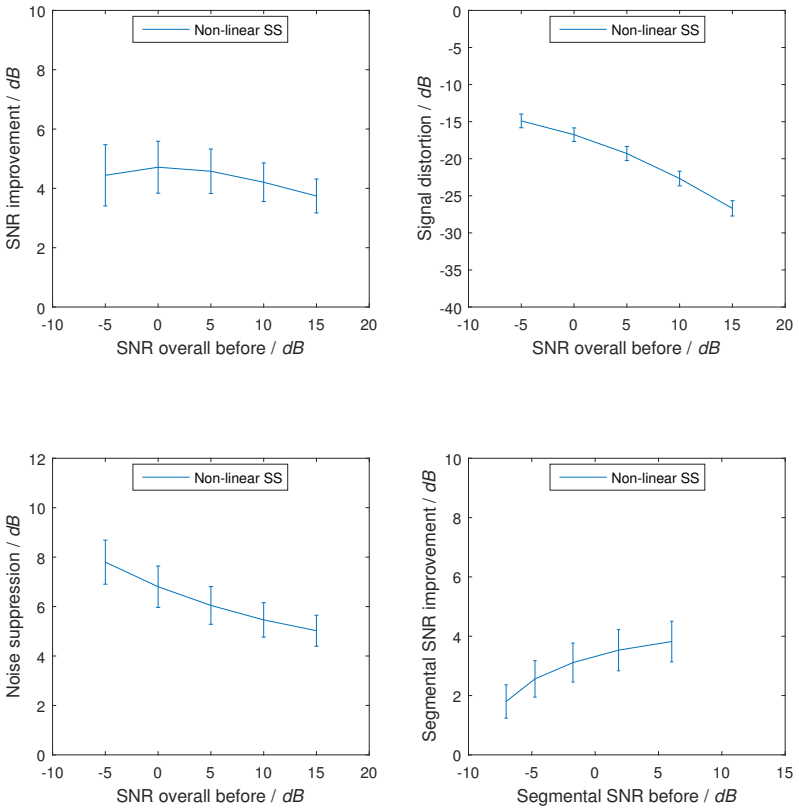

Figure 4.13: Mean and confidence interval for non-linear spectral subtraction with speech (Clean sound 3 from A.1.1) with all noises.

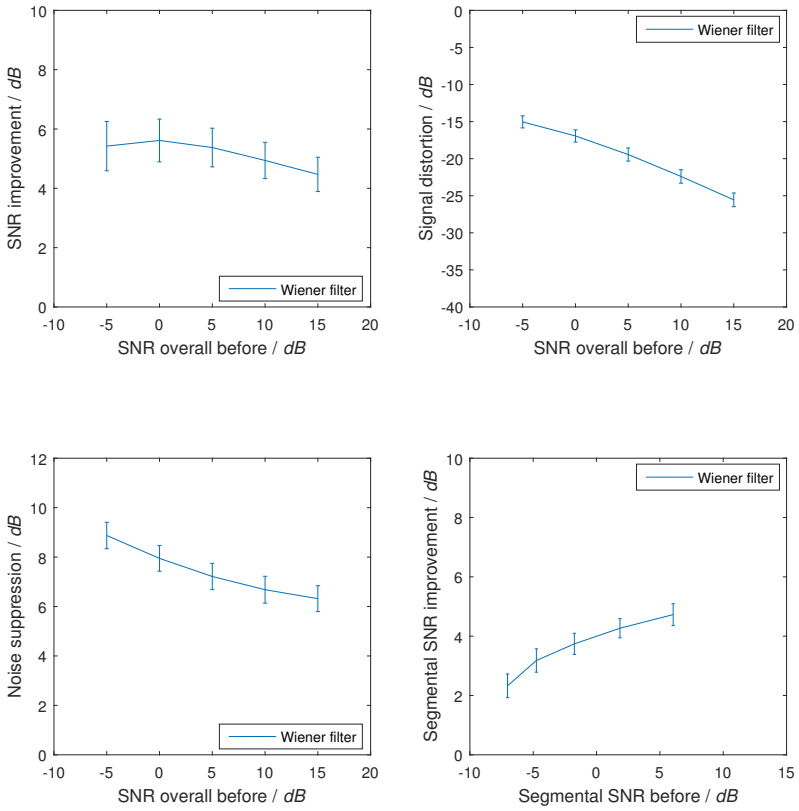

Figure 4.14: Mean and confidence interval for Wiener filter with speech (Clean sound 3 from A.1.1) with all noises.

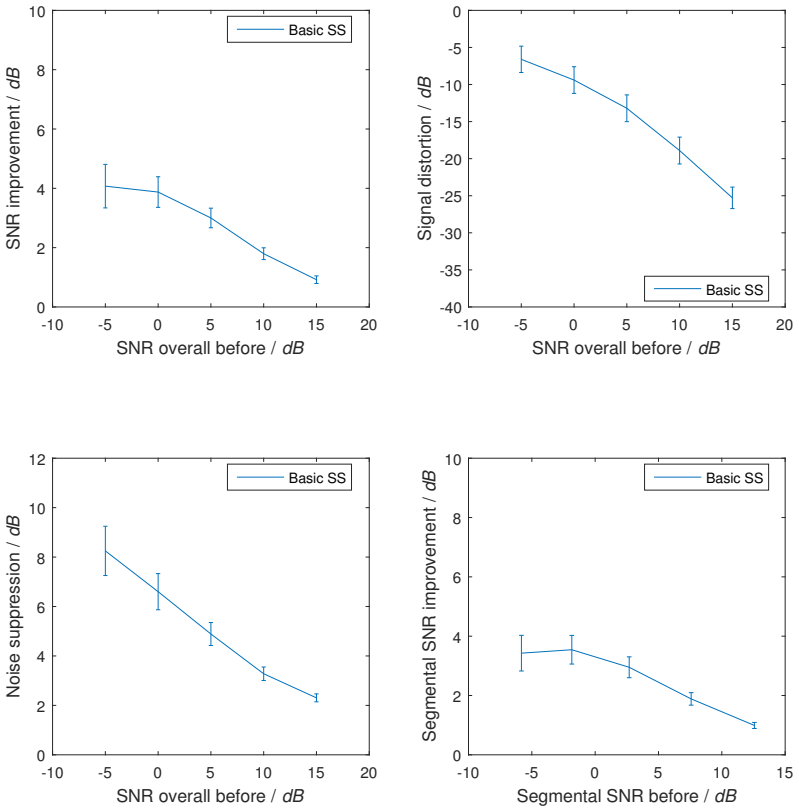

Figure 4.15: Mean and confidence interval for basic spectral subtraction with car alarm with all noises.

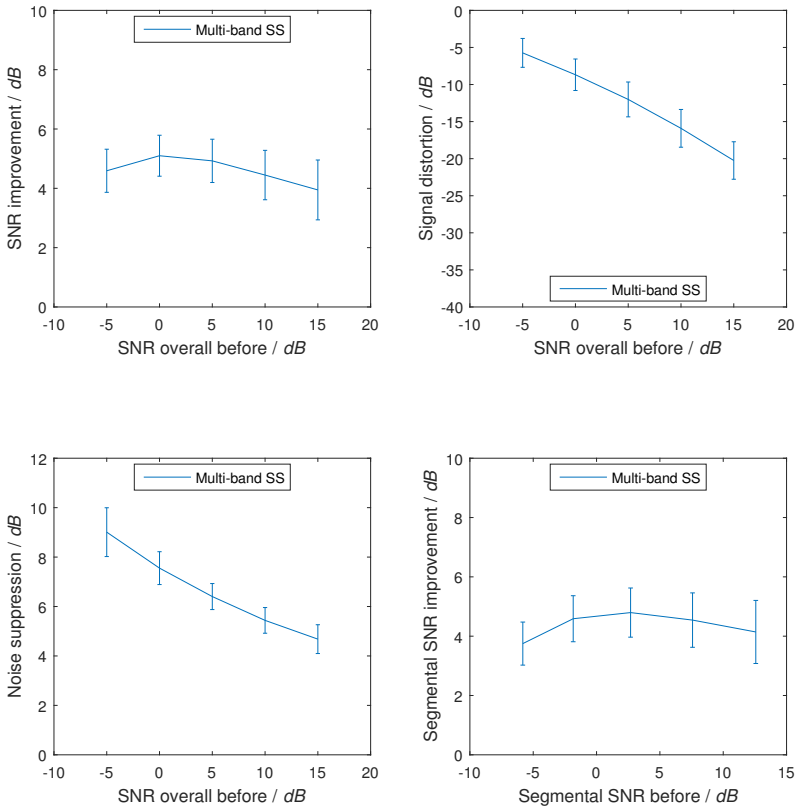

Figure 4.16: Mean and confidence interval for multi-band spectral subtraction with car alarm with all noises.

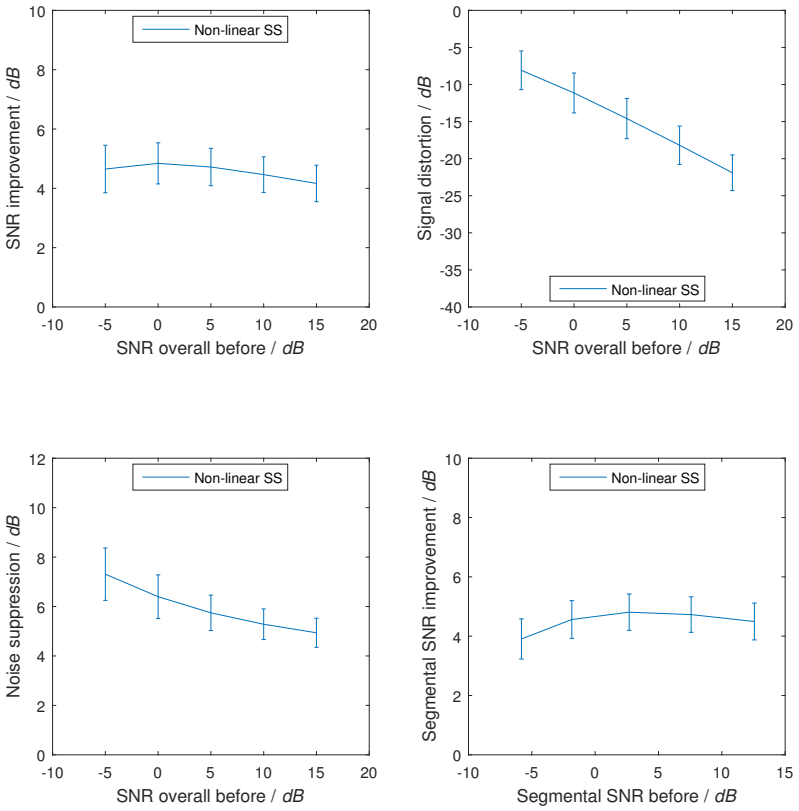

Figure 4.17: Mean and confidence interval for non-linear spectral subtraction with car alarm with all noises.

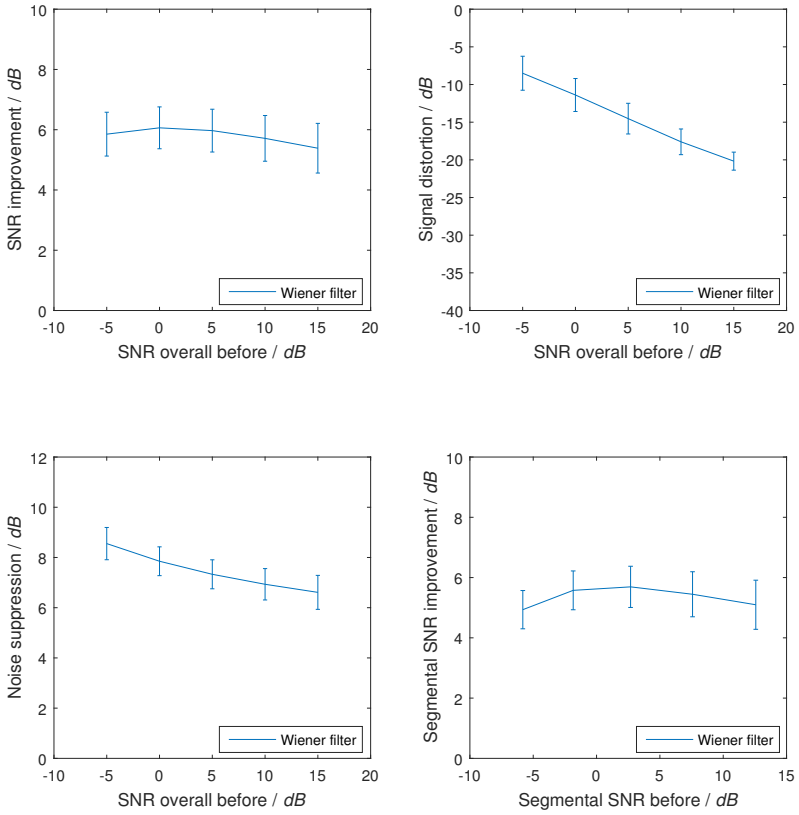

Figure 4.18: Mean and confidence interval for Wiener filter with car alarm with all noises.

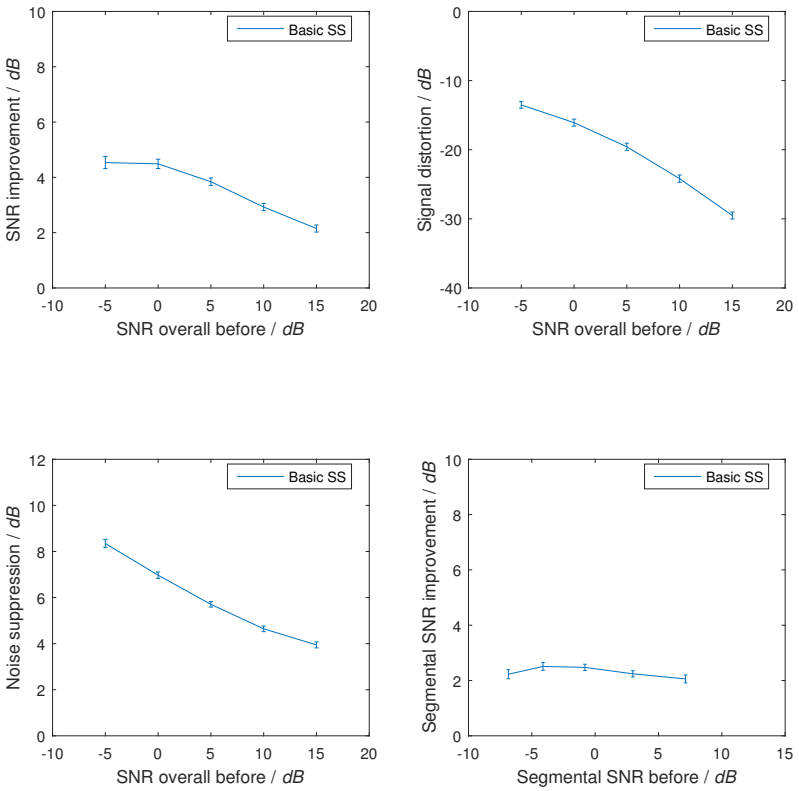

Figure 4.19: Mean and confidence interval for basic spectral subtraction with all noises and clean sounds.

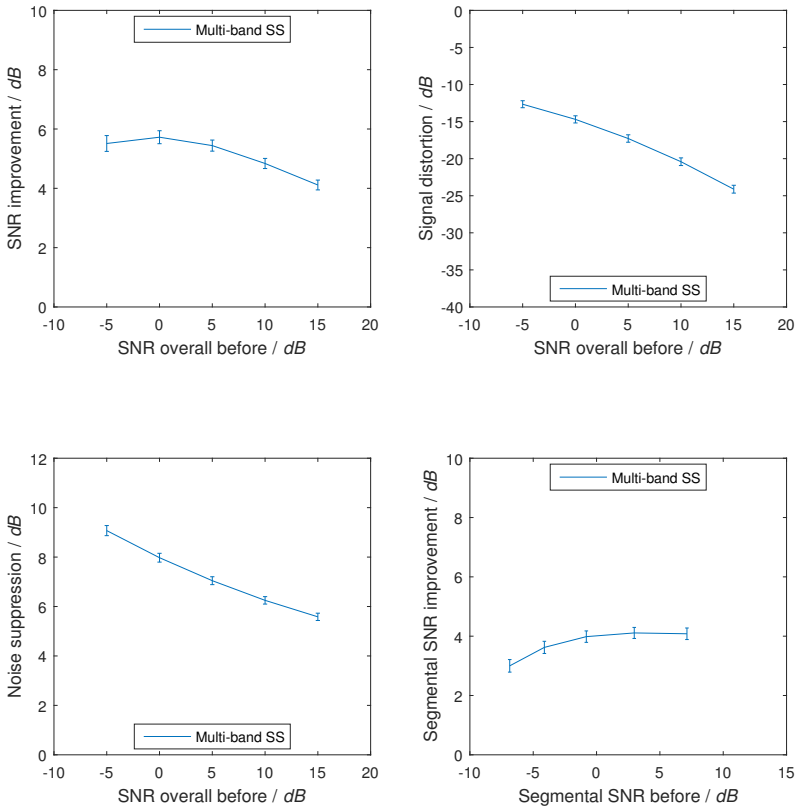

Figure 4.20: Mean and confidence interval for multi-band spectral subtraction with all noises and clean sounds.

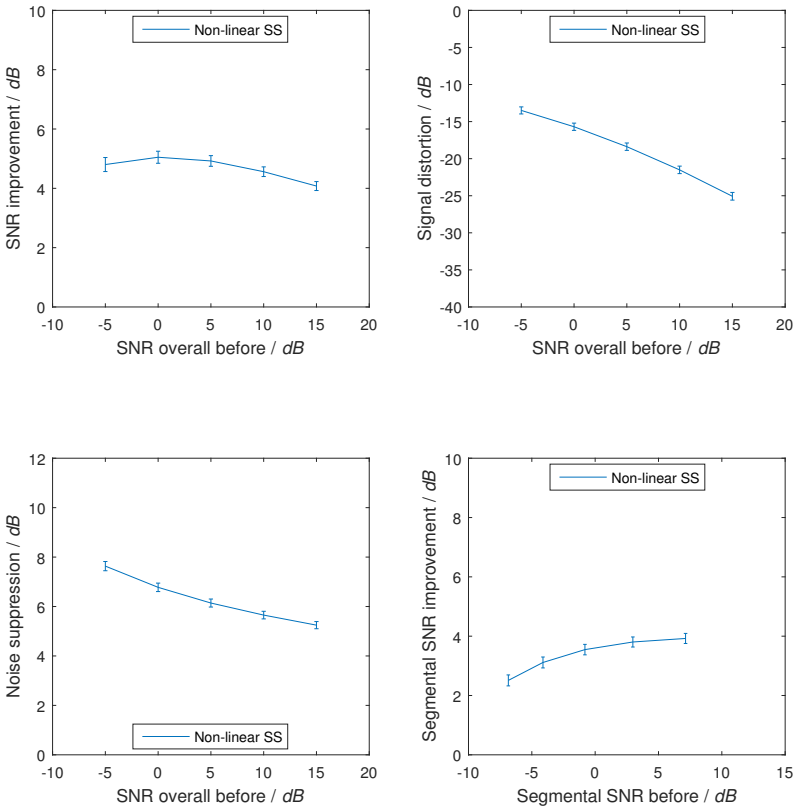

Figure 4.21: Mean and confidence interval for non-linear spectral subtraction with all noises and clean sounds.

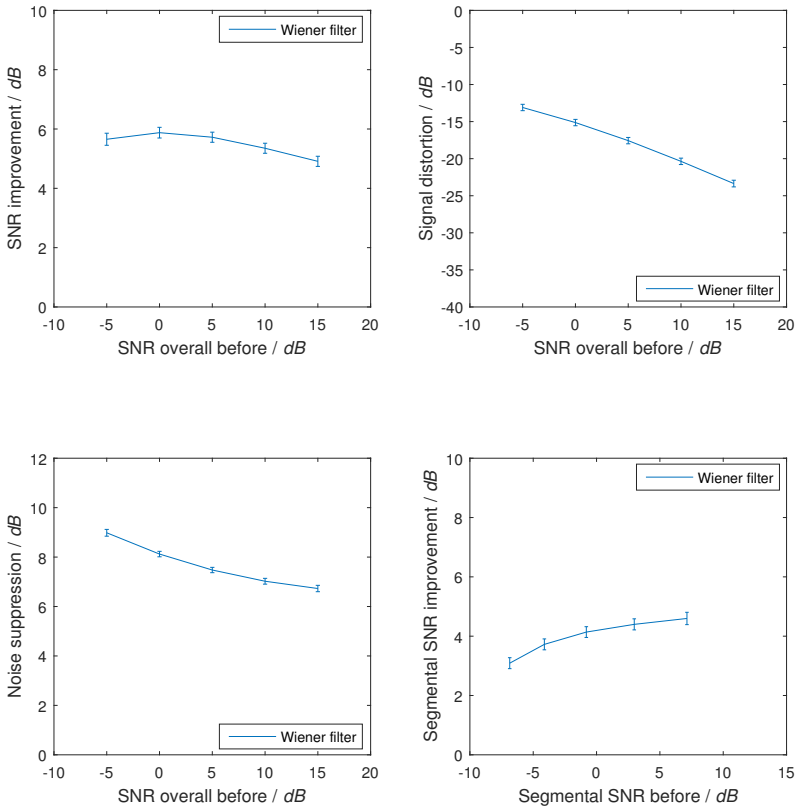

Figure 4.22: Mean and confidence interval for Wiener filter with all noises and clean sounds.

SNR value (dB)		-5	0	5	10	15
Signal Distortion (dB)	Basic	-13.52	-16.08	-19.58	-24.18	-29.52
	MB	-12.66	-14.7	-17.28	-20.41	-24.12
	NL	-13.49	-15.69	-18.38	-21.51	-25.07
	Wiener	-13.1	-15.13	-17.57	-20.36	-23.36
SNR improvement (dB)	Basic	4.54	4.49	3.84	2.93	2.15
	MB	5.51	5.72	5.44	4.84	4.11
	NL	4.80	5.05	4.92	4.56	4.08
	Wiener	5.66	5.88	5.72	5.35	4.91
Noise suppression (dB)	Basic	8.35	6.97	5.71	4.64	3.95
	MB	9.07	7.98	7.05	6.25	5.58
	NL	7.63	6.78	6.14	5.65	5.25
	Wiener	8.98	8.12	7.48	7.02	6.73

Table 4.2: Mean values of the performance for four noise reduction algorithms tested with all noises and clean sounds.

4.3.2 Performance and memory usage

To be able to get an idea of the number of different operations and memory usage that were used for the noise reduction algorithms, multi-band spectral subtraction was used as an indication. Therefore the number of different operations and memory usage have been calculated for the multi-band spectral subtraction method. All four methods evaluated have similar performance and memory usage. Each iteration, one FFT and one IFFT is used, both with $O(n \log n)$ complexity. The rest of the operations performed each iteration such as addition and multiplication are linearly dependent on the blocksize, but the number of blocks per second is f_s/blocksize so the number of these operations performed per second is only dependent of the sample frequency. The number of operations per second with 48kHz sample frequency can be seen in Table 4.3. The memory usage in terms of variables stored is linearly dependent on the blocksize and calculated to be a factor 34 times the blocksize. The memory usage for a few different blocksizes can be seen in Table 4.4. As previously mentioned the delay for all methods is $3 \cdot$blocksize because of the WOLA. For a sample frequency of 48kHz and a blocksize of 256 which were the values that were used in the final implementations the delay is 768 samples or 16ms.

	Operations per second $[\times 10^3]$
Addition	960
Multiplication	2064
Division	384
Arctangens	192
Cosinus	192
Sinus	192
Absolute value	96
Memory move	1104

Table 4.3: Number of operations performed each second for the multi-band spectral subtraction method using 48kHz sample frequency.

Blocksize	**64**	**128**	**256**	**512**	**1024**
Variables stored	2176	4352	8704	17408	34816

Table 4.4: Required number of variables stored for the multi-band spectral subtraction for different blocksizes.

Chapter 5
Discussion and Conclusion

In this chapter the microphone, equalizer and noise reduction methods are discussed. A comparison of the noise reduction algorithms and which algorithm to finally use is presented. Also a short discussion of possible future work is included.

5.1 Microphone

Even though the electret microphone lacked some specifications it was still chosen as one of two microphones because of its high SNR value. It also had high sensitivity and a good frequency response. PSR and PSRR is 0 for the electret microphone since the output is taken from the same pin as the supply voltage and therefore there is no rejection of noise. There were lots of disturbances in the low frequencies so the noise was measured with 200Hz to 20kHz bandwidth instead of the full range. As seen in Table 3.1 and 4.1, the electret microphone performed better than the specified value in terms of SNR and according to specification in terms of sensitivity. The MEMS microphone's measured SNR came close to the specified, but the specfied value was A-weighted which affects the value. The sensitivity of the MEMS microphone was also according to specification. The frequency responses for the MEMS microphone and electret microphone can be seen in Figure 4.1 and 4.2 respectively. Based on the results the Invensense MEMS microphone was chosen as the better one for this project. The electret microphone had slightly better SNR but the sensitivity was about 3.5dB higher for the MEMS. AOP, THD and EIN couldn't be compared since the electret lacked in specifications but the values for the MEMS were good so it was safer to go with the one with specified values. The electret microphone had a flatter frequency response overall but the shape itself would be harder to flatten if analog filters would be used which was the case for another master thesis done in parallel with this.

5.2 Equalizer

The implemented equalizer were able to create a more flat frequency response. Where the frequency response for the camera without equalizer had a difference of about 30dB from low to high frequencies as seen in Figure 4.3, the new implementation using the equalizer only differs about $\pm$2dB, see Figure 4.4.

5.2.1 Filtering in time vs frequency domain

Filtering can be done in both time domain and frequency domain. For this master thesis all filtering was done in the frequency domain since the noise reduction algorithm already needed the signal to be in the frequency domain. Furthermore the computational complexity for filtering in the frequency domain is generally less according to 2.59 and 2.60 unless the filter length is really small.

5.2.2 Blocksize

Depending on the choice of the blocksize the equalizer will have different characteristics. In this thesis the blocksize was set to 256 but other values were also explored. Bigger blocksizes will have less frequencies in each frequency bin and therefore have a better resolution. This will especially be of good use at the lower frequencies where distortions are more audible compared to higher frequencies. But bigger blocksizes will also require more memory and create a longer delay.

5.2.3 Equalizer measurement

The measurements for the equalizer was done in a sound-proof chamber with a built in speaker. Even though the sound-proof chamber attenuates disturbing noise from the surroundings it was noticed that some sounds still leaked in from the outside. The calibration using the reference microphone helped compensate for this but the results would be even better if the measurements was done in a more sound protected environment.

5.3 Noise Reduction

When implementing a noise reduction algorithm it is a big advantage if it is known in which type of environment it should be used in and what sounds that are especially interesting for the user. In this thesis the algorithms are general and not customized for a specific purpose. The idea is to let the customer have some control over the algorithm. A GUI has been implemented in Matlab to showcase how the options could look like for an end customer. The GUI can be seen in the Appendix, see Figure A.11. The performance of the implemented algorithms varies with different environments and SNR levels and therefore the options have been given to the user to vary the amount of noise reduction and also to completely turn it off. In most conditions the noise reduction is considered beneficial but for some conditions and settings there is audible musical noise present that could be more disturbing than the actual noise. Using the GUI the amount of noise subtracted can be lowered to a point where the musical noise is no longer audible and allowing the user to always benefit from the implementation but to a varying degree. This is something that ideally would be automatic and thus allowing the algorithm to automatically reduce the amount of noise subtracted when the musical noise is too audible. However it is very difficult to measure the amount of musical noise currently present and therefore this option has instead been given to the user.

The implemented noise reduction was also found to work quite well for different environments and simple to adjust. As seen in the Result section the SNR improvement tends to increase from low to mid SNR and then decrease again for higher SNR. It might seem odd that the improvement decreases with higher SNR but the idea is that when the SNR is already high it is more important to preserve the signal with less distortion than to increase SNR even further. The SNR improvement would decrease over the whole range if not for the limitation of 15dB attenuation.

5.3.1 Choosing the noise subtraction algorithm

When comparing the different methods it's important to not only look at the measured performance, but also the listening experience. The musical noise that can occur doesn't correspond to large variations in the measured result but can still completely ruin the listening experience. For most noise-clean sound combinations, the basic spectral subtraction method suppresses the least noise and improves SNR the least, however it also has the least signal distortion. The reason is that it does not matter where the two signals are in relation to each other in the spectrum, the algorithm will perform less attenuation over the whole noise spectrum whenever a clean signal is active independent of the frequency content. The musical noise is least audible for this method but when the clean signal is active the background noise increases noticeable. When looking at the overall performance of the multi-band and non-linear spectral subtraction methods they seem to be quite similar. However the non-linear method has the advantage that it can remove a fair amount of noise even if the noise and clean signal have similar frequency content but at the risk of increasing the musical noise too much. Regarding the multi-band method if the clean signal and noise have similar frequency content they will probably also be in the same frequency band and therefore less noise will be removed. Since the frequency bins are grouped together in bands, the gain function varies less over time so the multi-band method generally results in less musical noise. Wiener filter is generally the method with most noise suppression and SNR improvement but also the one with most musical noise.

Figure 4.19 to 4.22 shows the mean and confidence interval of the algorithms' performances with all noise sounds used together with all clean sounds, a summary of their mean performance can be seen in Table 4.2. From the figures it is seen that the confidence intervals are relatively small for all methods which means that if new sounds would be tested the performance would be predictable. This is a good result which suggests robustness and adaptability for the different noise reduction methods.

After comparing the different methods it was time to choose which method to use. It was decided that multi-band worked best in most cases and therefore it was chosen as the final noise removal algorithm. However all the methods were pretty similar and some methods also performed better compared to multi-band under some conditions. But combining the measured results and the listening experience multi-band had the best overall performance.

5.3.2 Multi-band spectral subtraction parameters

The parameters that were tuned to optimize the multi-band algorithm were subtraction factor, max attenuation and the number of frequency bands and their placement. The subtraction factor was set to vary between zero and one depending on the SNR. It was noticed that too high subtraction factor would cause a lot of signal distortion and therefore it was limited to never go above one. During high SNR values there was no need for any noise subtraction and thus it was set to zero. Max attenuation was set to 15dB which allowed the algorithm to remove a fair amount of noise. It was noticed that with higher values the signal distortion and musical noise would be too noticeable. The number of frequency bands and their placements were harder to decide. Too few bands and the results were very similar to basic spectral subtraction, and too many and the results were very similar to non-linear spectral subtraction. Finally six bands were decided to give the best results and their placement can be seen in Table 3.2.

5.4 Future Work

In this section a short summary of possible future work is presented.

5.4.1 Equalizer

The calibration equalizer is created to calibrate the camera and its microphone in a sound proof chamber. However when the product is mounted on e.g. a wall the new surroundings affects the microphone. Therefore it may be desirable to do a new calibration when the product is mounted.

5.4.2 Noise Reduction

There have been a lot of research in noise reduction methods for the last decades. A noise reduction algorithm is never finished and can always be improved and tuned, e.g. big cell phone companies have spent many years developing noise reduction algorithms which are still today being improved. This master thesis presents a working noise reduction algorithm but it can of course always be improved and tuned for different products, their characteristics and for different environments.

Furthermore this work only explores the possibilities when using one microphone. In products that uses several microphones beamforming can be implemented which creates the possibility to listen in certain directions. This opens up many new opportunities and can in combination with noise reduction get even better results.

5.4.3 Implementation in camera

The next step, given more time would be to implement the algorithms on the camera. Either on a DSP or directly on the camera as a GStreamer plug-in for Linux [16]. The camera implementation would have to be translated from Matlab to C code.

References

[1] Invensense, *Microphone Specifications Explained*, URL `https://www.invensense.com/wp-content/uploads/2015/02/MICROPHONE-SPECIFICATIONS-EXPLAINED2.pdf` 2013

[2] URL `http://www.spectraplus.com/` 2016-05-03

[3] URL `http://www.ap.com/` 2016-05-03

[4] URL `http://www.puiaudio.com/product-detail.aspx?categoryId=4&partnumber=POM-3535L-3-R` 2016-04-27

[5] URL `http://www.invensense.com/products/analog/ics-40720/` 2016-04-27

[6] B. Sällberg, *Digital Signal Processors ET1304*, Department of Electrical Engineering, Blekinge Institute of Technology, 2010

[7] M. H. Moattar and M. M. Homayounpour, *A Simple but Efficient Real-time Voice Activity Detection Algorithm*, Laboratory for Intelligent Sound and Speech Processing (LISSP), Computer Engineering and Information Technology Dept., Amirkabir University of Technology, Tehran, Iran 2009

[8] P. C. Loizou, *Speech Enhancement - Theory and Practice*, CRC Press, ISBN 9781466504219, Second Edition, 2013

[9] Nedelko Grbic, *Optimal and Adaptive Subband Beamforming*, Department of Telecommunications and Signal Processing, Belkinge Institute of Technology, 2001

[10] M. Berouti, R. Schwartz and J. Makhoul, *Enhancement of speech corrupted by acoustic noise*, Proc. IEEE Int. Conf. on Acoust., Speech, Signal Procs., pp. 208-211, 1979

[11] P. Lockwood, J. Boudy and M. Blanchet, *Non-linear spectral subtraction (NSS) and hidden Markov models for robust speech recognition in car noise environments*, Acoustics, Speech, and Signal Processing, 1992. ICASSP-92., 1992 IEEE International Conference on (Volume:1)

[12] S. Kamath and P. Loizou, *A multi-band spectral subtraction method for enhancing speech corrupted by colored noise*, Acoustics, Speech, and Signal Processing (ICASSP), 2002 IEEE International Conference on (Volume:4)

[13] Harald Gustafsson, Sven Erik Nordholm and Ingvar Claesson, *Spectral Subtraction Using Reduced Delay Convolution and Adaptive Averaging*, IEEE Transactions on Speech and Audio Processing, VOL. 9, NO. 8, pp. 799-807, 2001

[14] Thomas Esch and Peter Vary, *Efficient Musical Noise Suppression for Speech Enhancement Systems*, Institute of Communication Systems and Data Processing, RWTH Aachen University, Germany, 2009

[15] Samantha R. Summerson, *Filtering in Frequency Domain - Better or worse?*, 2009

[16] URL `https://gstreamer.freedesktop.org/` 2016-05-23

Appendix A
Appendix

A.1 Test files

Here the clean sounds and noise sounds are presented.

A.1.1 Clean sounds

The clean sounds used were:

1. Speech 1 - Glue the sheet to the dark blue background
2. Speech 2 - It's easy to tell the depth of a well
3. Speech 3 - Her purse was full of useless trash
4. Car alarm
5. Car Passing
6. Car door open and shut
7. Broken plates
8. Gun shots
9. Window breaking
10. Truck passing
11. Smashing a car window
12. Person running
13. Phone ringing
14. Male coughing
15. Knocks on wooden door
16. Jackhammer
17. Hammering
18. Explosion

19. Drilling
20. Door being opened and closed
21. Cheering
22. Yelling

The following Figures, A.1 to A.5 show a selection of the clean sounds in the time- and frequency domain. These clean sounds are used together with noise sounds in the Result section, see Figures 4.6 to 4.10.

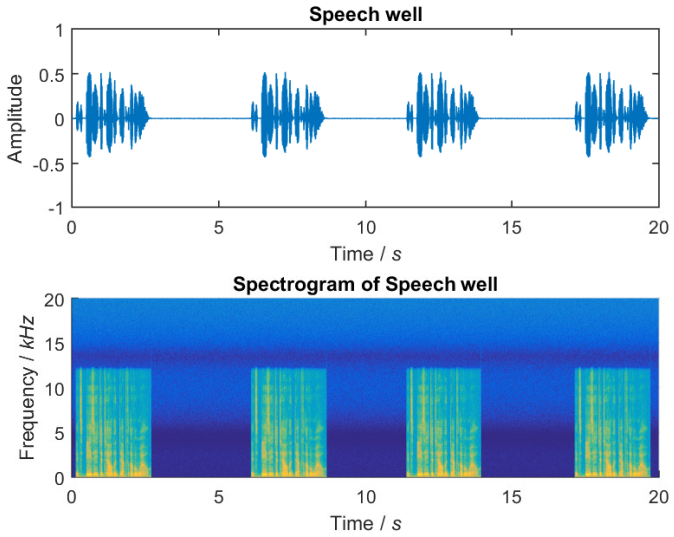

Figure A.1: Clean sound Speech 2 visualized in time- and frequency domain.

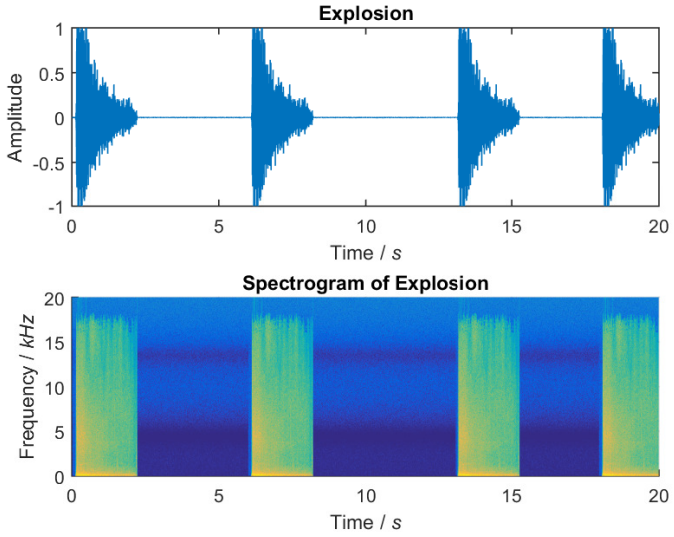

Figure A.2: Clean sound Explosion visualized in time- and frequency domain.

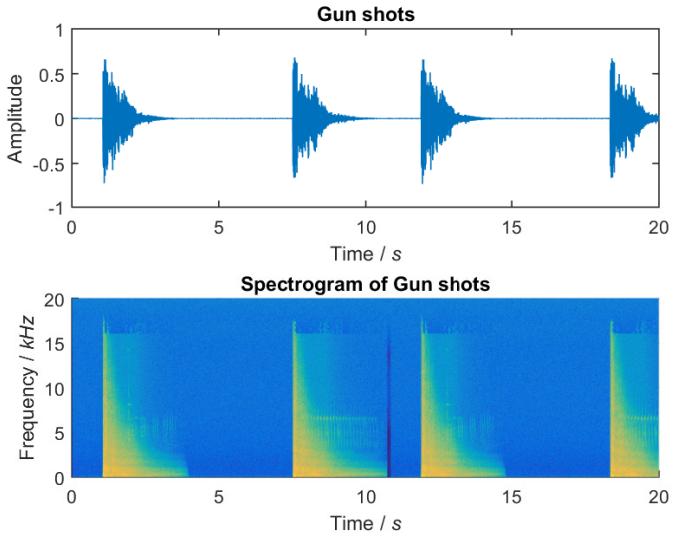

Figure A.3: Clean sound Gun shots visualized in time- and frequency domain.

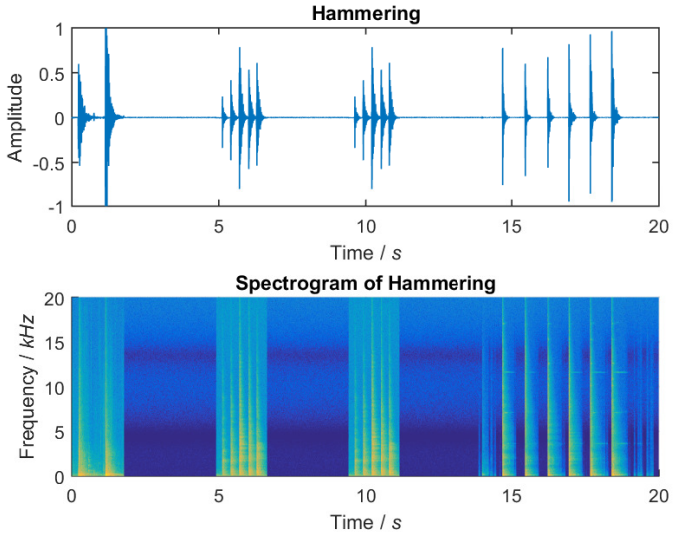

Figure A.4: Clean sound Hammering visualized in time- and frequency domain.

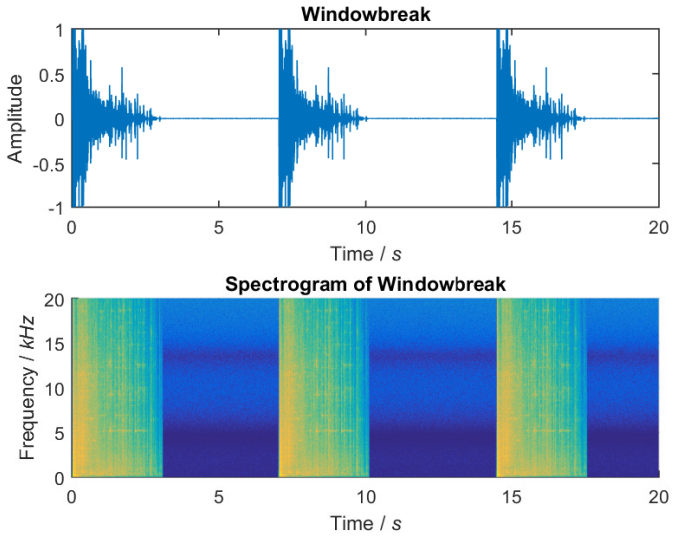

Figure A.5: Clean sound Windowbreak visualized in time- and frequency domain.

A.1.2 Noise sounds

The noise sounds used were:

1. PSU switching
2. Air conditioner
3. Cooling system
4. Car interior while driving
5. Electric fan
6. White noise
7. Highway
8. Industry fan
9. Airplane in flight
10. Sine tones
11. Vacuum
12. Vacuum 2
13. Water cooler

The following Figures, A.6 to A.10 show a selection of the noise sounds in the time- and frequency domain. These noise sounds are used together with clean sounds in the Result section, see Figures 4.6 to 4.10.

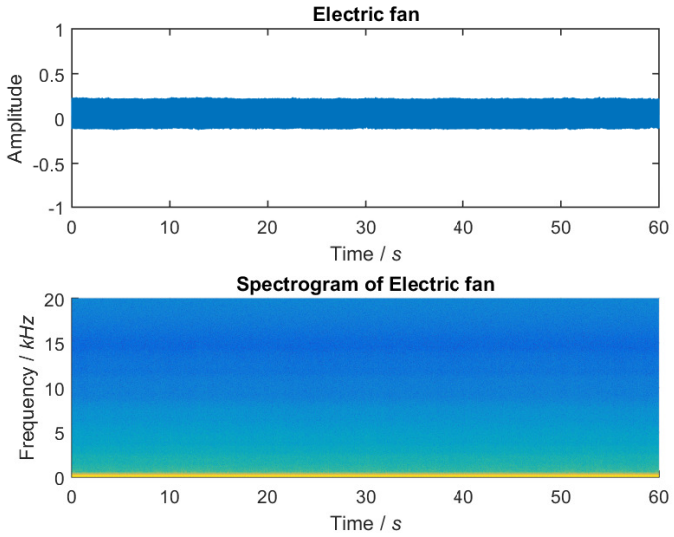

Figure A.6: Noise sound Electric fan visualized in time- and frequency domain.

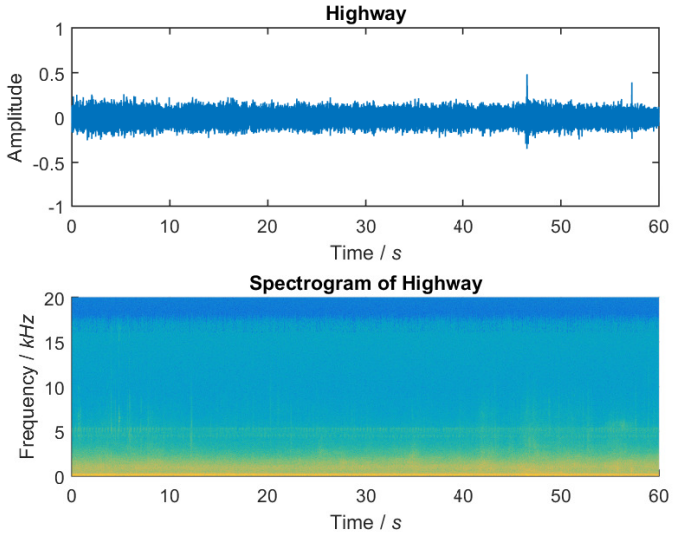

Figure A.7: Noise sound Highway visualized in time- and frequency domain.

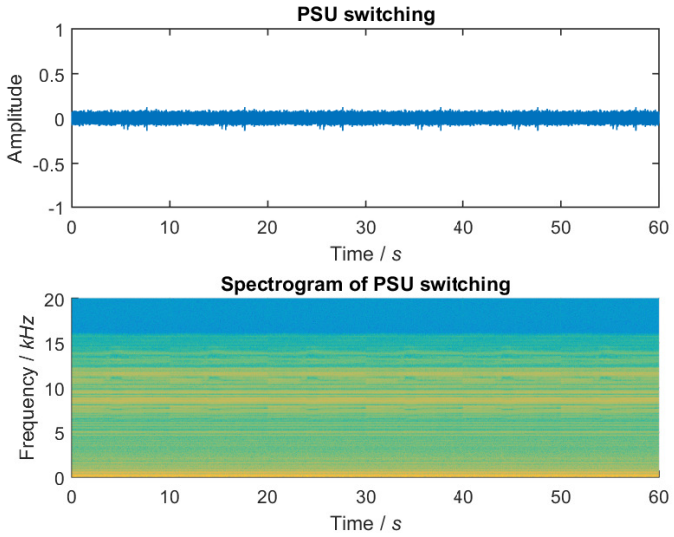

Figure A.8: Noise sound PSU switching visualized in time- and frequency domain.

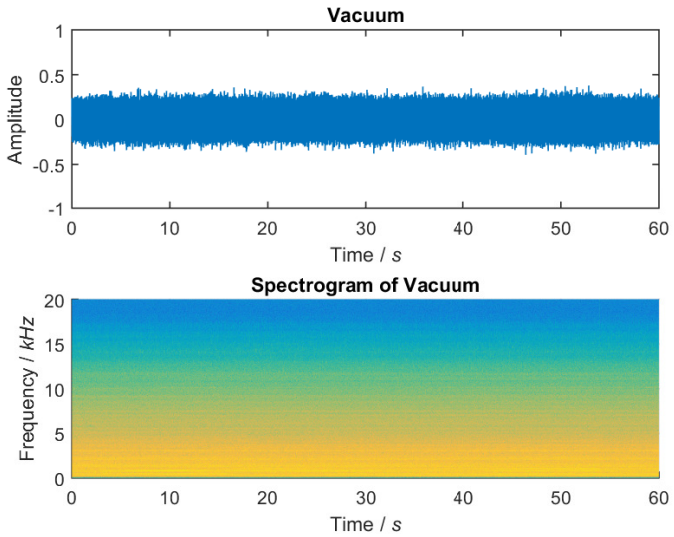

Figure A.9: Noise sound Vacuum visualized in time- and frequency domain.

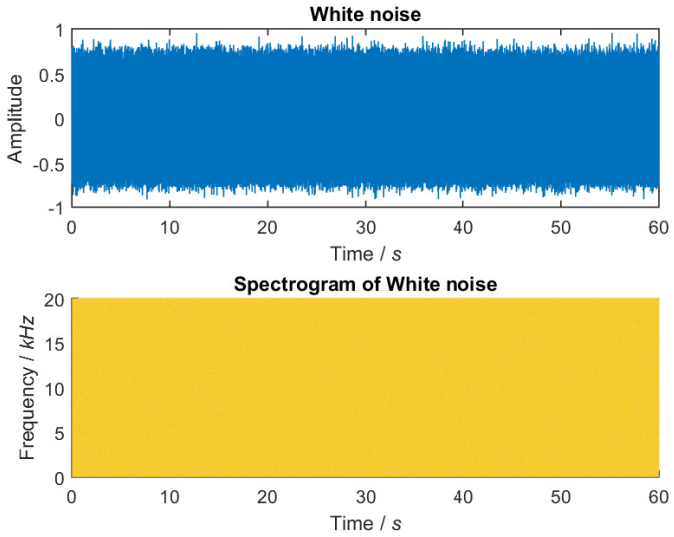

Figure A.10: Noise sound White noise visualized in time- and frequency domain.

A.2 GUI

Figure A.11 shows the GUI implemented in Matlab.

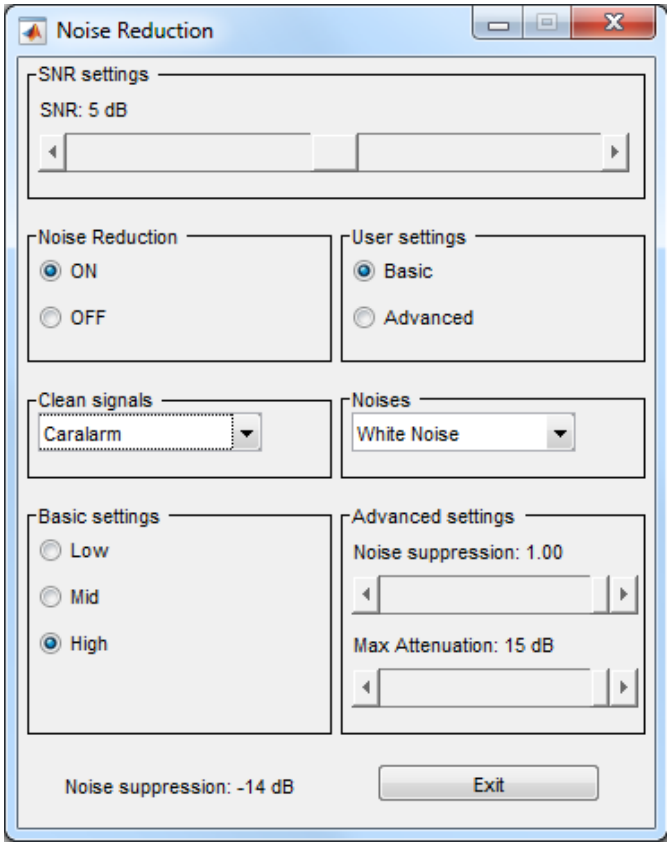

Figure A.11: GUI implementation in Matlab that allows the user to modify the amount of subtracted noise.

LUND
UNIVERSITY

Series of Master's theses
Department of Electrical and Information Technology
LU/LTH-EIT 2016-516

http://www.eit.lth.se