Embedded Sound Localization Using Multilateration in Network Camera Systems

Mazdak Farzone
dt07mf2@student.lth.se
Kim Smidje
dt07ks3@student.lth.se

Department of Electrical and Information Technology
Lund University

Advisor: Bengt Mandersson
bengt.mandersson@eit.lth.se

April 19, 2013

Printed in Sweden
E-huset, Lund, 2013

Abstract

This thesis deals with the analysis of different equipment and methods used in sound source localization combined with the use of Axis network cameras. The developed system was designed to work indoors as well as outdoors.

By using the time when the sound arrives to the microphones, also known as TDOA, together with a method called multilateration it is possible to determine the position where the sound source originated from. The calculated position is then transformed and used to direct a PTZ camera to capture the event visually.

The problems with sound localization using the TDOA technique and some of the TDOA alternatives that are accessible when implementing and developing such a system, are reviewed as well.

The final system can run the developed algorithm and position the camera within 2 seconds with an mean error of 0.6 meters, which was acceptable relative to the size of the enclosed area.

Acknowledgements

Our thanks go to Axis Communications AB that provided us with the necessary space, equipment and competence to complete this thesis. Magnus Jendbro and Willy Sagefalk, our advisers at Axis and our supervisor Bengt Mandersson at LTH were helpful and supported the choices we made.

We would also like to thank Henrik Fasth at Axis for providing us with information, along with helpful thoughts and ideas regarding the audio chip and everything concerning ALSA.

Table of Contents

Chapter 1

Introduction

1.1 Background

As the surveillance industry evolves, with increasing image quality and faster computations, more advanced features are demanded. To be able to know if something else occurs outside of the cameras view could be one of many desirable qualities.

Because of the nature of sound and its ability to travel around objects and in darkness, sound events can be easier to detect and track. But since sound tracking depends on other things than ordinary motion tracking, such as several receivers and very high accuracy on the time synchronization between the devices, not very much have been done in the field of sound tracking in smaller surveillance systems.

1.2 Problem description

The purpose of this thesis is to investigate if existing Axis Network Cameras are technically compatible to localize sound sources in a defined indoor or outdoor area. If so, then construct a real time application that is able to perform sound localization from events in the enclosed area. Since network cameras are used, the latency of the traffic over Ethernet must be taken into account.

The equipment will mainly consist of several cameras, with microphones, that act as clients. There will also be a server that connects to the clients and computes heavy calculations, such as the sound localization algorithm.

If the results would prove to be inadequately calculated by the system, the source problems should be identified to explain the performance loss. Also, what is required to be able to achieve an acceptable accuracy and performance on the calculations? In relation with this, discuss possible improvements of the system as well as further features.

1.3 Problem analysis

The actual sound localization puts restraints on the way the system can be designed. However some principles are still followed and in a greater perspective one would follow these operations, or states, to achieve real time positioning:

Figure 1.1: *The designed system flow*

As the state diagram in Figure 1.1 describes, the system would have an idle state which is called *Event triggering*. In this state, the information is collected and calculated iteratively. If the amplitude threshold is breached, the system passes the information on to the *Localization*-state. If not, the system returns back to first state and awaits the next event.

The Localization-state consists of several segments but is referred as a whole to the *localization algorithm* in this report. The algorithm consists of the actual sound positioning, the trimming of the signals to synchronize the time of each audio file with each other and the filtering to remove the background noise.

When a reasonable position is found the last state in this iteration is entered. The position coordinates are then converted to match the coordinate system used by the camera and set. This concludes the last state which restarts the system to the Event triggering state again. The process then iterates continuously until the process is cancelled.

1.4 Related work

There have been several papers done previously on sound localization in different areas of interest. The methods and hardware they use are applicable for their purposes, resulting in varying methods and hardware for each paper. Areas of interest have been ranging from military applications such as location of enemy artillery guns, to consumer application such as localization of speech in conference speakerphones.

What differentiates this thesis from previous works is the uniqueness of the entire implemented system and the use of an Ethernet network instead of having audio cables connected directly to the computer running the localization algorithm.

There has been a master thesis previously done about basic sound localization indoors, however only one camera and two microphones were used [3]. As stated in the thesis [3], the time synchronization between the devices would be an issue and very little work have been done regarding this problem.

1.5 Thesis outline

The states described in the Problem analysis-section is thoroughly examined in Chapter 2, evaluating different approaches available in each state. The implementation of the chosen methods are described in Chapter 3, giving a programming perspective of the problem. Coding issues as well as optimizations will be also be reviewed and explained in this chapter. The final results are then presented in Chapter 4 with further analysis on parameters such as accuracy and execution speed. The conclusion of this thesis is presented in Chapter 5 together with an error analysis of the entire system. This report concludes with a discussion and examines future work of the developed system in Chapter 6.

Chapter 2
Theory

The theory is mainly focused on the actual sound localization, as the implementation of the system is code-based and reviewed later on. Other ways of detecting and evaluating sound is also examined and reviewed. However these will be treated lightly and are described mainly to get an assessment of the options available.

2.1 Event triggering

Events can be everything from trespassing to tampering with the system. In this thesis these events are defined as loud or irregular noises in the enclosed area. There are mainly two different ways to detect events:

- *Raw data triggering:*
  By breaching a selected threshold value in the amplitude measurements, an event can be identified. This would be the fastest and also the easiest way to implement, as one single read will beat any recognition algorithm or complex analysis. However it will be more susceptible to background noise and wind gusts.
- *Algorithm-based triggering:*
  By examining chunks of audio samples and applying a matching or noise reducing algorithm, one could get more accurate results at the cost of computing power. This is a more complex way of triggering an event. This would however have a direct impact on the speed of the Event triggering-state.

When an event has been identified and triggered, the system can handle the situation in two ways:

- The server ignores the fact that other clients may hold vital information and calculates only with the data received from the clients that have actually been triggered.
- The server is aware of how many clients are connected and requests the information from the remaining clients.

Both methods have advantages and disadvantages compared to each other but the first method will be the fastest way of achieving a position. Since sound travels relatively fast as the analyzed area is relatively small, the server will receive

information from the triggered clients theoretically at the same time. Because there is no need to wait and send a request to the untriggered clients, this method will be quicker. Still less information may be collected, since it is not hard to imagine a client not receiving any or very little information concerning an event, resulting in an inaccurate answer. Even though the other method receives more information, the accuracy may not improve the results. It can on the contrary deteriorate the answer.

2.2 Localization

There are three different methods to calculate the position of the sound, namely: multiangulation, multilateration and trilateration. Each approach has been used in various systems in the world and have been proven to work in practice. Some examples are GPS that uses trilateration, LORAN (***LO**ng **RA**nge **N**avigation*) that uses multilateration and simple surveying that utilizes triangulation [6][14][15].

All of these techniques are based on the same principles, that the sound will be perceived in different times depending on the position of the sound source, also known as *Time Difference Of Arrival* or *TDOA*.

2.2.1 Cross-correlation

One way to measure TDOA is to simply extract the time stamps at an amplitude peak and then compute the time difference between them. However this becomes wildly inaccurate as sound waves are perceived differently in each microphone. Using cross-correlation will put harder restrains on the hardware clock synchronization between the clients. Another way to derive the TDOA is to use cross-correlation:

$$\phi_{fg}(m) \stackrel{\text{def}}{=} \sum_{n=-\infty}^{\infty} f(n) \cdot g^*(n+m) \tag{2.1}$$

where $f(n)$ and $g(n+m)$ are two similar waveforms with a time lag m and g^* denotes the complex conjugate of g. However as recorded sounds always are real signals the complex conjugate g^* is equal to g. The equation above can be visualized as the signal g sliding on the signal f and calculating the sum of their product at each point. The result is a function were the highest peak represents were the signal match the most, thus will TDOA be located at the highest peak from the resulting function:

$$\Delta t_{fg} = \underset{m}{argmax}(\phi_{fg}(m)) \tag{2.2}$$

When transforming a signal from the time domain into the frequency domain, using the *Fourier Transform*, the numerical approximation of the signal is taken [18]. By sampling the signal one thousand times each second, depending on the frequencies of the source, the signal will maintain its shape. The sample frequency must be twice the rate of the highest frequency component in the signal, in accordance

with the *Nyquist sampling theorem* [25]. The *Discrete Fourier Transform* (DFT) is, as the name indicates, based on the Fourier transform. This transformation is defined as follows:

$$X(\omega) \stackrel{\text{def}}{=} \int_{-\infty}^{\infty} x(t)e^{-i\omega t}dt, \quad \omega \, \epsilon \, (-\infty, \infty) \tag{2.3}$$

Equation 2.3 has the numerical meaning that there will be an infinite numbers of points when calculating the sum. Since the sampled signal has a finite numbers of points, the integral can be rewritten as [18]:

$$X(\omega_k) \stackrel{\text{def}}{=} \sum_{n=0}^{N-1} x(t_n) \cdot e^{-i\omega_k t_n}dt, \quad k = 0, 1, 2, \ldots, N-1 \tag{2.4}$$

The advantage of transforming into the frequency domain is that calculation will be less demanding and filters can easily be applied on the collected data [18].

Afterwards the result can be transformed back to the time domain by taking the inverse of the DFT:

$$x(t_n) = \frac{1}{N} \sum_{k=0}^{N-1} X(\omega_k)e^{-i\omega_k t_n}dt, \quad n = 0, 1, 2, \ldots, N-1 \tag{2.5}$$

and the $x(t_n)$ is now the filtered version of the original signal.

Based on the DFT algorithm, the *Fast Fourier transform* (FFT) is an algorithm with the purpose of enhancing the calculation speed. The first and most common version of FFT is the *Cooley-Tukey-algorithm*. Simply explained the N samples is divided into two $N/2$ blocks and thereafter several butterfly operations is performed. This is repeated until the blocks only consist of a signal sample. This can lead to the complexity $O(2N^2)$, as it ordinary takes for N number of samples, can be reduced down to the complexity $O(2N \cdot log_2(N))$ but this is only valid as long as N is a power of two [19][21].

Generalized cross-correlation - Phase transform

In order to determine the real peak in the cross-correlation spectrum, a noise free environment is desirable. When dealing with real life situations, as previously stated, there will ambient noise and reverberation present. This noise can give false peaks in the cross-correlation and thus taint the TDOA estimations of the signals. To be able to circumvent this problem, a common alternative called *GCC-PHAT* is used [2][5][8]. It stands for *Generalized Cross Correlation with Phase Transform* and is defined as:

$$G_{PHAT}(n) \stackrel{\text{def}}{=} \frac{X_1(n) \cdot X_2(n)^*}{|X_1(n) \cdot X_2(n)^*| + \varepsilon} \tag{2.6}$$

where X_1 and X_2 are two input signals, which have been Fourier transformed into the frequency domain. The transformed signals are cross-correlated as described in Equation 2.1.

The popular PHAT-weighting scheme in GCC is used to normalize frequency magnitudes, preserving phase information which is of course used in TDOA calculation. It should be noted that the variable ε is a very small constant used to avoid division by zero if X_1 or X_2 would somehow become equal to zero. Then by taking the inverse Fourier transform:

$$g_{PHAT}(t) = \mathcal{F}^{-1}[G_{PHAT}(n)] \tag{2.7}$$

theoretically, the behavior of the whitened correlated function will be more like an unit impulse function compared to the ordinary cross-correlation function [8]. Like described with the ordinary cross-correlation, the highest peak of $G_{PHAT}(t)$ corresponds to the TDOA [7].

$$\Delta t_{fg} = \underset{t}{argmax}(g_{PHAT}(t)) \tag{2.8}$$

2.2.2 Multiangulation

This technique utilizes the calculated TDOA to find the position of the sound source. In order to calculate one position, at least four microphones are required.

It is assumed that the sound can be treated as a plane wave due to the large relative distance to the sound source and the small distance between the individual microphones in each pair. The TDOA is depicted as Δt and the speed of sound is v_{sound}. Hence the angles can be calculated as follows:

$$d \cdot cos(\psi) = \Delta t \cdot v_{sound} \Rightarrow \psi = arccos(\frac{\Delta t \cdot v_{sound}}{d}) \tag{2.9}$$

where ψ gives a measure of angle from each microphone pair and d is the distance between the microphones, an example of this is illustrated in Figure 2.1.

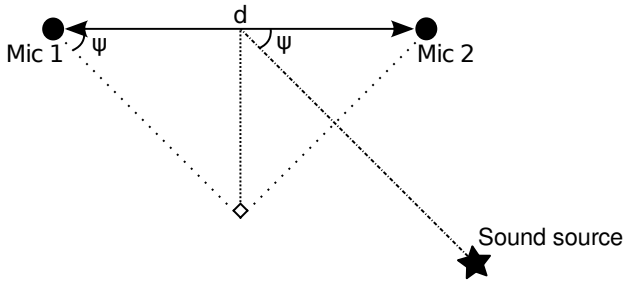


Figure 2.1: *Using the triangulation to calculate from the direction of the sound source.*

What remains is to compare the angles from two different microphone pairs to find an intersection point:

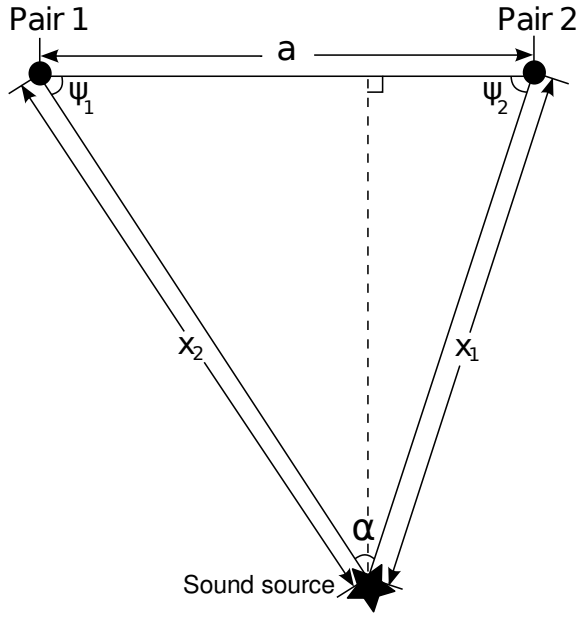


Figure 2.2: *Determining the position of the sound source by using multiangulation. Each pair consists of two microphones.*

The acquired angles are used to calculate the remaining angle α and the actual distance to the point of origin, by using the law of sines:

$$\frac{sin(180-(\psi_1+\psi_2))}{a} = \frac{sin(\psi_1)}{x_1} = \frac{sin(\psi_2)}{x_2} = \frac{sin(\alpha)}{a} \tag{2.10}$$

When the distance vectors x_1 and x_2 are known, the position becomes simple to calculate. If several microphone pairs are used, they are compared so that each comparison is unique. The evaluated points are then weighted together and depending on the size of the area enclosing all points, the accuracy is determined. Finally these points are then evaluated with a weighing algorithm to decide upon one final point that will be the estimated point of the sound source.

One drawback with this method is that for each client included into the system, it will require two additional microphones to be able to depict the angle from which the sound came from. So if a system has n cameras it will require $2n$ microphones in order to allow each camera to determine from which direction the sound originated from.

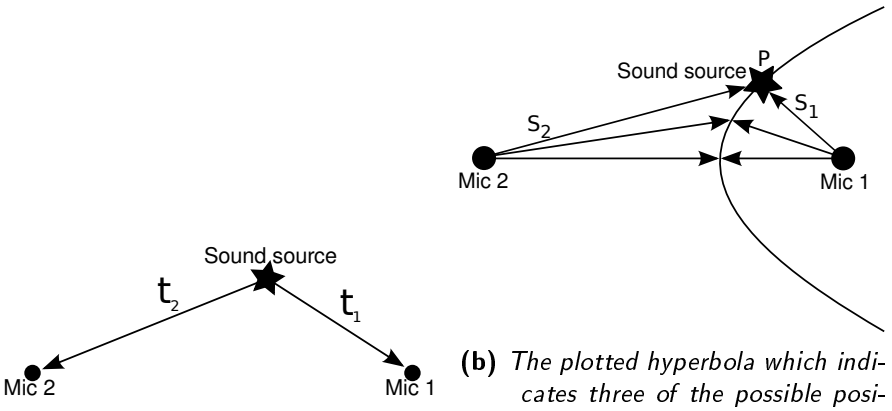
2.2.3 Multilateration

Another technique is to use the relative distance between the recorded sound from each and every one of the microphones. Multilateration is based on the principle

that the TDOA shapes a hyperbola of location points around a focus point, which in this thesis are the microphones. Consider a simple example with only two microphones: The TDOA between the microphones, Δt, can then be derived from $t_2 - t_1$, were t_i is the time index when microphone i perceives the sound. The the difference can in distance between microphone 1 and 2, can be calculated as:

$$s_{relative} = v_{sound} \cdot (t_2 - t_1) = s_2 - s_1 \tag{2.11}$$

where $s_{relative}$ is the travelled distance relative to microphone 1. With the relative distance and the microphone position coordinates known, a hyperbola can be constructed, based on the definition of the hyperbola [17]. This is illustrated in Figure 2.3(b). If the $s_{relative}$ is ≈ 0, the resulting curve will more or less be a straight line, in the middle of the microphones. A point can not be found with a



(a) *The times t_1 and t_2 when the microphone receives the sound.*

(b) *The plotted hyperbola which indicates three of the possible positions of the sound source at point P.*

Figure 2.3: *Positions of the sound source.*

single hyperbola equation, as can be seen in Equation 2.12 and Figure 2.3(b).

$$S_{12} = |S_1 - S_2| = |\sqrt{(x_1 - x_P)^2 + (y_1 - y_P)^2} - \sqrt{(x_2 - x_P)^2 + (y_2 - y_P)^2}| \tag{2.12}$$

There are two unknown variables and only one equation. If an additional receiver is included into the system, there will be two new equations:

$$S_{13} = |S_1 - S_3| = |\sqrt{(x_1 - x_P)^2 + (y_1 - y_P)^2} - \sqrt{(x_3 - x_P)^2 + (y_3 - y_P)^2}|$$

and

$$S_{23} = |S_2 - S_3| = |\sqrt{(x_2 - x_P)^2 + (y_2 - y_P)^2} - \sqrt{(x_3 - x_P)^2 + (y_3 - y_P)^2}|$$

Mathematically, this will solve for a single point which will be the sound source location in a noise-free environment. As the area of interest is noisy and reverberant, the hyperbolas will intersect at several different points.

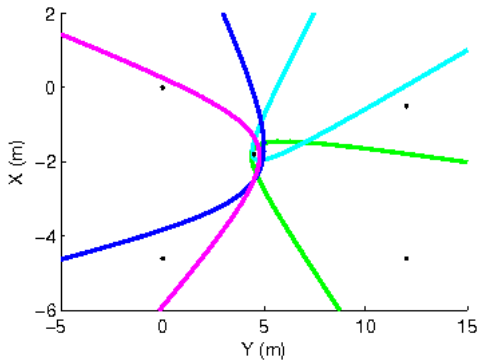


Figure 2.4: *The hyperbolas marked in different colors and microphone positions marked with black dots.*

A possible improvement is to add more microphones to increase the number of hyperbolas and possible intersections. The number of hyperbolic functions K with n microphones can be described with the following relation:

$$\binom{K}{2} = \binom{\binom{n}{2}}{2}, \quad n \in \mathbb{N}^* \tag{2.13}$$

A quick observation yields that the system will be *overdetermined* as $n \geq 3$ which is for every case since at least three microphones are needed for at least another TDOA equation. The equation system is thereby solved with one of the three different approaches available. The *Non-linear Least squares*-algorithm was the only option assessed. Other options include the *Extended Kalman filter* and the *Particle Filter* [7].

2.2.4 Non-linear least squares

In a system of equations the number of unique unknown variables gives a degree of freedom. If there exists an exact number of equations to solve for the degree of freedom, the system can be solved. When the number of equations increases, the system becomes *overconstrained*, also known as *overdetermined*. Due to the non-linearity of the equation system, a global minimum is sought amongst multiple minimum points. The numerical solution provided by Levenberg-Marquardt, also known as the *LM-algorithm*, is an optimal candidate.

The LM-algorithm used in data-fitting applications, is an iterative algorithm that combines the best of two major methods. The *Gauss-Newton algorithm* and the *Gradient descent* method. Like any non-linear least squares algorithm the LM requires a initial guess P_0. If P_0 is far from the local minimum the LM utilizes the Gradient descent method which is slow but guaranteed to converge. As the LM

iterates it either continues using Gradient descent or switches to Gauss-Newton if the current solution is close to exhibit fast convergence [16].

By using the LM-algorithm, a solution is found even if P_0 is poorly chosen, thus making it a bit slow but also more robust compared to only using the Gauss-Newton method [9][20].

2.2.5 Time synchronization

The clients need to be synchronized if the correlation calculation of the audio samples are to be correctly performed. Since there are multiple clients, each running on a separate processor, the issue of synchronization becomes relevant to the system. A difference of n milliseconds yields:

$$A(n) = \pm 0.343 \cdot n \tag{2.14}$$

An iterative process is created to compensate for the clock drift iteratively in each processor.

The clock drift is a phenomena where the internal clock does not run at the correct speed due to temperature changes, quality of the clock, surrounding magnetic fields and air pressure. It could be assumed that every clock will drift eventually and that it is ironically only a matter of time. To synchronize the clients two options were available:

- *Protocols*: There are protocols that are designed to reduce clock drift and increase precision between clients.
- *Time broadcast*: Because the issue is to reduce relative time difference between server and clients, the clients can inherit the server time.

2.3 Positioning

As the positioning-camera works in a polar coordinate system and the evaluated position, P_{sound}, is in Cartesian coordinates, a conversion is necessary. The origin in the polar coordinate system is set to the camera's origin P_{camera}.

$$P_{relative} = P_{sound} - P_{camera}$$

thus the pan of the camera becomes a conversion from $P_{relative}$'s coordinates $[x, y]$ to the polar coordinates $[r, \alpha]$:

$$r = \sqrt{(x^2 + y^2)}$$
$$\alpha = tan^{-1}(\frac{y}{x})$$

The derived values are then adjusted depending on which quadrant of the coordinate system they are situated in. After the pan is set, the tilt is calculated using a fixed average height, h_{human}, of the standard object in observation:

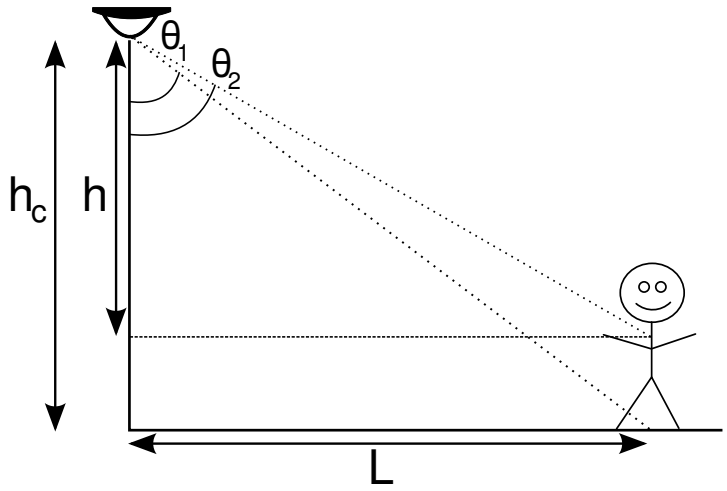


Figure 2.5: *The PTZ camera pans and focuses itself on a elevated plane. h_c is the height of the camera, h is the distance to the elevated plane, θ_1 is the angle to the given position and θ_2 is the angle for the new position.*

As Figure 2.5 shows, the angle needs to be adjusted to focus on the object, θ_2, and not the derived point from the Localization-state, θ_1. The camera position height h_c is known, hence θ_2 is calculated as:

$$h = h_c - \frac{h_{human}}{2}$$
$$\theta_2 = tan^{-1}(\frac{L}{h_c}) = tan^{-1}(\frac{y}{h_c})$$

The zoom function was not evaluated in this thesis, thus the final position in PTZ space, [*Pan, Tilt, Zoom*] becomes: $[\alpha, \theta_2, 0]$.

2.4 Noise sources

This real-time system will evidently be affected by different types of noise that can corrupt the calculations, thus giving a bad result to the user. The noise can be divided into two major categories:

- *Environmental noise and errors:*
  - *Reverberation* is one of the major issues when positioning indoors due to the waveform's sensitivity to sound reflections from *e.g.* ground or walls taking other paths to the microphone. The resulting waveforms will be difficult to correlate as they are not only phase shifted but different to each other as well.
  - *Temperature* will change sound propagation, resulting in less accurate calculations especially if there are variations in temperature between the clients. A temperature difference of 50^oC will result in $\approx 8.5\%$ difference in sound velocity.

    Refractive effects can be observed when temperature varies between the ground and in the air, shown in Figure 2.6. A normal sound wave propagates upwards when temperature is decreasing with altitude. The situation turns during sunset or night when the ground is hot and the air is colder. This causes the sound to refract down towards the ground [10].
  - *Wind* can change sound propagation and speed. Here the refractive effects from temperature difference occurs again. The sound refracts down when the wind travels with the same heading, also known as downwind. The microphone will receive more power thus produce the illusion that the sound is louder. The reversed effect, when the sound is refracted up occurs when the wind blows in the opposite direction. The microphone will theoretically receive less or no power due to effect of the *shadow zone*, as seen in Figure 2.6 [10].
  - *Natural noise* are the events that are hard to classify and can occur at any time. These events can be *e.g.* wind blowing directly in the microphone or noise from distant sources. It is the most frequent and apparent noise occurring in outdoor situations and is the hardest noise to suppress or filter out.
- *Computational errors:*
  - *Round-off errors* occur when approximated variables are processed when the exact mathematical value is needed for further computations.
  - *Timing errors* will be introduced because of the hardship of synchronizing devices with each other. Harder demands of precision results directly in harder restraints on the clock drift *e.g.* a system that needs a position error of maximum 1 cm requires a synchronization with maximum drift of $29 \cdot 10^{-6}$ s.

– *Frequency selection* is important due to the precision requirements. With a sampling frequency of 44100 Hz, each sample will correspond to $22 \cdot 10^{-6}$ s of information which gives a precision of 0.77 cm. Compared to a sampling frequency of 8000 Hz that has $12.5 \cdot 10^{-5}$ s of information resulting in $\sim$ 5 cm of accuracy. Also when using low frequencies one must take the Nyquist sampling theorem into account. If band limited functions are used with no higher frequency than B hertz, they will be reconstructable with $2 \cdot B$ samples/s.

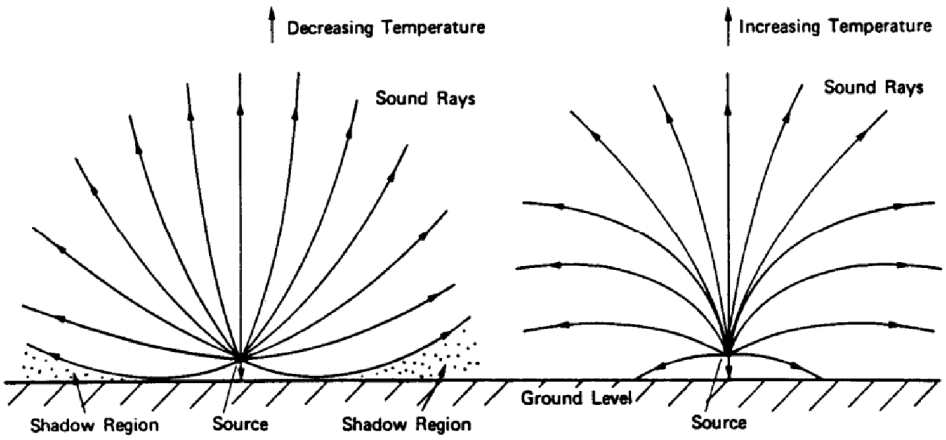


Figure 2.6: *Impact on sound rays with decreasing and increasing temperature with increasing elevation.*

Chapter 3
Implementation

This chapter describes details on how the project progressed, what steps were taken during the implementation of the system and why.

3.1 Microphone assessment

Two options were given during the microphone assessment stage of the implementation process. The internal microphones, to decrease cost and complexity of the system or an external alternative could be bought for precision and evidently increased system cost. Outdoor tests were conducted to get an idea of how well the internal microphone would apprehend the sound and how high the ambient noise level would be. The indoor requirements were considered covered if the microphone would be able to handle the noise polluted outdoor environment. The test that was conducted with a vehicle horn approximately 25 m from the internal and the external microphone.

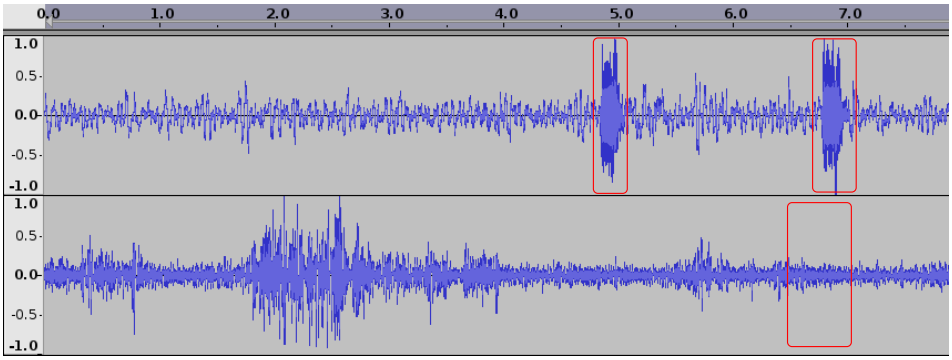
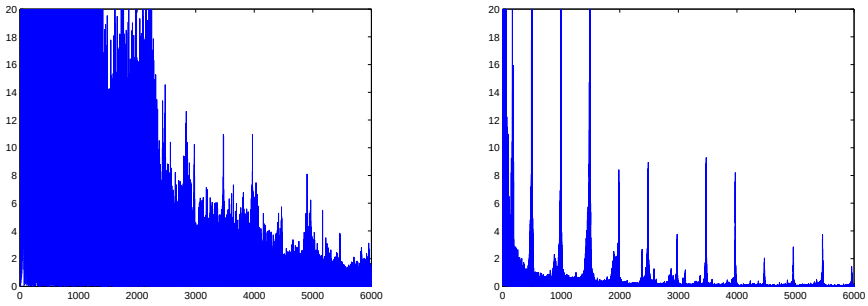


Figure 3.1: *Two different recordings done with the external microphone (top) and the internal microphone (bottom). The red frame marks the sound from the car horn.*

The amplitude of the noise proved to be so powerful when using the internal microphone, that it completely drowned the amplitude of the horn. The horn can not be distinguished in the recording done with the internal microphone by only

looking at the waveform. When the sound recording however is played, the horn can be heard around 7 s into the sound clip.

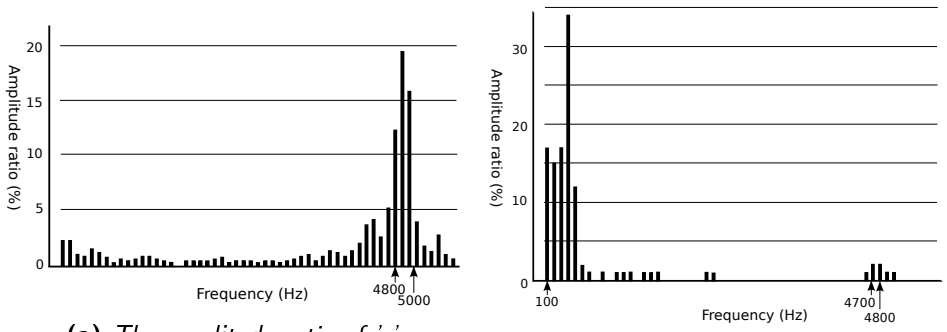
Another way to analyze the signal is to transform the signal and then note the frequency components. Optimally, these components will be clear spikes and visually distinguished on the plot shown in Figure 3.2. Figure 3.2(b) clearly dis-



(a) *Internal microphone* **(b)** *External microphone*

Figure 3.2: *Frequency spectrum between the different microphones*

plays the desired frequency spectrum of an audio capture as was expected with an expensive and outdoor-adapted microphone. Vehicle horns are not the only interesting audio events occurring in the analyzed area, as human made sounds must be analyzed as well. Human speech has frequency components that stretches between 100 Hz to around 5000 Hz. Consider the two consonants 'n' and 's' that have the following frequency spectrum with amplitude ratios:



(a) *The amplitude ratio of 's'* **(b)** *The amplitude ratio of 'n'*

Figure 3.3: *Difference between frequency components in human speech.*

Microphones are designed differently to create their own characteristic frequency and phase responses. For scientific purposes a more uniform response is required to create equal prerequisites for every measurement. Conditioning them unevenly would be more appropriate for musicians that are imitating certain

sounds or locations. The microphone characteristics are typically described with two plots:

- A *frequency response* diagram shows the sensitivity, most often over the common frequency range of $20 - 20000\ Hz$.
- The *polar pattern* or *directionality* describes the sensitivity when sound is arriving in different angles about its central axis.

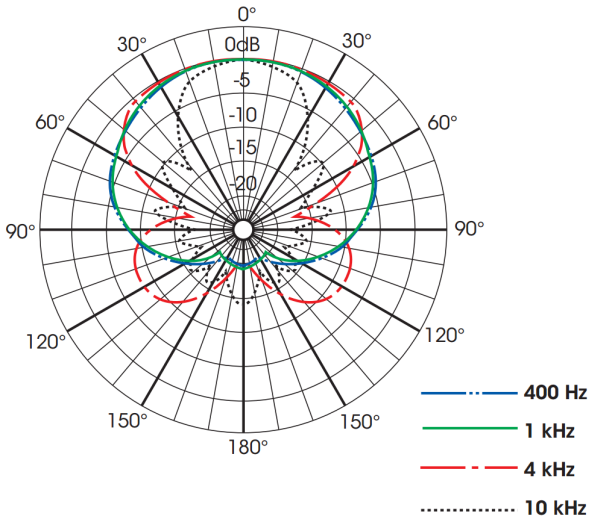


Figure 3.4: *Polar pattern of the external microphone*

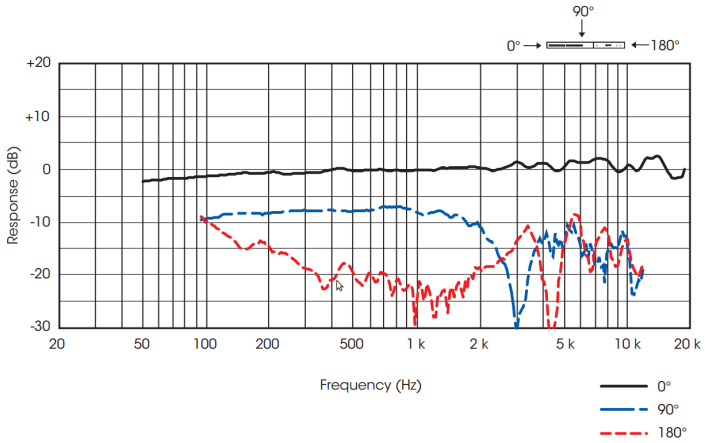


Figure 3.5: *Frequency response of the external microphone*

The clients did not have the internal microphone characteristics sheet available, but they are countersunk into the attachment plate of the camera. This gives an uni-directional polar pattern. The frequency response can be assumed to be more or less uniform as well.

The uni-directionality of both choices are preferable to give the analyzed area high sensitivity and in the surroundings lower sensitivity, but the internal microphone suffers greatly outdoors due to none or low noise-cancelling design. It is quite clear that they are not sufficient to capture the wide variety of sounds due to the high signal-noise-ratio, also known as *SNR*, in the $600 - 5000\ Hz$ band. The problem description clearly specifies that outdoor installations are required, which makes the external microphone the most appropriate choice for this system.

With the external microphone no complex calculations are needed to find sound events making Raw-data triggering, described in Chapter 2.1, a working and simple solution to implement.

3.2 Equipment

Table 3.1 lists the equipment used in the entire project. The project's test phase consisted of two major stages. The first stage was to implement a working localization-algorithm without the constraints or problems of a real-time system. To emulate the audio files that the clients would construct and send, a multitrack-recorder unit called Tascam DR-680 was used. This unit is designed to record multiple tracks with close to no delay between them, which was a demand for the first test stage. The unit the microphones need power to work, known as *phantom-power* that the multitrack recorder delivered in both stages.

When the first phase was done the multitrack-recorder was used merely to supply power and increase gain of the signals.

Each microphone is connected to a P3367-V camera that have an audio chipset that is ALSA supported with sampling frequencies up to 16 kHz.

The M5014 PTZ camera is a lot smaller than the Q6035 and made the installations easier when testing in the first phase. At the end of the project, a Q6035 was used instead since it was a much more sophisticated PTZ camera with a quicker response time.

Device	Description
Sony ECM-VG1 (a)	*A condenser microphone to be used indoors or outdoors.* *To reduce the noise from the wind, windscreens were applied onto the microphones. Five microphones were used during the project.*
Tascam DR-680 (b)	*A portable multitrack recorder used in the first test environment and to deliver phantom power to the microphones in the real-time system.*
Axis P3367-V (c)	*The network camera used during the course of the project.* *Each camera is coupled together with one of the microphones. This particular model does not have the ability to move and can not locate the position of the sound source when found.*
Axis M5014	*A lightweight Pan-Tilt-Zoom camera. Used in the first phase of the project.*
Axis Q6035	*The PTZ camera, used in the final phase of the project, since it is a much more capable camera than the M5014 camera.*
PoE Network switch	*Two different network switches were used, since all cameras required PoE, Power over Ethernet and one switch could only power a maximum of four devices.*

Table 3.1: *Table of the equipment and tools that were used in the project. Some of the devices can be viewed in Figure 3.6*

3.2.1 Power over Ethernet

The 802.3af PoE standard was developed to reduce wiring in larger networks where every device needed a power supply as well. It states that using mere CAT-5/5e Ethernet cables up to 100 m, devices could be powered with a minimum power of 12.95 W with 350 mA at a minimum voltage of 44 V [23].

The provided network cameras powered with PoE making installation a lot easier with only one cable instead of two or more.

Not every device is fit to deliver this power and Ethernet at the same time, therefore the provided network switches used in this project were compliant with the standard to deliver the required power to the devices.

(a) *The microphone, with the 'dead cat' applied, used when recording in and outdoors.*

(b) *The recorder used to sample, record and power the microphones.*

(c) *The clients, Axis P3367-V.*

Figure 3.6: *Some of the equipment that were used during the project.*

3.3 Synchronization

As described in Chapter 2.2.5, there are two ways of synchronize.

3.3.1 Network time protocol

The protocol-based solution utilizes the Network time protocol, or simply *NTP*, that ensures synchronized devices in the entire variable-latency network with a simple topology:

- *Stratum 0:*
  The top layer, composed of atomic, radio and GPS clocks. Extreme precision servers are situated in this layer and communicate directly to the lower layer with *e.g.* RS-232 cables.
- *Stratum 1:*
  Are used to answer NTP timing requests, from devices situated on layer Stratum 2.
- *Stratum 2:*
  Devices on this layer peer each other to exchange time information but performs also NTP requests from the Stratum 1 layer, to achieve a solid time. This layer is also used to handle NTP requests from Stratum 3.

A client that peers the different servers can achieve an acceptable precision with some calculations and measurements, that is implemented in NTP.

NTP version 3 assures a maximal error of a few milliseconds in local area networks. It is the most used version right now but version 4 has been available since June 2010 [12]. NTP version 4 has "*the potential accuracy to the tens of microseconds with modern workstations and fast LANs*" [11][12].

All of the Axis cameras implement an open-source peer-only version, of NTP version 3, called *OpenNTPD*, that has lower precision than the standard implementation of NTP version 3 and 4.

With this in mind, the accuracy is compromised even further considering Equation 2.14.

3.3.2 Server time broadcast

The final system is intended for local networks where there is reasonable low load on the system and low hop-counts for packets. This advantage is employed to construct another more simple implementation of a synchronization algorithm, called server time broadcasting.

At defined intervals, the server broadcasts its kernel time stamp to the clients that apply it in their separate calculations when fetching sound samples. Hence if the clients are on the same network switch the theoretical accuracy loss becomes a matter of measuring and compensating for the *Packet delay variation*.

Packet delay variation, also known as jitter, is "the difference between the one-way-delay of the selected packets" [24]. This means that for a packet that has e.g. 10 ms round-trip time and one that has 20 ms round-trip time, a jitter of 10 ms relative to the first packet and -10 ms relative to the second packet, is calculated.

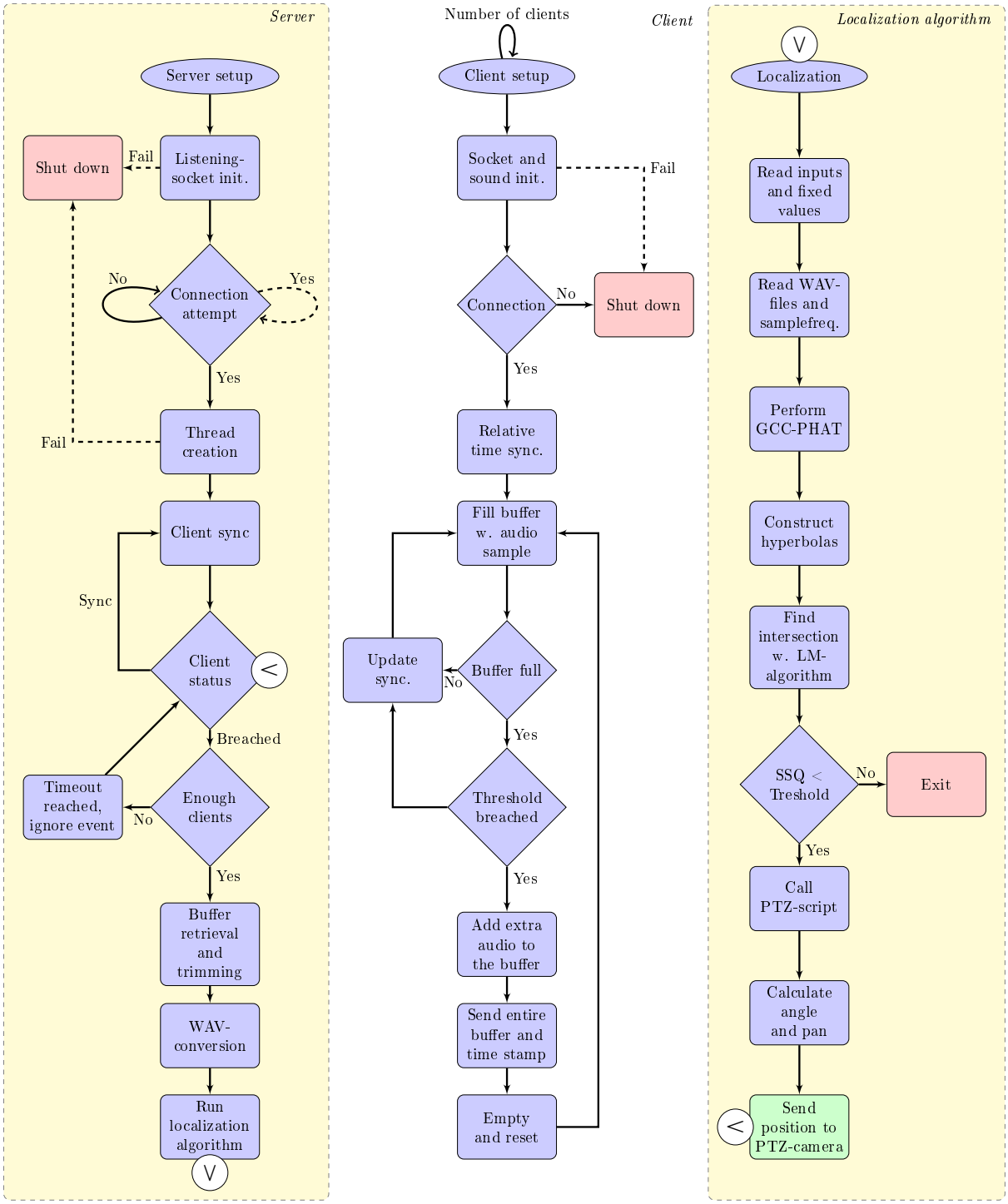
Using the *Internet Control Message Protocol*, a mean jitter of 0.20 ms was measured in a loaded local network with five clients. By applying Equation 2.14 the solution becomes reasonable as it gives a mere $A(0.2) \approx \pm 6.8$ cm precision loss, compared to OpenNTPD that has $n = 2$ $ms \Rightarrow A(2) = \pm 68.7$ cm.

If NTP version 4 was implemented in the cameras then the choice would be protocol-based, but as the current implemented protocol lacks precision, server time broadcast was adequate.

3.4 Programming

The goal of the implementation was to make the system as generic, in the aspect of client-count, as possible. Other inputs were evaluated to make the system as user-friendly as possible while at the same time be accurate and have advanced features, such as the option to deliver debug-data if requested.

The code was structured into the three distinguishable parts described in the report: the *server* and the *client* side are both written in C-language, whereas the *localization* algorithm is written with MATLAB-scripts.


Server
Server setup
Listening-socket init.
Fail
Shut down
Connection attempt
No
Yes
Yes
Thread creation
Fail
Client sync
Sync
Client status
Breached
Enough clients
No
Timeout reached, ignore event
Yes
Buffer retrieval and trimming
WAV-conversion
Run localization algorithm
Number of clients
Client
Client setup
Socket and sound init.
Fail
Connection
No
Shut down
Yes
Relative time sync.
Fill buffer w. audio sample
Buffer full
No
Update sync.
Yes
Threshold breached
Yes
Add extra audio to the buffer
Send entire buffer and time stamp
Empty and reset
Localization algorithm
Localization
Read inputs and fixed values
Read WAV-files and samplefreq.
Perform GCC-PHAT
Construct hyperbolas
Find intersection w. LM-algorithm
SSQ < Treshold
No
Exit
Yes
Call PTZ-script
Calculate angle and pan
Send position to PTZ-camera

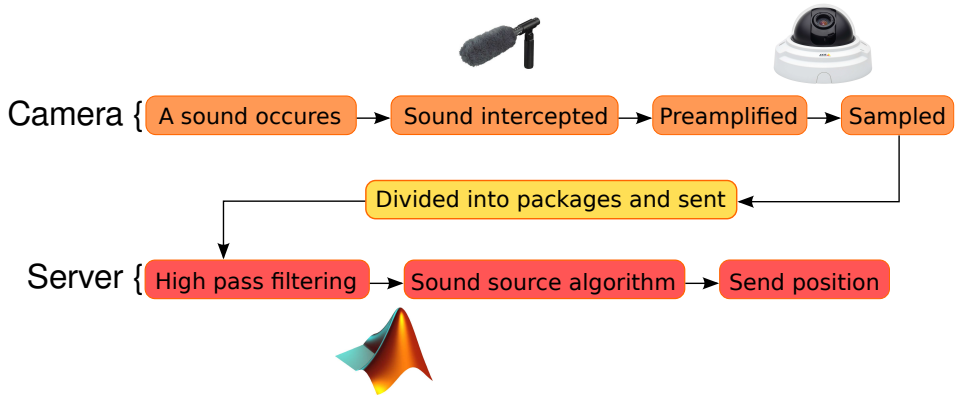


Figure 3.7: *The summarized data flowchart of the system, showing how the data is being processed throughout the system.*

3.4.1 Client implementation

The client's many states can be summarized into the states described by Figure 3.7 and a startup state which is not a part of the iterative process. The startup state consists of the connection with the server, the setup of the internal buffer and the initialization of the audio components. As audio is collected when measurable sound has occurred, it is preamplified and sampled with the Tascam DR-680. This amplification is needed to ensure good coverage and sensitivity. Furthermore in this phase the client sends a synchronization request in a fixed time interval to compensate for the clock drift.

The ring-buffer is also a part of the audio collection phase and allows for some pre-buffering of the data shall the threshold be breached when there is a slow amplitude increase. To gather the tail of the sound the sample collection iterates over a fixed time. This provides more crucial information when performing the correlation analysis which will improve accuracy.

Once the sound padding is done, the client divides the content of the ring-buffer into TCP packages and sends it to the server together with a time stamp that corresponds to the start of the buffer. This is used for trimming the audio files in the server implementation.

TCP provides reliability over latency which guarantees less errors in the correlation calculation.

3.4.2 Server implementation

The server's primary tasks is to keep track of the clients by sending synchronization time stamps, receive audio data, trimming the data and to start the localization algorithm when enough clients have sent data. The choice of running tasks in parallel was needed to run the server effectively. Developed in a Unix-environment, the intention of running on other OS:s such as Windows was conceived as irrelevant. This made the choice of how to thread the server easier and reduced the number of options to the following:

- *Forking* clones the process of the parent running the function and creates a child process that has a separate ID and a complete copy of the memory in a separate address-space.
- *POSIX threads* or more known as *pthreads*, takes full advantage of the capabilities provided by the definition of threads. It allows easy access to shared data with mutual exclusion in critical areas [26].

The need of having a shared address-space made the choice of using *pthreads* superior, even with the danger of having deadlocks or other phenomenon such as race conditions [28].

The result was a multi-threaded solution using a manager/worker thread model. Each client is supported with a single thread, having one separate thread listening for further connections to the system and one thread that evaluates the results when the system is triggered. This is shown in the system flow chart's first server-states.

When the Trimming-state is entered the time stamps are compared and the audio is trimmed to have the same start time. A more conservative option is to simply shift the file to preserve information, but since the synchronization differ theoretically only a couple of milliseconds, trimming of the signals becomes easier to perform and less costly time-wise.

The launch of the script is done from the server using the open source alternative called *Octave* that is able to interpret and run MATLAB-functions and commands.

3.4.3 Octave implementation

As described earlier, the implementation-process started out with the MATLAB-scripts, making offline calculations and optimizations on different parameters, such as algorithm speed. Octave gave many advantages with the library functions such as reading the wav-files, sampling speed, having a built-in FFT and easy accessible matrix/vector calculations.

Since the FFT algorithm usually takes $O(2N^2)$ computations for N points, the number of computations can easily become very large. One second of recorded sound, sampled with a frequency of 16 kHz will yield 16000 points. If $N = 16000$ then $2 \cdot 16000^2 = 512000000$ computations will be needed however if the number of points N is a power of two, the number of computations will only take $O(2N \cdot log_2(N))$. The closest number that is a power of two is $16383 = 2^{14}$. So

with $N = 16383$, the number of computations will be:

$$2 \cdot 16383 \cdot log_2(16383) = 32766 \cdot 14 = 458724$$

This is less than 0.09% of the computations done with the non-optimal solution. With this knowledge, the recorded sound will be padded up with zeros to the closest number that is a power of two and will thus improve the speed of the mathematical part of the algorithm.

GCC-PHAT was implemented using the built-in FFT due to the documentation stating that the algorithm implementation has been optimized and improved. The comparisons resulted in n vectors $[t_{1,2}, t_{1,3}, \ldots, t_{1,n}]$, where n was the number of clients in the system and $t_{i,j}$ is the TDOA between client i and j.

With different microphone references a TDOA matrix was constructed. Since the correlation between $t_{1,2}$ is equal to $-t_{2,1}$, the matrix can be written like this:

$$\begin{pmatrix} t_{1,1} & t_{1,2} & t_{1,3} & \ldots & t_{1,n} \\ t_{2,1} & t_{2,2} & t_{3,2} & \ldots & t_{2,n} \\ t_{3,1} & t_{2,2} & t_{3,3} & \ldots & t_{2,n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ t_{n,1} & t_{n,2} & t_{n,3} & \ldots & t_{n,n} \end{pmatrix} = \begin{pmatrix} 0 & t_{1,2} & t_{1,3} & \ldots & t_{1,n} \\ -t_{1,2} & 0 & t_{2,3} & \ldots & t_{2,n} \\ -t_{1,3} & -t_{2,3} & 0 & \ldots & t_{3,n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ -t_{1,n} & -t_{2,n} & -t_{3,n} & \ldots & 0 \end{pmatrix} \tag{3.1}$$

To simplify the expression, the matrix can be divided into two triangular matrices:

$$\underbrace{\begin{pmatrix} 0 & t_{1,2} & t_{1,3} & \ldots & t_{1,n} \\ 0 & 0 & t_{2,3} & \ldots & t_{2,n} \\ 0 & 0 & 0 & \ldots & t_{2,n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \ldots & 0 \end{pmatrix}}_{\mathbf{T}} + \underbrace{\begin{pmatrix} 0 & 0 & 0 & \ldots & 0 \\ -t_{1,2} & 0 & 0 & \ldots & 0 \\ -t_{1,3} & -t_{2,3} & 0 & \ldots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ -t_{1,n} & -t_{2,n} & -t_{3,n} & \ldots & 0 \end{pmatrix}}_{-\mathbf{T}} \tag{3.2}$$

and this can be viewed as

$$\begin{pmatrix} & \mathbf{T} \\ -\mathbf{T} & \end{pmatrix} \tag{3.3}$$

Only $\mathbf{T}$ needs to be calculated in order to derive all the time vectors in the TDOA matrix. Both multilateration and multiangulation require the use of the TDOA but they differ in the use of this information.

Important to the system was the question of accuracy, simplicity and computing speed. These three parameters do not need be implicated to each other, on the contrary they often are not as accuracy is often compromised when computing speed is important.

Described in Chapter 2.2.2 using multiangulation, the need of the angle of incident creates a new source of accuracy loss. The system cost is greater too due to the added microphones, not to mention that trigonometric functions are time-costly for as computer and approximated.

In Chapter 2.2.3 using multilateration, the only way to calculate the point is to have a TDOA matrix which promotes simplicity. The resulting system uses less

microphones and has no angle calculation either, which made the choice of using multilateration apparent.

Using the derived equation system the LM-algorithm calculates the final estimation. The overdetermined system used to calculate the position was optimized to increase accuracy as the microphone may have been corrupted with excess noise or other audio pollutants. The problem with the given hyperbolas, is to identify which of the hyperbolas are faulty or bad. The implementation is a simple comparison between every hyperbola to find a point with a small SSQ value. The estimation that has the smallest SSQ value becomes the point in the angle- and pan calculations. Finally a shell-script sends the parameters through the HTML-based *VAPIX* interface to the Q6035 camera.

Chapter 4
Results

The results from the developed system are presented in this chapter for both in- and outdoors. Some reasons and reflections regarding the results are also given to specific tests.

4.1 Outdoor testing

The system was designed to run the calculations on a computer, making the outdoor tests hard to carry out in real-time. It was therefore decided that the sound would first be recorded outside and then invoke the algorithm manually to get an estimated location of the source.

The recording equipment was placed in a roughly measured square that was approximately 25 m at each side. Two different sounds were produced at two different positions, a clap and a loud sound which gave a total of four different conducted tests. Each test contained several events, *e.g.* the first clap test consisted of six different claps. The tests were performed at $x_1 : (8.70, 8.50)$ and $x_2 : (9.70, 20.45)$ but these coordinates were coarsely measured by hand using a measuring tape. The error is derived by taking the Euclidean distance between x_1 or x_2 and the calculated point.

The sound recordings that were fed into the sound localization algorithm gave the following results:

Coordinates ($meter$)	Error ($meter$)
(8.78,8.8207)	0.3305
(8.8185,8.7697)	0.2945
(8.6406,8.4782)	0.0632
(8.9515,8.7621)	0.3632
(8.8629,8.6852)	0.2466
(8.7319,8.8727)	0.3740

Table 4.1: *The estimated coordinates based on a sound from the clap at position x_1, along with the estimated radius of error.*

The average of the coordinates are $(8.797567, 8.73143)$ giving an error estimate of $\approx 0.25\ m$.

The next sound was a loud human made speech or more specifically a word: The corresponding average of the estimated coordinates are $(9.0434, 8.8126)$ which

Coordinates $(meter)$	Error $(meter)$
(8.8684, 8.2554)	0.2969
(8.9031, 8.3255)	0.2677
(8.746, 8.1309)	0.3719
(10.28, 11.35)	3.2586
(8.6832, 8.4021)	0.0993
(8.7797, 8.4117)	0.1189

Table 4.2: *The estimated coordinates from the human made sound at position x_1 along with the estimated radius of error.*

is a slightly worse result compared to the test described by Table 4.1. The system's guess is however only off by $\approx 0.46\ m$. Two similar tests were conducted at position x_2 as well:

Ordinary clap at x_2:

Coordinates $(meter)$	Error $(meter)$
(9.1882, 21.286)	0.9802
(9.6249, 21.235)	0.7885
(9.6239, 21.364)	0.9171
(9.4570, 21.289)	0.8734
(9.5861, 21.693)	1.2482
(9.6467, 21.359)	0.7199
(9.4909, 21.359)	0.9327

Loud word at x_2:

Coordinates $(meter)$	Error $(meter)$
(10.327, 22.183)	1.8429
(10.066, 21.486)	1.0987
(9.6625, 21.628)	1.1785
(10.009, 21.961)	1.5422
(10.454, 21.685)	1.4469
(10.029, 21.643)	1.2375
(9.7858, 21.764)	1.3167
(9.5628, 21.689)	1.2465

Table 4.3: *The coordinates from both the clap and the loud word at position x_2.*

The mean coordinates from these tests are $(9.5168, 21.3420)$ for the clap and $(9.9870, 21.7549)$ for the loud word. The average error estimate was calculated to $\approx 0.91\ m$ and $\approx 1.33\ m$ respectively.

Since these tests were only recorded and then processed by the positioning algorithm, many low frequencies are still present in the audio files. By using the digital audio editor Audacity's built in function *Noise Removal*, a filtered signal was created. Audacity uses Fourier analysis to identify the pure tones and removes the frequencies that are considered to be noise [27]. The differences between the signals can be viewed in Figure 4.1.

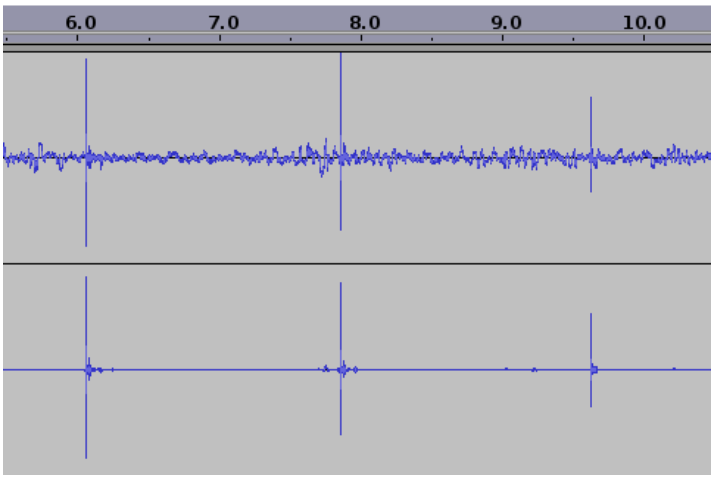


Figure 4.1: *The unfiltered signal (top) and the filtered signal (bottom).*

The filtered version of the recorded clap at position x_1 was processed to see if the results would improve.

Coordinates $(meter)$	Error $(meter)$
(8.8019,8.8428)	0.3576
(8.5167,8.9950)	0.5278
(9.2058,8.3700)	0.5222
(8.6905,8.5937)	0.0941
(8.8625,8.6857)	0.2467
(9.2544,8.5891)	0.5615

Table 4.4: *The results after the recorded sound have been filtered and then processed by the localization algorithm.*

The results are listed above in Table 4.4 and they yield an average of $(8.89, 8.68)$ which is $\approx 0.26\ m$ wrong. Comparing this result with the first outdoor test in

Table 4.1, the error estimate is not better but 0.01 m greater. This means that the signals are less correlated when filtered without precautions.

4.2 Real-time indoor testing

Most of the tests were conducted indoors in various rooms, even if the system was initially mostly intended to be used outside. When tests were carried out in the office landscape, the environmental noises were reversed giving strong reverberation and minimal weather effects. Even with the presumed better conditions the result fluctuated from positioning with close to no error, to positioning with an angular error of 180^o in camera space. This is likely caused by the shape and design of the office, mostly consisting of walls of glass. What also contributed to the poor result was that the system was connected to Axis own local area network. The network is often heavily loaded with video and audio streams from several other streaming cameras.

After those tests, a new test environment was decided upon. Instead a basement room was used for the indoors tests. This room has almost no glass windows and therefore more preferably to conduct tests in.

The outline can be seen in Figure 4.2 with the positions of the microphones S_1 to S_5 and the PTZ camera located at coordinate $(3.0, 3.3)$:

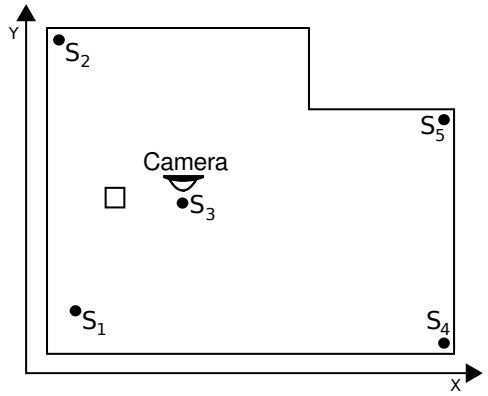


Figure 4.2: *A simple outline of the setup during the indoor tests with five microphones.*

Various types of sound were made to verify that the system would be able to detect all of them. Ordinary claps, thumps with a stick and thrown chairs are examples of sounds that were made. Two different tests were carried out and measured. In first test, a thump with a stick against the floor was done 146 times, all at different positions inside the enclosed area. The resulting frequency of deviation from the true positions can be viewed in Figure 4.3(a), with a mean of 0.535 m in deviation.

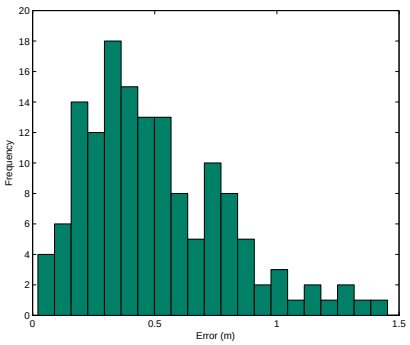
The system will almost never derive the exact position of the sound source. The interpretation of how exact the position is, may be a relative term. Even if the estimation is occasionally up to one meter from the true position, the estimation

Microphone	Coordinates ($meter$)
S_1	(0.0, 0.0)
S_2	(7.16, -0.6)
S_3	(3.0, 3.0)
S_4	(-0.6, 8.04)
S_5	(5.95, 8.04)

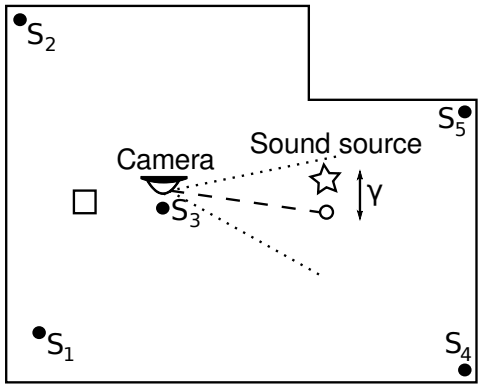
Table 4.5: *Coordinates for the setup during the tests indoors, seen in Figure 4.2.*

can nevertheless be regarded as 'correct' for the observer. The source of the sound is most of the time visible in the camera's field of view, *e.g.* a person or the broken window on a car. Out of the 146 tests done with the thump from the stick, the camera was unable to see the cause of the sound only two times. Hence the reliability is $\approx 98.6\%$. An example of this can be viewed in Figure 4.3(b).

Sometimes the system did not respond at all to an event. The cause is, as shown in the flow chart in Chapter 3.4, a large SSQ value from LM-algorithm making the derived position uncertain. When a large SSQ value is acquired, the intersection points between the hyperbolas are scattered, giving a bad estimation. The system then ignores the derived position and does not position at all. Another reason would be if the microphone did not register the sound event, *i.e* if the clap was not loud enough. The system did however responded to $\approx 88.2\%$ of the events.



(a) *The histogram of the deviation from the true position in the meters during the first test (thump with stick).*



(b) *Even though the system estimated the position γ meters wrong, the camera might still be able to visualize the sound source.*

Figure 4.3

In Figure 4.4 the positions where the sounds originated from are marked with crosses. From each point where a sound was made, a circle of deviations is drawn for that specific recorded event. This shows that most of the circles grow as

location of the sound source approaches the edges of the enclosed area. Also, as seen in Figure 4.2, there is a beam present in the enclosed area. This did most likely cause the larger radius on the estimated error circle around the beam.

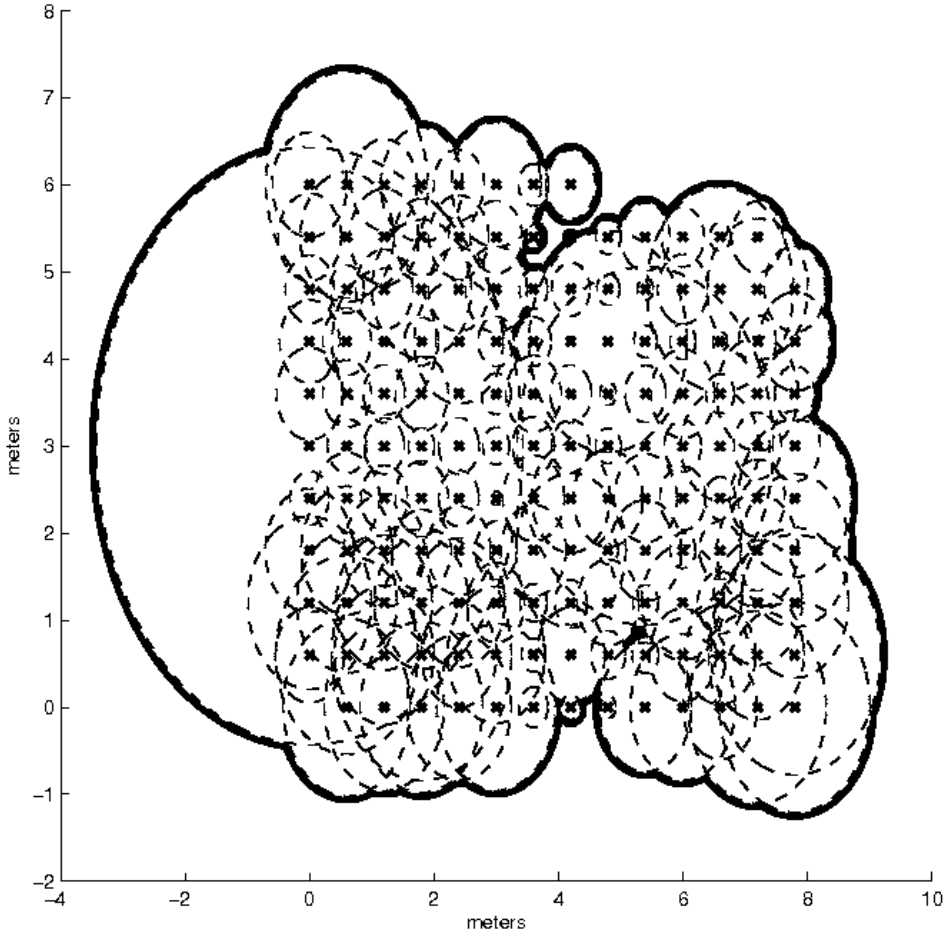


Figure 4.4: *Plot showing where the sound (thump with stick) originated from along with their corresponding estimated error circle.*

The next test was done with an ordinary clap, directed against the PTZ camera in the middle of the enclosed area. The graph in Figure 4.6 shows the histogram of the deviation from the true position. With a total of 148 claps and a mean deviation of approximately 0.762 m, it yields a slightly worse result comparing to the first test. During this test the system responded to $\approx 80.8\%$ out of 183 events and failed to position correctly four times in total, giving it a reliability of $\approx 97.3\%$.

Summing up both tests, 294 sound events were analyzed with a total mean deviation of 0.649 m and a total reliability value of 97.8% during the indoor testing and is illustrated in Figure 4.6 and Figure 4.7. Out of 262 events the system

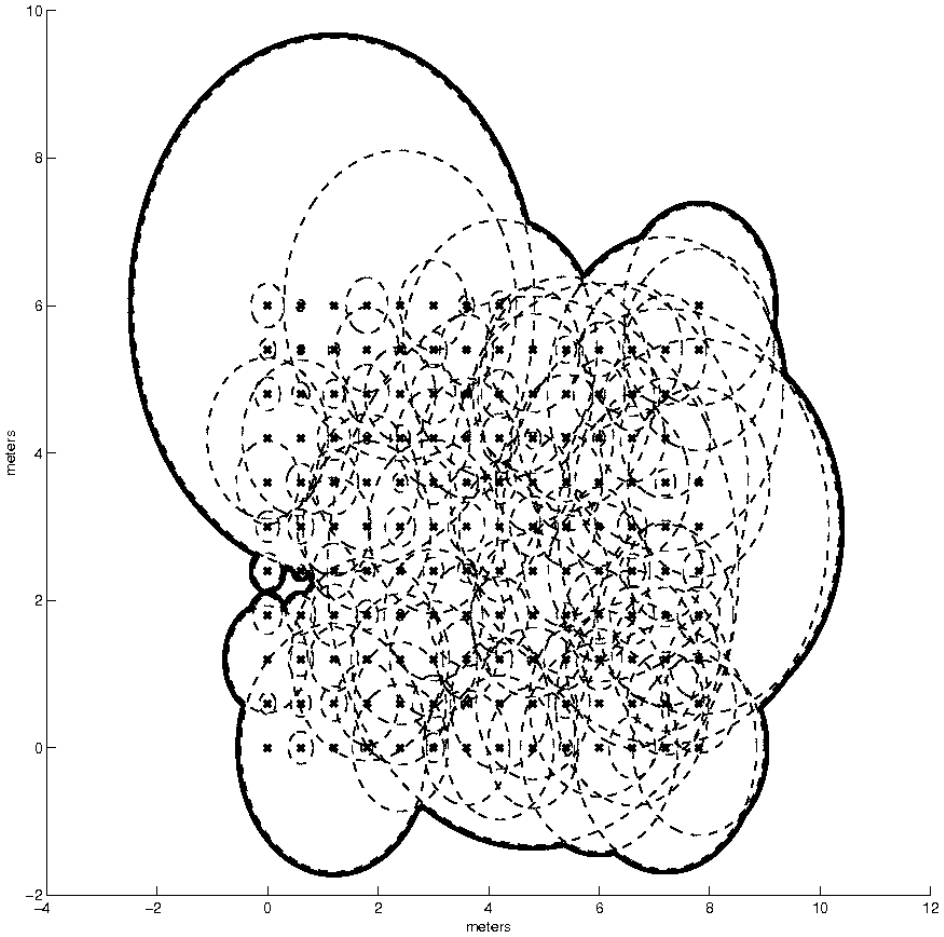


Figure 4.5: *Plot showing where the sound (ordinary clap) originated from along with their corresponding estimated error circle.*

responded to $\approx 82\%$ of them and positioned incorrectly 47 times.

The response time for the algorithm from the moment the server initiates the calculations to the position is sent to the camera, was measured and the average response time was 0.23495 s. The most time expensive part is the non-linear least squares, also known as NLMS, when the solution is sought amongst all the intersections.

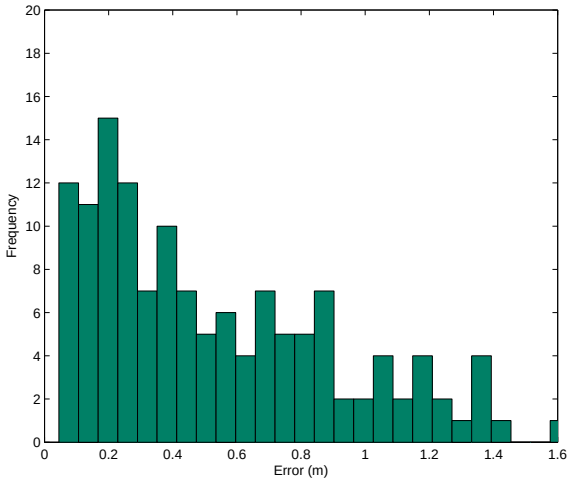


Figure 4.6: *The graph shows the histogram on the deviation from the true position in the meters during the second test (ordinary clap). Totally 148 events measured with a mean deviation of 0.762 meters.*

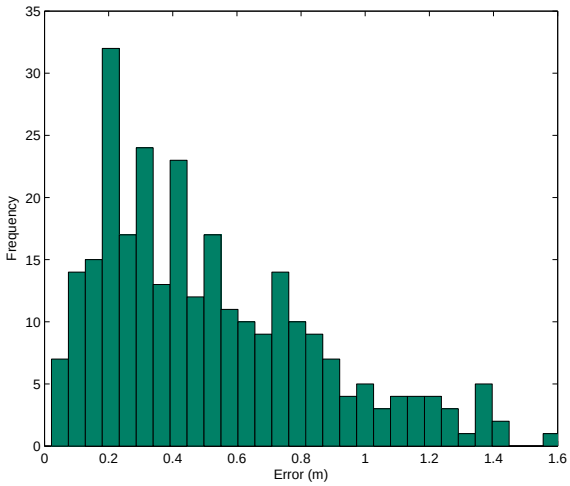


Figure 4.7: *The graph shows the histogram on the deviation from the true position in the meters based on the two tests indoors. Totally 294 measurements were made with a mean deviation of 0.649 meters.*

Chapter 5
Conclusions

The main purpose of the thesis was to create a system that is able to detect and locate sound events. By doing so, the system would know were the sound occurred and forward this position to a PTZ camera. A system was designed and constructed as described by the flow-chart in Chapter 3.4 and Figure 3.7, using the equipment described in Table 3.6. The clients synchronizes with the server by allowing the server to send its kernel time that they adjust upon at fixed time intervals.

When an event has occurred the server gathers the recorded data from all the triggered clients and trim the received data to make the starting sample for all audio clips correspond to the same time. The sound localization-algorithm, that is implemented with Octave, is then able to find a position by calculating using multilateration and the Levenberg-Marquardt-algorithm.

Several different types sounds and noise were tested, to see what the system would comprehend but mostly ordinary claps were tested. The indoor tests showed problems with reflections from walls, furniture and beams but the system gave often a reasonable result, giving a position with a small error relative to the analyzed area. During the outdoor tests, the sound was only recorded offline then later computed with the sound localization-algorithm. These tests did also show relatively good results with a reliability of over 90 %. Making the PTZ camera very likely to capture the sound source within its field of view.

Since the tests outdoors were fairly few in comparison with how many were conducted indoors, it can be a bit optimistic to assume that the system would perform as well outdoors as indoors. Nevertheless, should the system be placed outdoors with optimal weather conditions, it will not have the major issue of reverberation or other types of noise making the system, most likely, reliable.

With some improvements of the system and better conditions, the results of the sound localization could be enhanced in many ways such as precision, calculations speed and filtration.

In conclusion, the system performed as expected.

Chapter 6
Discussion

6.1 Future work

The localization can be used to perfect many already existing features. The features that analyze images to find certain objects can use this system in synergy with its own position localization algorithm. An example of such a feature is motion detection, that can be initialized by finding the position of the sound source first and then analyze the surroundings for motions.

By characterizing the sound of the event such as weapons, screams or vehicles, sound localization can be used in parallel to enhance security and capture the specific event. Other environments have restrictions on video capture of certain objects. Sound matching can be used in this system as well, giving the localization-algorithm a security check before positioning on the event.

Even though the sound detection system is working, it contains some flaws and weaknesses that is known already. These accuracy losses were described throughout the project and are described below:

Client buffer When designing the client program for the cameras, it was decided that the buffer containing the raw sound data would only be able to transmit the buffer if it had been filled. Thus will the system only be able to detect sound events when the buffer was full. Since the buffer is cleared once an event occurs, the system becomes 'deaf' during the time it refills the buffer. This has an advantage being the ability to recover and not adjust the PTZ constantly, the disadvantage being the obvious ability to trick the system during the refill process. This becomes an optimization problem to find the optimal size of the ring-buffer.

Client hardware The audio code that is implemented supports only low sampling frequencies that allow for accuracy losses described in Chapter 2.4. With higher frequency the accuracy increases at a cost of more samples to calculate upon. The timing and synchronization problem has roots in the hardware too. The need of microsecond precision requires the use of NTP version 4 which was unfortunately not portable due to the complexity of the program code. Future versions of the camera may support NTPv4 and therefore become more accurate than using Server time broadcast.

Filtering Since the cameras only send raw data directly to the server, no filtering

is done at all. An improvement would be filter the data in the buffer before sending it to the server, thus removing uninteresting information from the signal.

Numerical errors When constructing the TDOA matrix at section 3.4.3 the top triangle should be mirrored in the diagonal expect the change of sign. When the matrix however was computed by the server, this was not the case. There were small differences between the the mirrored elements, *e.g.* $t_{1,2}$ and $t_{2,1}$. This is caused by small numerical error when computing the values of the time differences. In this thesis, in order to both achieve better results and to improve the speed of the system, it was decided that instead of deriving the whole matrix, only the upper triangular matrix was needed. With this, the whole matrix could easily be done by only copying the elements already calculated and changing signs on them, as described in section 3.4.3.

Alternatives to the LM-algorithm There are other numerical approximations to the non-linear least squares problem, such as particle filters that use Monte Carlo based techniques. Other options include interpreting the bad sound audio clips properly and not analyze them at all.

References

[1] Ralph Bucher and D. Misra
A Synthesizable VHDL Model of the Exact Solution for Three-dimensional Hyperbolic Positioning System
Department of Electrical and Computer Engineering,
New Jersey Center for Wireless and Telecommunication,
New Jersey Institute of Technology,
Newark, NJ 07102, USA, 2001.

[2] Bassilio Dahlan, Wathiq Mansoor, Milad Abbasi, Parham Honarbakhsh
Sound Source Localization for Automatic Camera Steering
American University in Dubai School of Engineering
Department of Electrical and Computer Engineering, 2011.

[3] K. Björk, N. Svensson
Embedded Sound Localization on an Axis Camera.
Deparment of Electrical and Information Technology,
Faculty of Engineering, LTH, Lund University, 2011.

[4] G. Valenzise, L. Gerosa, M. Tagliasacchi, F. Antonacci, A.Sarti
Scream and Gunshot Detection and Localization for Audio-Surveillance System
Dipartimento di Elettronica e Informazione - Politecnico di Milano, 2007.

[5] Alice Clifford, Josh Reiss
Calculating Time Delays of Multiple Active Sources In Live Sound
Centre for Digital Music, Queen Mary
University of London, London, E1 4NS, UK, 2010.

[6] Lantmäteriverket
Anvisningar För Planläggningsmätning
Lantmäteriverkets Centralförvaltningen

[7] Fredrik Gustafsson and Fredrik Gunnarsson
Positioning Using Time-difference of Arrival Measurments
Department of Electrical and Engineering,
Linköping University, SE-581 83 Linköping, Sweden.

[8] Bert Van Den Broeck, Alexander Bertrand, Peter Karsmakers,
Bart Vanrumste, Hugo Van hamme, Marc Moonen
Time-domain GCC-PHAT Sound Source Localization For Small Microphone Arrays
ESAT, KU Leuven, IBBT, Future Healt Department –
Kasteelpark Arenberg 10, 3001, Heverlee, Belgium
MOBILAB, KH Kempen –
Kleinhoefstraat 4, 2440, Geel, Belgium

[9] Manolis I.A. Lourakis and Antonis A. Argyros
Is Levenberg-Marquardt the Most Efficient Optimization Algorithm for Implementing Bundle Adjustment?
Institute of Computer Science, Foundation for Research and Technology - Hellas Vassilika Vouton, P.O. Box 1385, GR 711 10, Heraklion, Crete, Greece

[10] Stephanie L. Heath and Gerry L. McAninch
Propagation Effects of Wind and Temperature on Acoustic Ground Contour Levels
NASA Langley Research Center
Hampton, Virginia 23681-2199

Websites:

[11] David L. Mills
Network Time Protocol (Version 3) Specification, Implementation and Analysis
`http://tools.ietf.org/html/rfc1305`
University of Delaware, 1992.
Read 2012-10-24

[12] David L. Mills
Network Time Protocol Version 4: Protocol and Algorithms Specification
`http://tools.ietf.org/html/rfc5905`
University of Delaware, 2010.
Read 2012-10-24

[13] John Loomis
Cross Correlation
`http://www.johnloomis.org/ece561/notes/xcorr/xcorr.html`
University of Dayton, 2005 Read 2013-02-18

[14] Konowa.de
Position Determination with GPS
`http://www.kowoma.de/en/gps/positioning.htm`
Read 2013-02-20

[15] United States Coast Guard
LORAN-C General Information
`http://www.navcen.uscg.gov/?pageName=loranMain`
Read 2013-02-20

[16] Eric W. Weisstein - MathWorld
Method of Steepest Descent
http://mathworld.wolfram.com/MethodofSteepestDescent.html
Read 2013-02-21

[17] Eric W. Weisstein - MathWorld
Hyperbola
http://mathworld.wolfram.com/Hyperbola.html
Read 2013-02-21

[18] Julius Orion Smith III
Mathematics of The Discrete Fourier Transform (DFT)
https://ccrma.stanford.edu/ jos/st/
Read 2013-02-26

[19] Eric W. Weisstein - MathWorld
Fast Fourier Transform
http://mathworld.wolfram.com/FastFourierTransform.html
Read 2013-02-27

[20] Eric W. Weisstein - MathWorld
Levenberg-Marquardt Method
http://mathworld.wolfram.com/Levenberg-MarquardtMethod.html
Read 2013-02-27

[21] C. Sidney Burrus
The Cooley-Tukey Fast Fourier Transform Algorithm
http://cnx.org/content/m16334/latest/
Read 2013-03-11

[22] Advanced Linux Sound Architecture (ALSA) *Introduction*
http://www.alsa-project.org/main/index.php/Introduction
Read 2013-03-14

[23] Hirschmann Automation and Control GmbH
Power over Ethernet - IEEE 802.3af
http://www.belden.com/docs/upload/poe_basics_wp.pdf
Read 2013-03-14

[24] C. Demichelis, P. Chimento
IP Packet Delay Variation Metric for IP Performance Metrics (IPPM)
http://tools.ietf.org/html/rfc3393#section-1.1
Read 2013-03-14

[25] Eric W. Weisstein - MathWorld
Sampling Theorem
http://mathworld.wolfram.com/NyquistFrequency.html
Read 2013-03-13

[26] Barney Blaise - Lawrence Livermore National Laboratory *POSIX Threads Programming* https://computing.llnl.gov/tutorials/pthreads/ Read 2013-03-19

[27] Audacity
How Noise Removal Works
http://wiki.audacityteam.org/wiki/How_Noise_Removal_Works
Read 2013-03-19

[28] FreeBSD Developers' Handbook
http://www.freebsd.org/doc/en/books/developers-handbook/secure-race-conditions.html Read 2013-03-22

Images:

[29] Axis Camera P3367
http://www.axis.com/files/image_gallery/ph_p3343_p3344_ceiling_mount.jpg
27-12-2012

[30] Tascam DR-680
http://www.performanceaudio.com/images/products/212/20293_1.jpg
27-12-2012

[31] Sony ECM-VG1
http://www.visuals-switzerland.net/4133-thickbox/sony-ecm-vg1-electret-condenser-microphone.jpg
27-12-2012

[32] Axis Q6035
http://www.axis.com/files/image_gallery/low_res/ph_q6034e_left_1108_low.jpg
03-01-2013