

Detection and Localisation of Gunshots Using Sound Data

Martin Chan, Sofie Karlsson

Master's thesis
2018:E30



LUND UNIVERSITY

Faculty of Engineering
Centre for Mathematical Sciences
Mathematics

LUND UNIVERSITY

CENTRE FOR MATHEMATICAL SCIENCES

MASTER THESIS

Detection and Localisation of Gunshots Using Sound Data

Authors

Sofie Karlsson
Martin Chan

Supervisors

Kalle Åström
Gabrielle Flood
Arvid Nihlgård Lindell
Danny Smith

Abstract

The goal of this master's thesis is to detect and position sharp sounds using Axis speakers with built-in microphones. Sharp sounds of special interest are gunshots. The system needs at least five speakers to function and is designed for usage in indoor environments. The project follows a pipeline in order to position sound sources containing recording, synchronisation, detection of gunshot in sound data, and positioning of the sound. Detection of gunshots in recorded files is done by a binary classification with a deep neural network created in Python. The algorithms for positioning are implemented in MATLAB. The final neural network has an accuracy of 98%. It is pretrained by VGG-team with data from ILSVR and transfer learning is applied to fit the model for gunshot data. After testing a few methods to synchronise the speakers and to calculate the position of the sound source, the final system has a mean error of 0.28 m. The model's precision is adequate for large areas.

Acknowledgments

Thanks to Axis communications for providing the equipment and space necessary for this master's thesis. A huge thanks to our supervisors Danny Smith and Arvid Nihlgård Lindell for providing knowledge and software within GStreamer and ALSA. We also want to thank Carl Hansson for lending out equipment for the tests. Thanks to Mazdak Farzone for an early introduction and ideas to solving the problem.

This master's thesis would not have been completed without the knowledge and expertise of our supervisors at Lund University. We are thankful for all the great ideas provided to us from Kalle Åström and Gabrielle Flood.

Lastly, a special thanks to Niklas Carlsson and Hanne Heingård, our fellow student colleagues at Axis Communications, for helping us with C++ code.

Contents

1	Introduction	1
1.1	Related Work	1
2	Theory and Method	2
2.1	Recording	3
2.1.1	GStreamer	3
2.2	Detection of Sound	4
2.3	Classification of Sound	5
2.3.1	Model	5
2.3.2	Parameters and Structure	7
2.4	Estimation of TDOA	9
2.4.1	Cross Correlation	9
2.4.2	Generalised Cross Correlation with Phase Transform	10
2.4.3	Flank Detection	11
2.4.4	Adaptive Eigenvalue Decomposition	11
2.5	Positioning of Sound	13
2.5.1	Multilateration	13
2.5.2	Levenberg–Marquardt	15
2.5.2.1	Gradient Descent	15
2.5.2.2	Gauss-Newton	16
2.6	Synchronisation with Calculation of Latencies	16
3	Tools and Equipment	18
4	Set-up and Results	19
4.1	Performance of the Neural Network	20
4.2	Method for Estimation of the TDOA	21
4.3	Synchronisation of the Speakers	23
4.3.1	Synchronisation with AudioInterleave in GStreamer	23
4.3.2	Synchronisation by Adding Latencies to the Equation System	23
4.3.3	Synchronisation Results	23
4.4	Visualisation of the Performance	25
5	Discussion and Conclusions	26
5.1	The Neural Networks	26
5.2	Information in z-axis Lost	26
5.3	Choice of Method for Estimation of TDOA	27
5.4	Synchronisation	27
5.5	The Final System	28
5.6	Future Work	30
	References	31

1 Introduction

When utilising a surveillance system, the exact positions of sound sources can provide additional information about events. In this thesis, speakers located in an indoor environment with built-in microphones are used to detect and localise gunshots. However, as real gunshots could not be tested, the localisation part was done on sound from two spoons hitting against each other instead.

The main goal of this thesis is to achieve adequate localisation results using microphones inside speakers. The sound data will be less distinct and resonance may occur as the receivers are located inside a small protective covers. An important step in positioning of sound sources are the estimations of time difference of arrivals, or TDOAs, to the microphones. Four different methods to calculate this quantity are explored in order to find a good alternative for sound from spoons hitting against each other. A synchronisation problem is also introduced as the system records and transfers sound data over a network. Different approaches to solve this issue will be tested and evaluated. It turns out that this issue is crucial to solve as our sound source positioning system requires all recorded files to be synchronised down to milliseconds.

In addition to position sound sources, the possibilities to detect gunshots in recorded sound files are also investigated. For this task, neural networks are used. The question whether it is possible for a machine to accurately separate gunshots from other sounds in a recorded file will be studied.

1.1 Related Work

Positioning of sound sources is a common problem approached by many scientists and engineers [2], [16], and [14]. Mazdak Farzone and Kim Smidje analysed different method for estimation of TDOAs in order to direct a camera and capture events in real-time [13]. They faced a network latency problem, just like us, as the recording was done with microphones connected to a joint network instead of with audio cables. This was also a master's thesis at Axis Communications. The difference between their thesis and ours is the sounds analysed. Farzone and Smidje focused on continuous sounds such as music and car engines instead of gunshots and sound from spoons hitting against each other.

In an article published by Shotspotters [26], gunshot localisation used to counter snipers are briefly described. The environment considered is a huge outdoor area as opposed to this thesis where the main goal is to find and localise sounds indoor. In a third related work, the TDOAs are used to estimate positions of transmitters and receivers. The experiments were tested on real data; in this article the data were claps, collected using microphones and a recorder by Kalle Åström and Yubin Kuang. To get the TDOA measurements from the collected data, the generated sounds were matched to the recorded sound flanks from the microphones [21]. As opposed to us, Åström and Kuang did not assume knowledge of the positions of the receivers while we in this thesis, measure the positions of the microphones with measuring tape.

2 Theory and Method

In order to detect and position a gunshot sound, a few steps are needed. These are presented in Figure 2. Firstly, the sound needs to be recorded. If any sharp sounds are detected in the recording, they are cut out and saved in another file. If not, the recording is no longer interesting for our purposes. In this thesis the words sharp sounds and flanky sounds is used to describe a sudden sound with high amplitude at the beginning of the signal, as can be seen in Figure 1. Examples of sharp sounds are claps, balloons popping, gunshots, and two spoons hitting against each other.

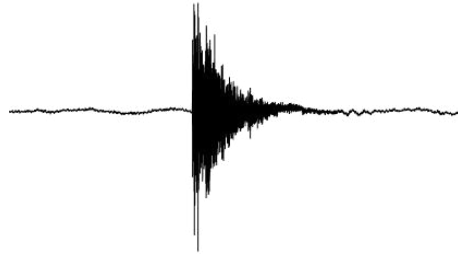


Figure 1: A recording of a sharp/flanky sound.

The saved file, containing the sharp sounds, is sent to be analysed in search for gunshots. If gunshots are present, the time differences of arrival to different microphones needs to be calculated and then the gunshot can be positioned.

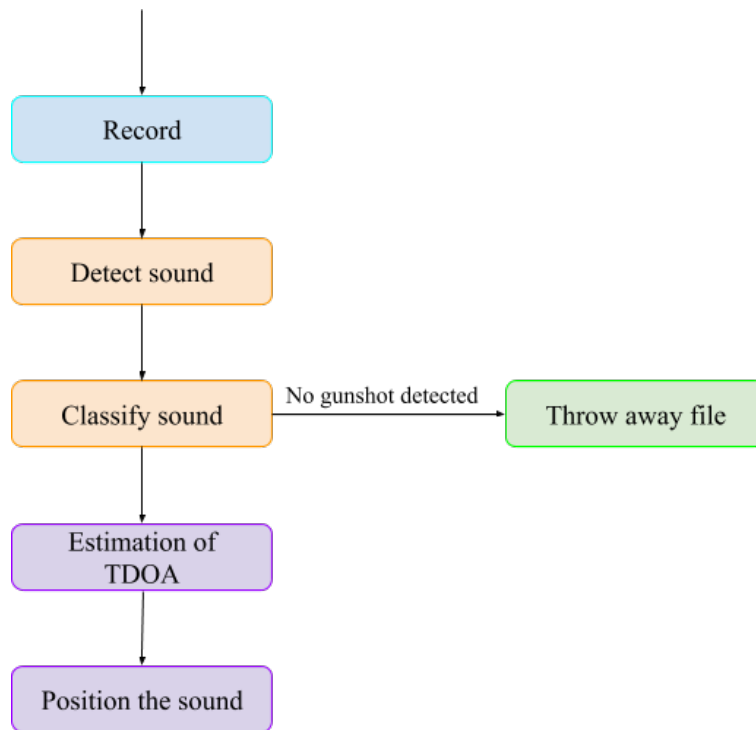


Figure 2: Our pipeline with illustrated steps that leads to positioning of the sound source.

2.1 Recording

The protocol to transport real time data over the Internet is called Real-time Transport Protocol and is used worldwide. It transports audio and video for both media and Internet telephony. This standard is actually two protocols, RTP and RTCP. RTP is used to transport data, it supports applications with continuous media (such as audio and video). It also has loss detection, content identification, security and timing reconstruction. RTCP is used to control and get feedback about the quality of the data flow, it supports real-time video-conferences of any size within the Internet and can synchronise these calls for different medias [34]. A framework that supports RTP is GStreamer [10] and it can be used for recording. Another way to record is for example with Phonon, but since GStreamer is more used at Axis Communications this method was chosen for our thesis.

When data from different devices are sent to a central unit over a network, some kind of latency will always be introduced. Because of this latency, the speakers need to be synchronised which can be done in various ways. In this master's thesis, two very different approaches to solve the synchronisation problem are tested. The first method is to use the object "AudioInterleave" in GStreamer which uses timestamps on data chunks to sort the recorded signals (see section 2.1.1 for further knowledge about this method). This is done continuously through a recording. The second method is to add the latencies to the equation system acquired during the positioning task (section 2.6 covers this in depth). In Figure 3 the pipeline has been redrawn to show where the synchronisation can take place.

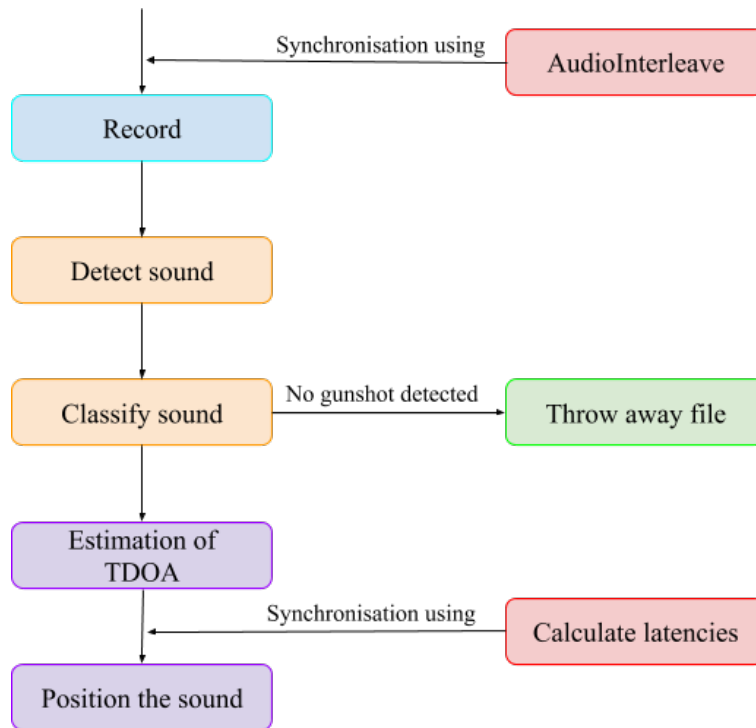


Figure 3: Modified pipeline with illustrated steps that lead to positioning of the sound source. In this image, synchronisation steps are included. The methods to synchronise are tested separately and together.

2.1.1 GStreamer

One way to synchronise the speakers is to use the framework GStreamer, which is also used to create streaming media applications that handles audio and video. However, it is not restricted to audio

and videos only as GStreamer can handle any kind of data flow. The framework is a good choice to build media players because of its many components that can support different audio types. To define the flow of the data, pipelines are used [11]. A pipeline is the steps a media travels through to get from the source element to the sink element. It can be described as the steps a product needs to pass to go from producers to consumers. In between, there are numerous tasks that need to be done; this is a pipeline [9]. Pipelines in GStreamer can also be used when developers want to mix and match different plugin components [11].

In this project, GStreamer was used to stream the data recorded by the speakers to the user's computer. By specifying the computer's IP to the program every speaker knows where to send the recorded data. This way, longer recordings could be made without having a problem with the size of the storage on the speakers. The speakers are connected one by one to the port in the computer. For the program to work, one of the speakers needs to be the clock-master. This means that other speakers will listen to one speaker when the timestamps are set.

The program starts with connecting the speakers and deciding a clock-master. All speakers and the computer use the clock-master in order to have the same view of current time which is necessary for synchronisation of the audio streams. The streams are transmitted using UDP, User Datagram Protocol, which is a connectionless communication protocol widely used for time-sensitive applications where one favours losing data instead of introducing latency. [20]. The data is sent to a jitter buffer, which is a temporary buffer where the data is stored. Because of network latencies there might be a risk that data is lost if it is sent too early from the jitter buffer. To avoid this, extra time is added before the data is transported to an element called "AudioInterleave". This object collects the data from the jitter buffers and uses the timestamps on the data to place them in the right order. Once the synchronisation is done, the data is saved.

2.2 Detection of Sound

Detection of gunshots in a sound file can be done by a flank detection algorithm. Let

$$a_m = \frac{1}{m} \sum_{k=1}^m |s(k)|$$

be the average of the absolute value of the first m samples of a signal s with n samples, where $m \ll n$. By moving this average forward in the signal, a flank (see Figure 1) can be detected very precisely by comparing the next sample with the moving average. Let us denote this time of detection with t_f acquired from

$$a_m(i) = \frac{1}{m} \sum_{k=i+1}^{m+i} |s(k)| \quad 0 \leq i \leq n - m \quad (1)$$

$$\text{where } t_f = m + i + 1$$

$$\text{if } |s(m + i + 1)| > h \cdot a_m(i).$$

The threshold, h , is a parameter used to check if a flank is detected. It needs to be calibrated together with the length of the moving average, m , as these can vary for microphones in different environments. Table 1 shows the values used in this thesis. They are calibrated from tests in a quiet office.

Table 1: Parameters for the flank detection

h	10
m	1000

In addition to gunshots, this flank detection algorithm will detect other kind of sharp sounds such as screams, balloon pops, and hand claps. To distinguish different sharp sound types, or more specifically, detect gunshots, a deep artificial neural network is used.

2.3 Classification of Sound

Deep artificial neural networks are machine learning models gaining more and more attention [12]. Trying to mimic a human brain, they consist of neurons connected to each other in layers with weights. By tuning these weights, a model can classify classes and predict outcomes. Unlike most other machine learning algorithms, neural networks do not need guidance once the model is set-up. With enough labelled data to train on, it can adjust itself without manual modification. However, this means that a lot of data need to be presented to it. Otherwise, the model can not draw any conclusion. In this thesis, the basics of neural networks will not be described. The curious reader is encouraged to read chapter 5 in [5].

2.3.1 Model

Convolutional neural networks, CNN, have been successful in image classification problems and show promise for audio as well (see [5] for more information on CNNs). By transforming wav-files into log-mel spectrogram, sound data can be used as input to a CNN. In this thesis, the data transformation (from wav to spectrogram) is not implemented and code from [8] are used instead. To get more information about the method, detailed description can be found in [17]. A mel-spacing in the spectrogram is desired as the mel-scale is judged by listeners to be equal in distance from another. A popular formula to convert frequency into the mel-scale is

$$m = 2595 \cdot \log_{10}\left(1 + \frac{f}{700}\right)$$

where f is the frequency [23].

A VGG16-network pretrained by the VGG-team with substituted top layers is used as a base model [27]. Different variations of top layers can be experimented with to optimise the performance of the network (see section 2.3.2 for these designs). Figure 4 shows the structure of a VGG16 network. Worth to note is that this neural network is not designed in this thesis but rather by the VGG-team in Oxford. As the task is to classify only gunshots, the output layer consists of only one node reading 1 if gunshots are detected, 0 otherwise.

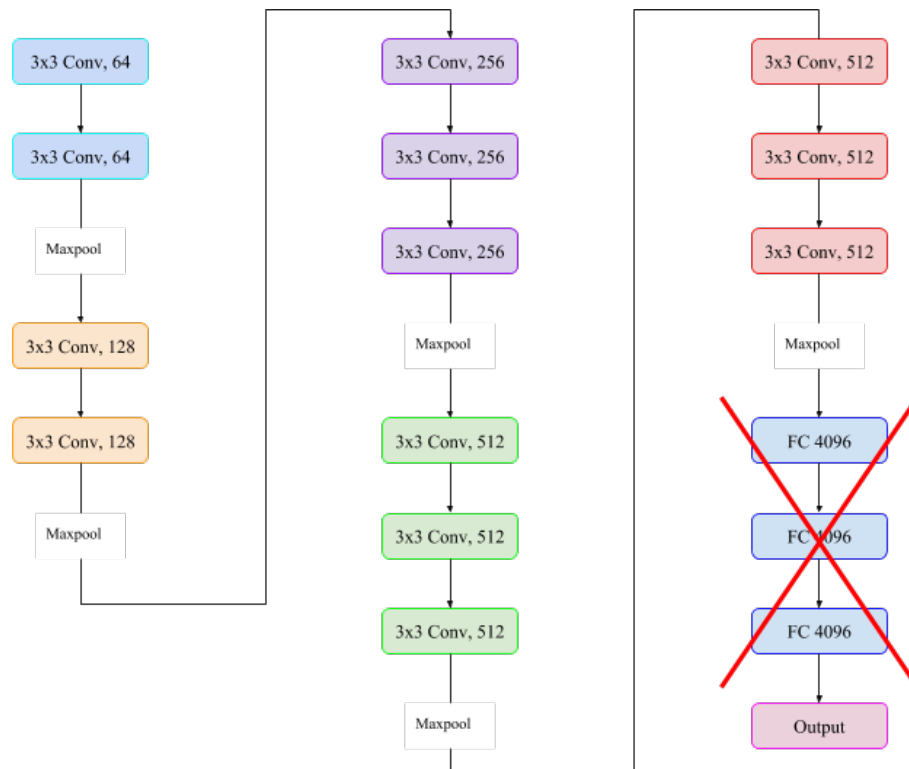


Figure 4: The structure of VGG16 network containing 13 convolutional layers, 3 fully connected layers, and an output layer. The fully connected layers (the three blocks named "FC 4096") are not used in this thesis and will be replaced, therefore the red cross. Worth to note is that this network is not designed in this thesis but rather by the VGG-team in Oxford.

Figure 4 contains blocks each representing a layer in VGG16. 13 of the blocks are labelled "3x3 Conv" followed by a number. These represents the 13 convolutional layers. 3x3 means that the filters in the layer contains 9 weights dimensioned 3x3 (see [5] for more information about filters in convolutional layers). The numbers 64, 128, 256, and 512 after "3x3 Conv" represents how many filters are applied. This number is increased with the depth of the network as late layers analyses more specific data compared to early layers which filters more generic data. "Maxpool" is seen between some of the layers. This is a method to decrease the dimensionality in an image as it passes through the network resulting in less total parameters in the model. For every patch, 2x2, of an image, maxpool decreases four pixels into one by selecting the maximum value. Figure 5 illustrates it.

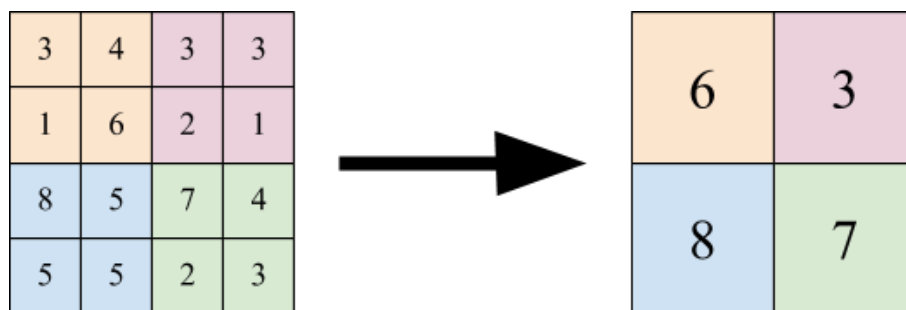


Figure 5: An illustration of Maxpool. As indicated by the name, the maximum value for every 2x2 patch is used.

The three crossed out blocks labelled "FC 4096" in Figure 4 represents the fully connected layers at the end of VGG16. The number 4096 shows that each layer contains 4096 nodes. An illustration of fully connected layers can be seen in Figure 6. As mentioned before, details about neural networks will not be introduced in this thesis and the curious readers are directed to chapter 5 in [5].

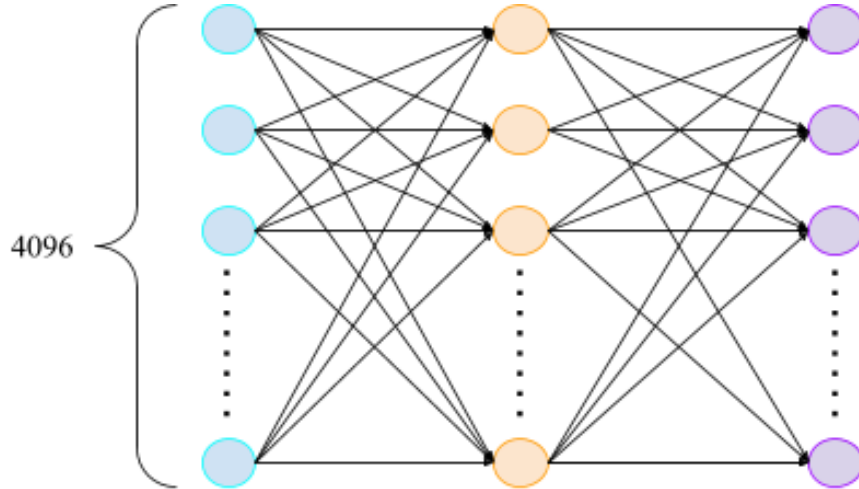


Figure 6: An illustration of fully connected layers in neural networks. Every node in one layer is connected to every node in the next layer. The number 4096 shows how many nodes every layer contains. This number can of course vary from model to model.

The optimisation algorithm chosen is Adam. It is a computationally efficient stochastic gradient descent algorithm requiring little memory [18]. Gradient descent will be further described in section 2.5.2.1. Adam is short for Adaptive Moment Estimation and it, in addition to descending, adaptively updates learning rates according to averages and standard deviations of past gradients (first and second moment estimates). This trait makes Adam a suitable optimiser for smaller data sets as it can adapt quicker to new features. When using Adam as optimiser, there are some parameters that need to be chosen. These are the learning rate α , the first and second order decay rate for the moment estimates β_1 and β_2 , and a constant, ϵ , to prevent division by zero.

2.3.2 Parameters and Structure

The parameters for the training of the neural network can be seen in Table 2. These values are proposed by the authors of [18], the inventors of Adam, to be used as default values. Tuning of these parameters were tested in this thesis without improvement, and therefore, the default parameters were kept.

Table 2: The parameters for the ADAM-optimiser and training.

α	0.0001
β_1	0.9
β_2	0.999
ϵ	0.00000001

The pretrained VGG16 has only been exposed to image data from ILSVRC and thus, all weights are

unlocked for training in order to fit all layers to the new log-mel spectrogram data [24]. Unlocking all weights for training means that every parameter in every layer can be tuned. Of course, this results in slower training sessions. However, every session will be more efficient as more variables can be changed. As the top layers of the VGG16-model are removed they need to be redesigned. We have chosen to try three different patterns of top layers. The first one is to simply put the features extracted from the base layers directly to an output layer. Let us name this model "Direct model". The second one is three fully connected layers with 2048 nodes respectively between the base layers and output. We call this model "FC 2048". Lastly, five fully connected layers in decreasing sizes are added between the base layers and output layer. The last model is named "Diminution model" for its decreasing sizes of fully connected layers. Figure 7 illustrates all of the designed top layers alternatives.

A common problem for deep neural networks is overfitting of data. This phenomenon occurs when the same data is presented to the model session after session and results in a network that can not predict or classify new datasets. Dropout between the fully connected layers is added in our model as seen in Figure 7; it is a technique used to prevent overfitting. For every training session, or epoch, all nodes go through a decision whether to be activated or not with a chance of 50% for each outcome. The term "dropout" refers to the nodes and their connections getting inactivated or dropped out. As different nodes gets inactivated each time, the model will be slightly different every epoch. This results in less co-adaptions from the nodes producing a more flexible model.

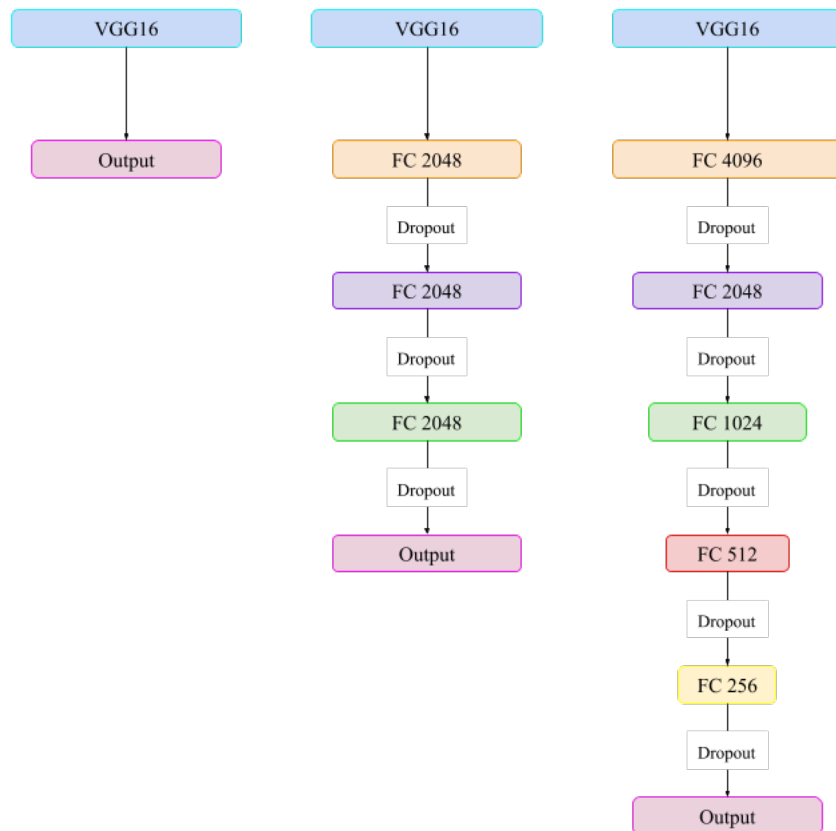


Figure 7: Three different approaches of design to the top layers for the pretrained VGG16. From left to right they are called "Direct model", "FC2048", and "Diminution model". Similar to Figure 4, FC 2048 refers to a fully connected layer with 2048 nodes.

2.4 Estimation of TDOA

When a gunshot is detected in a recorded file, the data needs to be analysed in order to position the sound source. The first step is to estimate the Time Difference of Arrival for the signal to reach microphone one as compared to the other microphones. This measure is later going to be used in order to calculate the position of the sound source. The TDOA is the difference in time it takes for a signal from the sound source to reach two microphones (or another type of receiver). Figure 8 is an illustration of this phenomenon. Even though the TDOA is a measure in time it will sometimes be used as a distance in our thesis (by multiplying the TDOA with the speed of sound).

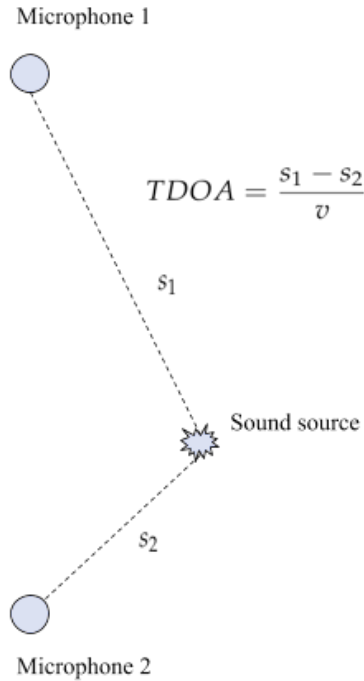


Figure 8: Illustration of the difference in distance in which a sound travels to reach different microphones. The direct paths s_1 and s_2 from the sound source to microphone one and two respectively have been drawn out. The speed of sound is denoted v .

In this master's thesis, four different methods to estimate the TDOA are tested and the one most suited for sharp sounds will be chosen. The selection is based on calculations of the mean error between estimated position and measured position of the sound source (the positioning is described in section 2.5).

2.4.1 Cross Correlation

The TDOA of two signals can be calculated using the cross correlation. Let the received signal in microphone one be denoted s_1 and microphone two s_2 , with a time lag defined as τ . The cross correlation for s_1 and s_2 is then defined as

$$[s_1 \star s_2](\tau) = \int_{-\infty}^{\infty} s_1^*(t) s_2(t + \tau) dt, \quad (2)$$

where $*$ is the complex conjugate. Since recorded signals are real and not complex, $s_1^*(t)$ is equal to $s_1(t)$ in this thesis. In the discrete case, the integral is replaced with a sum

$$[s_1 \star s_2](\tau) = \sum_k s_1(k)s_2(k + \tau).$$

The TDOA can be estimated as

$$\text{TDOA} = \underset{\tau}{\operatorname{argmax}}([s_1 \star s_2](\tau))$$

as the maximum value of $[s_1 \star s_2](\tau)$ is where the two signals correlates best. The τ that corresponds to the maximum value is the time difference between the signals. A way to visualise how the TDOA is obtained is to think of signal s_1 sliding over signal s_2 calculating the product at each point and then taking the sum over that. The highest value corresponds to where the signals match the most. The shift that corresponds to this value is the TDOA.

It is common to calculate the cross correlation in the frequency domain because there will then be no need for time shifting. This is done by using the Discrete Fourier transform of (2) [19]. The transformation F_m of a signal f_k is defined as

$$F_m = \sum_{k=0}^{M-1} f_k e^{-2\pi i m k / M}$$

where M is the number of samples in f_k [31]. From (2), (3) is acquired after transforming it with the Discrete Fourier Transform,

$$[s_1 \star s_2](\tau) = S_1(k) \cdot S_2(k) \quad (3)$$

where $S_1(k)$ is the Fourier transform of $s_1(k)$. The calculated value from (3) is then transformed back to the time domain by inverse Fourier transforming [19].

2.4.2 Generalised Cross Correlation with Phase Transform

Cross Correlation is not robust against noise and echo paths and therefore a weight can be added in the Fourier transform to reduce distractions. It is called the Phat weight, or Phase Transform. The weight is added to (2) and it preserves the phase where the information about the time difference lies but it also normalise frequency magnitudes. The Generalised Cross Correlation (GCC) with Phase Transform is then defined as

$$[s_1 \star s_2](\tau)_{Phat} = \int_{-\infty}^{\infty} s_1(t)s_2(t + \tau)\varphi(t)_{Phat}dt. \quad (4)$$

Again this is transformed to the frequency domain and the weight φ is defined as follows, where S_1 and S_2 are the Fourier transforms of s_1 and s_2 ,

$$\varphi(f)_{Phat} = \frac{1}{|S_1(f) \cdot S_2(f)|}. \quad (5)$$

The reason why GCC-Phat is better at estimating the TDOA than Cross Correlation is because theoretically, the result from equation (4) is a unit impulse function and noise and echos will have less influence in the correlation. Again the TDOA is the index of the maximum value of $[s_1 \star s_2](\tau)$ [7],

$$\text{TDOA} = \underset{\tau}{\operatorname{argmax}}[s_1 \star s_2](\tau)_{Phat}.$$

Noise can sometimes be very small and even close to zero. This will lead to division by zero in (5) and to avoid this problem, a condition is added. The condition is that if $|S_1 \cdot S_2| < T$ then instead $|S_1 \cdot S_2| + 1$ will be divided with. This can be written as

$$\varphi(f)_{\text{phat}} = \frac{1}{|S_1(f) \cdot S_2(f)| + \epsilon'}$$

where

$$\epsilon = \begin{cases} 1, & \text{if } |S_1(f) \cdot S_2(f)| < T \\ 0, & \text{otherwise.} \end{cases}$$

A typical value on T is $5 \cdot 10^{-3}$. The TDOA will still be the τ corresponding to the maximum value of $[s_1 \star s_2](\tau)_{\text{phat}}$, but with the new weight function.

2.4.3 Flank Detection

For a sharp sound, the flank detection algorithm described in section 2.2 can be used to estimate the TDOA. For a system with two microphones receiving their signals from the same source, let t_{f1} and t_{f2} be the timestamp for microphone one and two respectively acquired from (1). The TDOA can then be estimated with

$$\text{TDOA} = t_{f1} - t_{f2}.$$

2.4.4 Adaptive Eigenvalue Decomposition

All of the methods to estimate the TDOAs assume the signals from a sound source follows an ideal model, this model comes from the classic time delay estimation problem and looks like

$$x(n) = \alpha \cdot s(n - \tau) + b(n).$$

In this model only the attenuation factor, α , (how much the signal is weakened when it passes through objects), the delay τ , and some additive noise, b around the microphone are taken into account. In a more realistic model, the reverberation must also be included. The reverberation is reflections of the signal that occurs when it hits walls and objects, these reflections are later absorbed by other objects. In a typical indoor environment, because of the reverberation and background noise, the peak of a cross correlation might not be well defined or might not correspond to the TDOA.

In the new model, the impulse response from the sound source to the microphone is analysed instead. The model for the received signal is described as

$$x_i(n) = g_i \star s(n) + b_i(n).$$

Here, $b_i(n)$ is the additive noise at the i :th microphone, $s(n)$ is the signal from the sound source and $g_i(n)$ is the acoustic impulse response between the signal to the i :th microphone [2]. In a system that is linear, time invariant and with no noise, with only two microphones and one sound source, the relation can be defined as

$$x_1(n) \star g_2 = s(n) \star g_1 \star g_2 = x_2(n) \star g_1.$$

This can be rewritten into matrix form,

$$\mathbf{x}_1^T(n) \mathbf{g}_2 = \mathbf{x}_2^T(n) \mathbf{g}_1,$$

where

$$\begin{aligned}\mathbf{x}_i(n) &= [x_i(n) \quad x_i(n-1) \quad \dots \quad x_i(n-M+1)]^T \quad i = 1, 2 \\ \mathbf{g}_i &= [g_{i,0} \quad g_{i,1} \quad \dots \quad g_{i,M-1}]^T \quad i = 1, 2.\end{aligned}$$

Here, M is the length of the impulse response vector (same as the length of the received signal) and $\mathbf{x}_i(n)$ is the vectors containing the samples from the signal emitted from the sound source. Once again this can be rewritten as

$$\mathbf{x}^T(n)\mathbf{u} = \mathbf{x}_1^T(n)\mathbf{g}_2 - \mathbf{x}_2^T(n)\mathbf{g}_1 = 0, \quad (6)$$

where

$$\begin{aligned}\mathbf{x}(n) &= [\mathbf{x}_1(n)^T \quad \mathbf{x}_2(n)^T]^T \\ \mathbf{u} &= [\mathbf{g}_2^T \quad -\mathbf{g}_1^T]^T.\end{aligned}$$

Multiplying (6) with $\mathbf{x}(n)$ on the left side and then calculating the expectation gives us the following relation

$$\mathbf{R}_x \mathbf{u} = \mathbf{0}.$$

Here, $\mathbf{R}_x = E[(\mathbf{x}(n) - E[\mathbf{x}(n)])(\mathbf{x}^T(n) - E[\mathbf{x}^T(n)])]$ is the covariance matrix of the the input signal from the microphones. Since $\mathbf{x}(n)$ is zero mean, $\mathbf{R}_x = E[\mathbf{x}(n)\mathbf{x}^T(n)]$. This means that \mathbf{u} is the eigenvector to \mathbf{R}_x corresponding to the eigenvalue 0.

As the rank of the covariance matrix for the signal source $s(k)$ is full and the z-transform of the impulse responses does not share any common zeros, it is possible to uniquely determine the impulse responses g_1 and g_2 . Since this theory is not needed to implement the algorithm the interested reader can get more information from [33].

If noise is added to the room, the model will change and the eigenvalue for the covariance matrix \mathbf{R}_x will no longer be zero. An algorithm is needed to find the eigenvectors corresponding to the smallest eigenvalue of \mathbf{R}_x . This algorithm is adaptive and starts with computing the error signal as

$$e(n) = \hat{\mathbf{u}}^T(n)\mathbf{x}(n). \quad (7)$$

The notation $\hat{\mathbf{u}}$ means that the value on \mathbf{u} has been updated, the new values on $\hat{\mathbf{u}}$ are calculated using

$$\hat{\mathbf{u}}(n+1) = \frac{\hat{\mathbf{u}}(n) - \mu e(n)\mathbf{x}(n)}{\|\hat{\mathbf{u}}(n) - \mu e(n)\mathbf{x}(n)\|}, \quad (8)$$

with the constraints

$$\|\hat{\mathbf{u}}(n)\| = 1, \quad (9)$$

and the adaptation step μ . The algorithm will update the impulse responses M times in a loop using (7), (8) and (9). The algorithm was tested on simulations and the value on μ was decreased until the loop converges, the final value on μ was 0.6. The time for convergence varies depending on the length of the signal.

The impulse responses have been calculated using the algorithm above and the TDOA is then defined as follows [3]

$$\text{TDOA} = \underset{n}{\operatorname{argmax}} |\hat{g}_{2,n}| - \underset{n}{\operatorname{argmax}} |\hat{g}_{1,n}|.$$

Here, \hat{g}_2 is the first half of $\hat{\mathbf{u}}$ and \hat{g}_1 is the second half, the notation is again that these values have been updated using equation (8).

To implement this algorithm, the impulse responses need to be initialised. The vector \mathbf{u} that contains the responses g_1 and g_2 is set to zero at every position except one. At position $u_{M/2}(0)$ it should be equal to 1. Because of this, the delays can now be both positive and negative. This value should be kept dominant in the first half of vector \mathbf{u} (that is g_2) and it represents the direct path to the microphone. A mirror effect will appear in the second half of \mathbf{u} which will be the estimate of the direct path to the other microphone. As mentioned above the time difference of arrival matches the shift between these two peaks [4].

2.5 Positioning of Sound

To position a sound source with the acquired TDOAs, there are different techniques. Most commonly, multiangulation or multilateration are used depending on the setup. Multiangulation assumes that the distances from sound source to the microphones are large and that the distances between microphones are small. The receivers also need to be in pairs in order to find the position of the sound source [13]. Multilateration does not have these requirements and it is possible to calculate the position for the sound source once the TDOAs are estimated [16]. Therefore, this method is chosen for this master's thesis.

2.5.1 Multilateration

The TDOAs are calculated using the methods mentioned in section 2.4. These values contain the information of the difference in distance from a sound source to two microphones. In other words it is the distance from sound source to the first microphones subtracted with the distance from the sound source to the second microphone. If the sound source is located at (x_s, y_s, z_s) and microphone one and two at (x_1, y_1, z_1) and (x_2, y_2, z_2) , the TDOA can be written as

$$TDOA_{12} = \sqrt{(x_1 - x_s)^2 + (y_1 - y_s)^2 + (z_1 - z_s)^2} - \sqrt{(x_2 - x_s)^2 + (y_2 - y_s)^2 + (z_2 - z_s)^2}. \quad (10)$$

There are three unknown variables in (10) meaning that more equations are needed in order to get an unambiguous solution (see Figure 9 for an illustration).

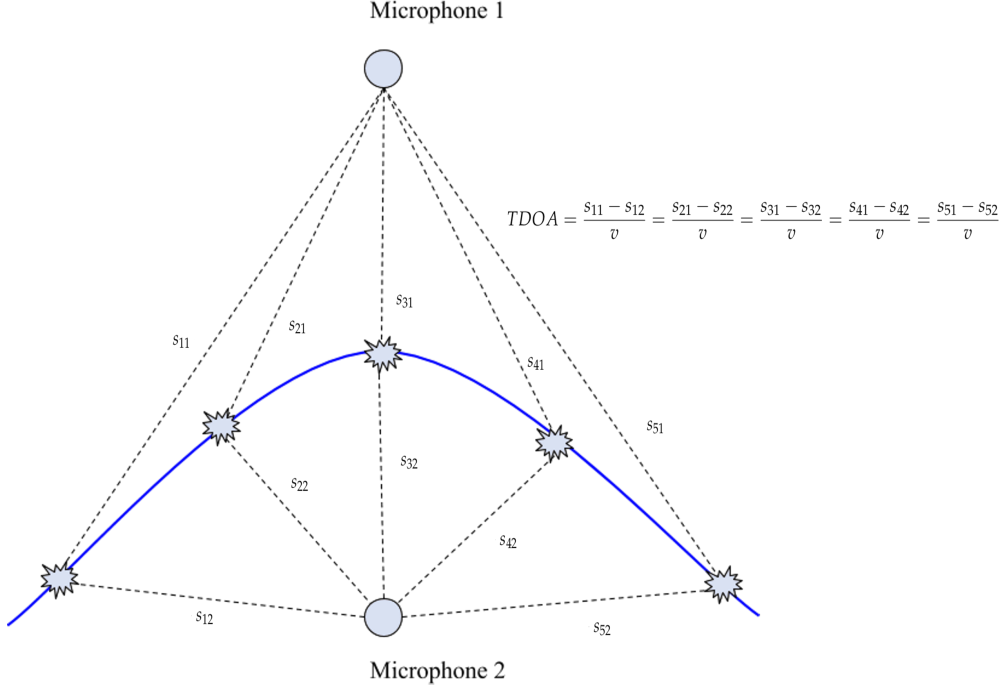


Figure 9: With only one estimated TDOA, the sound can originate from anywhere following a hyperbolic curve. The difference in distances between s_{11} and s_{21} is the same as between s_{12} and s_{22} for example. However, their sound sources come from very different positions. The speed of sound, denoted v is used to convert TDOAs to difference in distances instead of time.

Four microphones are needed in order to determine the position of one sound source. With four speakers there will be three estimated TDOAs. Lets denote them with $TDOA_{12}$, $TDOA_{13}$, and $TDOA_{14}$. $TDOA_{23}$, $TDOA_{24}$ and $TDOA_{34}$ can not be used as they do not contain extra information.

To find the position of the sound source, three equations are set-up as seen in (11). These equations are then minimised to find the (x_s, y_s, z_s) that gives the smallest error which is estimated as the sum of squares of

$$\begin{aligned} err_1 &= TDOA_{12} - \left(\sqrt{(x_1 - x_s)^2 + (y_1 - y_s)^2 + (z_1 - z_s)^2} - \sqrt{(x_2 + x_s)^2 + (y_2 - y_s)^2 + (z_2 - z_s)^2} \right), \\ err_2 &= TDOA_{13} - \left(\sqrt{(x_1 - x_s)^2 + (y_1 - y_s)^2 + (z_1 - z_s)^2} - \sqrt{(x_3 + x_s)^2 + (y_3 - y_s)^2 + (z_3 - z_s)^2} \right), \\ err_3 &= TDOA_{14} - \left(\sqrt{(x_1 - x_s)^2 + (y_1 - y_s)^2 + (z_1 - z_s)^2} - \sqrt{(x_4 + x_s)^2 + (y_4 - y_s)^2 + (z_4 - z_s)^2} \right). \end{aligned} \quad (11)$$

To get a clearer picture of what happens, the equations can be set up for two dimensions and (10) will then be the definition of a hyperbola [32]. In Figure 10, the hyperbolas for microphone two and microphone three has been drawn. These lines are calculated using possible different positions of the sound source and in the figure, some of these possibilities are shown. One of these sound sources have a darker colour (red) and it is the position obtained from minimising the error functions in (11) (but for two dimensions). This position is exactly where the hyperbolas for microphone two and microphone three intersects [16].

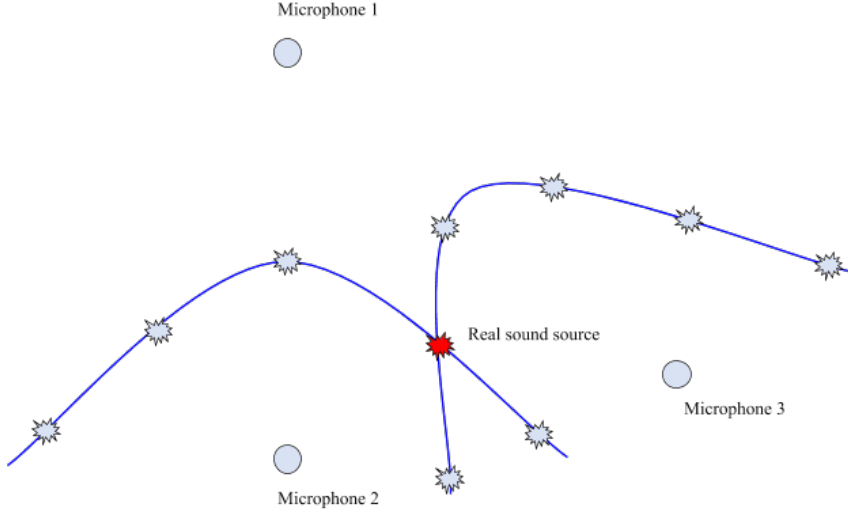


Figure 10: The image shows the microphones and the hyperbolas computed by inserting different positions for the sound source in (10) (in 2D). The red sound source is the position that minimises (11) (in 2D).

To minimise the equations, an optimisation algorithm is needed. A popular one is Levenberg-Marquardt used in related work such as [13] and [21]. Without further research, this algorithm is also chosen for this master’s thesis. Also, the algorithm is never implemented, but rather MATLAB’s built-in function “*lsqnonlin*” (with Levenberg-Marquardt as an option) is used.

2.5.2 Levenberg–Marquardt

The optimisation algorithm Levenberg–Marquardt is a combination of two methods, namely the Gauss-Newton method and the gradient descent method (for further reading on these methods see [29] and [15]). It solves nonlinear least squares problems by computing the sum of the squared errors between the function and the data points, the χ -function, and then minimises it. When the initial guess is close to the optimal value, the algorithm works more like the Gauss-Newton method and when the initial guess is further away from the optimal point, it acts more like gradient-descent. For a sound source positioning optimisation problem, the χ -function looks like

$$\chi^2 = \sum_{i=1}^n \left[TDOA_i - y(x_s, y_s, z_s) \right]^2$$

where

$$y = \left(\sqrt{(x_1 - x_s)^2 + (y_1 - y_s)^2 + (z_1 - z_s)^2} - \sqrt{(x_4 - x_s)^2 + (y_4 - y_s)^2 + (z_4 - z_s)^2} \right).$$

2.5.2.1 Gradient Descent

Gradient descent sets the parameterised χ -squared error function and updates the parameters iteratively along the negative gradient until convergence. The negative gradient is calculated as

$$\begin{aligned}
-\frac{\partial}{\partial \mathbf{p}} \chi^2 &= 2 \sum_{i=1}^n [TDOA_i - y(\mathbf{p})] \cdot \frac{\partial}{\partial \mathbf{p}} y(\mathbf{p}) \\
\Rightarrow -\frac{\partial}{\partial \mathbf{p}} \chi^2 &= 2(\mathbf{TDOA} - \mathbf{y})\mathbf{J},
\end{aligned}$$

where $\mathbf{J} = \frac{\partial}{\partial \mathbf{p}} y(\mathbf{p})$ is the Jacobian and the parameters \mathbf{p} are all the unknown positions and latencies (Δ_i, x, y, z) . For every iteration, the updates will be done by

$$\Delta \mathbf{p} = a(\mathbf{TDOA} - \mathbf{y})\mathbf{J} \cdot \mathbf{p},$$

where a is a parameter which decides how much the unknowns will be updated every iteration.

2.5.2.2 Gauss-Newton

Gradient-descent converges for simple functions and is a good choice when there are multiple parameters involved. However, the Gauss-Newton method converges faster but it assumes the function being approximately quadratic and the updated value on the initial point is close to the local minimum. With these two requirements met, one iteration will update the parameters according to [6]

$$\mathbf{p} = -(\mathbf{J}^T \mathbf{J})^{-1} \mathbf{J}^T \mathbf{y}(\mathbf{p}).$$

For the equations in (11), the Levenberg-Marquardt algorithm will detect if the error functions are quadratic and close enough to the local minimum. At this point, Gauss-Newton will be used instead of Gradient descent.

2.6 Synchronisation with Calculation of Latencies

The latency introduced by recording over a network can be dealt with by adding a latency parameter for every latency in the system of equations. This can be done either as a substitution or addition to the "AudioInterleave" object mentioned in section 2.1.1. Assuming the offset for every device is constant, (10) can be modified as

$$TDOA_{12} + \Delta_{t12} = \sqrt{(x_1 - x_s)^2 + (y_1 - y_s)^2 + (z_1 - z_s)^2} - \sqrt{(x_2 - x_s)^2 + (y_2 - y_s)^2 + (z_2 - z_s)^2},$$

where Δ_{t12} is the difference of start time for microphone one and two when recording, here presented in distance by multiplying the latency with the speed of sound. Now, four microphones will not be enough to determine the positions of the sound source as the offset in time is also an unknown variable. In fact, we have introduced an amount of new unknowns equal to the number of speakers minus one (the first speaker is of course synchronised to itself). To get around this problem, more speakers needs to be put into the system. However, for every speaker added, a new latency variable is also introduced and thus, the number of equations will never reach the number of unknowns. For a specific event, when a series of several sharp sounds are present (note they can not happen at the same time), there is a solution to the problem mentioned above.

The trick is to position several sound sources at once having in mind that the offset in time for the different devices are constant. Using more than four speakers/microphones, every sound event will add additional equations to the system. The goal is to have the same amount of unknown variables and equations. The dependency can be explained with the following equations

$$\begin{aligned}
(x-1) \cdot y &\geq (x-1) + 3 \cdot y, \\
\Rightarrow y &\geq \frac{x-1}{x-4} \quad x > 4,
\end{aligned}$$

where y is the number of sound events and x is the number of microphones in the system. If the number of sound events are less than the required amount, the method will not work. With 6 microphones for example, 3 sound events need to be present. The new equation system is a modified version of (11). With three sound events, a , b , and c and 6 microphones, the following equations are acquired and minimised

$$\begin{aligned}
err_1 &= TDOA_{a12} + \Delta_{t12} - \left(\sqrt{(x_1 - x_a)^2 + (y_1 - y_a)^2 + (z_1 - z_a)^2} - \sqrt{(x_2 - x_a)^2 + (y_2 - y_a)^2 + (z_2 - z_a)^2} \right) \\
err_2 &= TDOA_{a13} + \Delta_{t13} - \left(\sqrt{(x_1 - x_a)^2 + (y_1 - y_a)^2 + (z_1 - z_a)^2} - \sqrt{(x_3 - x_a)^2 + (y_3 - y_a)^2 + (z_3 - z_a)^2} \right) \\
&\vdots \\
err_6 &= TDOA_{b12} + \Delta_{t12} - \left(\sqrt{(x_1 - x_b)^2 + (y_1 - y_b)^2 + (z_1 - z_b)^2} - \sqrt{(x_2 - x_b)^2 + (y_2 - y_b)^2 + (z_2 - z_b)^2} \right) \\
err_7 &= TDOA_{b13} + \Delta_{t13} - \left(\sqrt{(x_1 - x_b)^2 + (y_1 - y_b)^2 + (z_1 - z_b)^2} - \sqrt{(x_3 - x_b)^2 + (y_3 - y_b)^2 + (z_3 - z_b)^2} \right) \\
&\vdots \\
err_{15} &= TDOA_{c16} + \Delta_{t16} - \left(\sqrt{(x_1 - x_c)^2 + (y_1 - y_c)^2 + (z_1 - z_c)^2} - \sqrt{(x_4 - x_c)^2 + (y_4 - y_c)^2 + (z_4 - z_c)^2} \right).
\end{aligned} \tag{12}$$





3 Tools and Equipment

The microphones used for this master’s thesis were the built-in microphones in Axis Communications product C1004-E Network Cabinet Speakers. These were connected to PoE Network switches with network cables and could be connected to a main computer. The cabinet speakers are mainly used to play background music and for live and scheduled announcements in stores. The microphones are used to test if the speakers are working, it does this by playing a series of test tones that is measured by the microphone. The microphones are placed at the back side of the speakers in a protective cover, which makes it difficult to know how well the microphones receive sound and how much internal vibrations and noise interferes with the recording.

Since a synchronisation problem is present with the speakers, an alternative is also used in order to try out the algorithms in a more forgiving setup. TASCAM DR-680 with SONY ECM-VG1 microphones does not introduce any synchronisation problem and are also better at receiving sound than the built-in microphones in the speakers. In Table 3, the equipment is listed.

As mentioned before, sharp sounds are of interest in this thesis. It would not be appropriate to do tests with guns, thus, spoons hitting against each other are used as sound source instead. The speakers will never be used to generate sounds.

Table 3: The equipment used in the project

Equipment	Image
C1004-E Network Cabinet Speakers [1]	
PoE Network Switch [22]	
TASCAM DR-680 [30]	
SONY ECM-VG1 [28]	

4 Set-up and Results

All tests for this master's thesis were conducted in an office environment. It consists of shelves where the microphones/speakers were placed, tables, and chairs. Figure 11 illustrates the office viewed from above. The width of the room is 4 m and the length is 10 m. Sound made by hitting two spoons against each other are used to mimic gunshots. The coordinate system defined can be seen in Figure 11. The z-axis corresponds to the height of the room and the floor will therefore get value $z = 0$. The x- and y-axis represents the width and length of the room respectively.

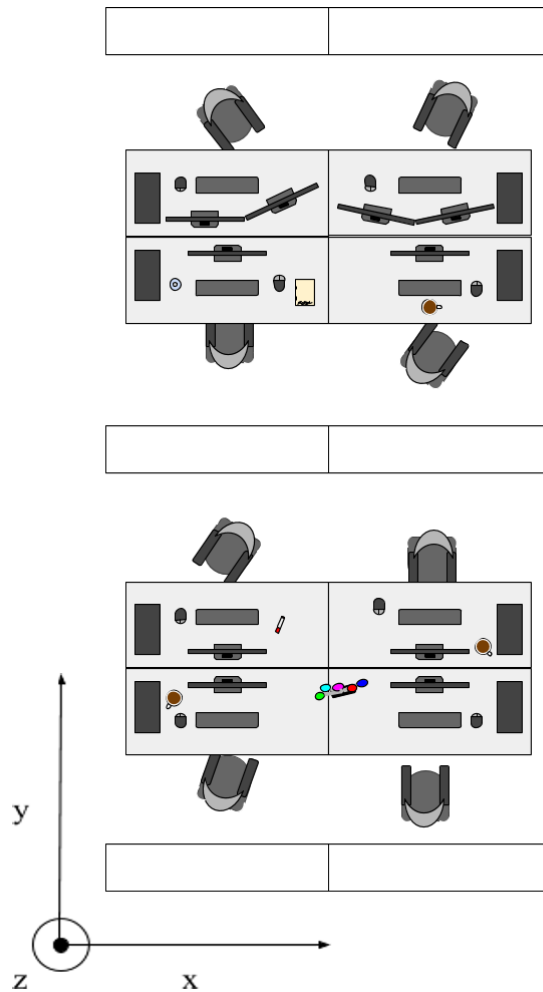


Figure 11: The office in which all tests were conducted. Note the coordinate system, z-axis represents height.

4.1 Performance of the Neural Network

To train a neural network, data needs to be presented to it. The "Urban sound 8K" is a dataset containing 8732 labelled sound excerpts of urban sounds with 10 different classes [25]. One of these classes is gunshots. By gathering all gunshots samples (roughly 300) labelling them 1 and then from the remaining data picking out an similar amount (roughly 400 samples) of sounds not containing gunshots labelling them 0, a balanced training/test set is acquired. A small validation set containing roughly 65 samples of the data (not used in training or test) is also used to evaluate the models. As the data set is not very big, the number of epochs (number of times the model trains on each data point) during training is set to 5 in order to prevent overfitting of data. A model trained with a higher number of epochs performs very well for the training data but lacks accuracy for test and validation data. Training this neural network takes about 15 minutes, but once it has been trained it takes almost no time at all for the model to classify a sound.

Training all three models according to Figure 7 with parameters from Table 2 and letting them classify the validation set gives the results in Table 4. The confusion matrix for the different models can be seen in Table 5, 6, and 7.

Table 4: Performance of the three neural network models from Figure 7 trained in the same way.

Model	test accuracy	validation accuracy
Direct model	0.95	1.0
FC2048	0.97	0.98
Diminution model	0.89	0.98

Table 5: A confusion matrix for "Direct model" on the validation set.

Direct model	predicted gunshot	predicted non-gunshot
actual gunshot	26	0
actual non-gunshot	0	39

Table 6: A confusion matrix for "FC2048" on the validation set.

FC2048	predicted gunshot	predicted non-gunshot
actual gunshot	26	0
actual non-gunshot	1	38

Table 7: A confusion matrix for "Diminution model" on the validation set.

Diminution model	predicted gunshot	predicted non-gunshot
actual gunshot	26	0
actual non-gunshot	1	38

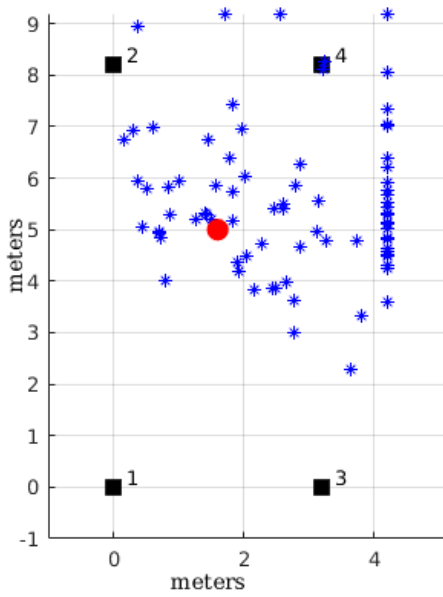
4.2 Method for Estimation of the TDOA

In order to test the different methods for estimation of the TDOA, the digital recorder TASCAM was used with SONY microphones. Since TASCAM does not have any synchronisation problems, no latencies were introduced. The SONY microphones were placed on four shelves in the office, forming a rectangle. In Figure 11 these are the shelves at the top and bottom of the image. The test is conducted in the following way: A person stands in the office area and makes a sharp sound by hitting two spoons against each other during recording. The exact position of the sound source is measured with measuring tape and written down together with the calculated positions. By repeating the experiment 100 times, estimations of the performances are achieved.

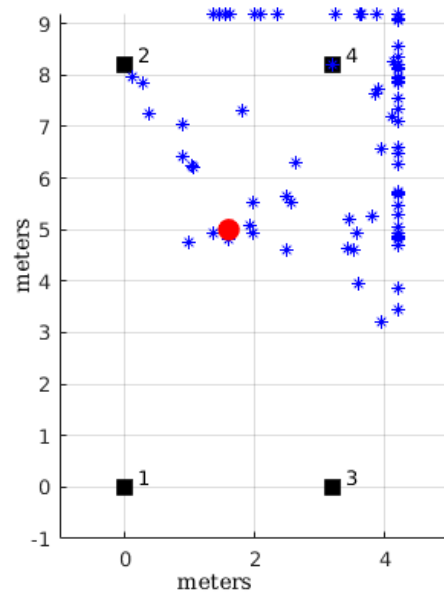
The TDOAs for the recorded data are calculated using the methods from section 2.4. The position for the sound source is calculated with the method from section 2.5 and the mean error for each method can be seen in Table 8. A boundary on every axis is set to the office walls in the optimisation algorithm as the sound is known to be inside a limited area. In Figure 12 the measured position of the sound source is plotted together with the calculated positions from the 100 sounds. The image shows the result for all mentioned methods for estimation of the TDOAs.

Table 8: The mean error for the different methods of calculating the TDOA and the runtime for the algorithms. Note that the mean error are calculated in 2D only. The error in height is not added.

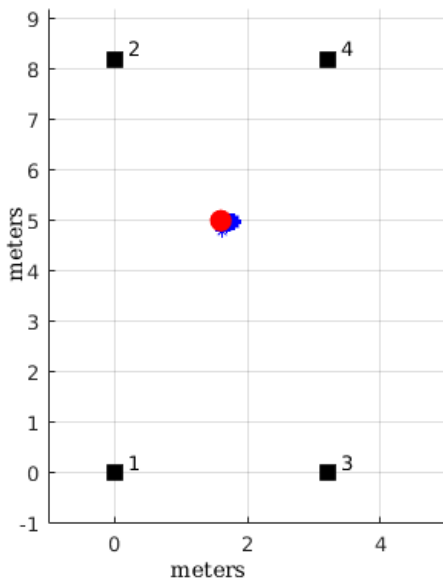
Method	Mean error (m)	Standard deviation (m)	Time (s)
Cross Correlation	2.1	1.08	6.85
GCC-Phat	3.0	1.2	7.35
Flank Detection	0.13	0.04	3.55
AED	1.9	1.45	9640



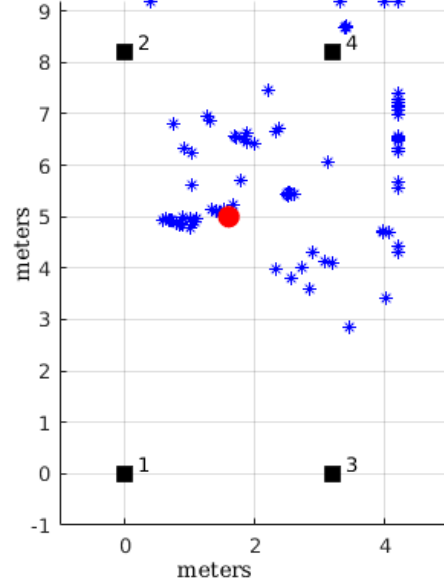
(a)



(b)



(c)



(d)

Figure 12: The plots shows the position of the sound source (the red dot), the microphones (the black boxes) and the calculated positions for the different methods of calculating the TDOA (the blue stars). Figure 12a shows the result using Cross correlation, Figure 12b shows the result using GCC-Phat, Figure 12c shows the result using Flank detection and Figure 12d shows the result using Adaptive Eigenvalue Decomposition.

4.3 Synchronisation of the Speakers

Two methods are tested in order to solve the synchronisation problem. Sections 4.3.1 and 4.3.2 contain detailed description on how these experiments were conducted. Every test is done 100 times in order to acquire an estimation of the performances. Of course, results from recordings with no synchronisation is also presented to show how the system performs without synchronisation.

4.3.1 Synchronisation with AudioInterleave in GStreamer

With software implemented in C code using GStreamer containing the AudioInterleave object, recording is started and the speakers are synchronised. The speakers with built-in microphones are placed on the shelves in Figure 11. One person stands somewhere in the office and hits two spoons against each other. This position is measured with measure tape and written down. The sound files are sent to MATLAB and analysed. The TDOAs are estimated with flank detection and the calculated position is written down. This process is then repeated 100 times.

4.3.2 Synchronisation by Adding Latencies to the Equation System

Recording is started without synchronisation. The speakers are located in the same manner as in section 4.3.1. One person stands in the office and hits two spoons against each other. Again, this position is noted. However, as described in section 2.6, adding the latencies into the equation system in (12) requires several sharp sounds in order to position all of them at once. Therefore, 9 additional spoon hits are done after the first and important one (minimum is two). The sound files are sent to MATLAB and analysed. The TDOAs are estimated with flank detection and the calculated position of the first sound is written down. Worth to note is that the 9 latter spoon hits only purpose was to estimate the latencies, they were never positioned. This process is then repeated 100 times.

4.3.3 Synchronisation Results

A performance summary of the three tests can be seen in Table 9. The mean error from the 100 tests are listed for every synchronisation method.

Table 9: The mean error for the different methods of synchronising the speakers. The mean error are calculated in 2D only. The error in height is not added.

Method	Mean error (m)	Standard deviation (m)
No synchronisation	1.0540	0.6362
AudioInterleave	0.5748	0.3472
Calculate latencies	0.2827	0.1536

In Figure 13, the sound source position is plotted with the 100 calculated positions for each sound on top of the image of the office. This illustrates better where the speakers were located and where the sound was made.

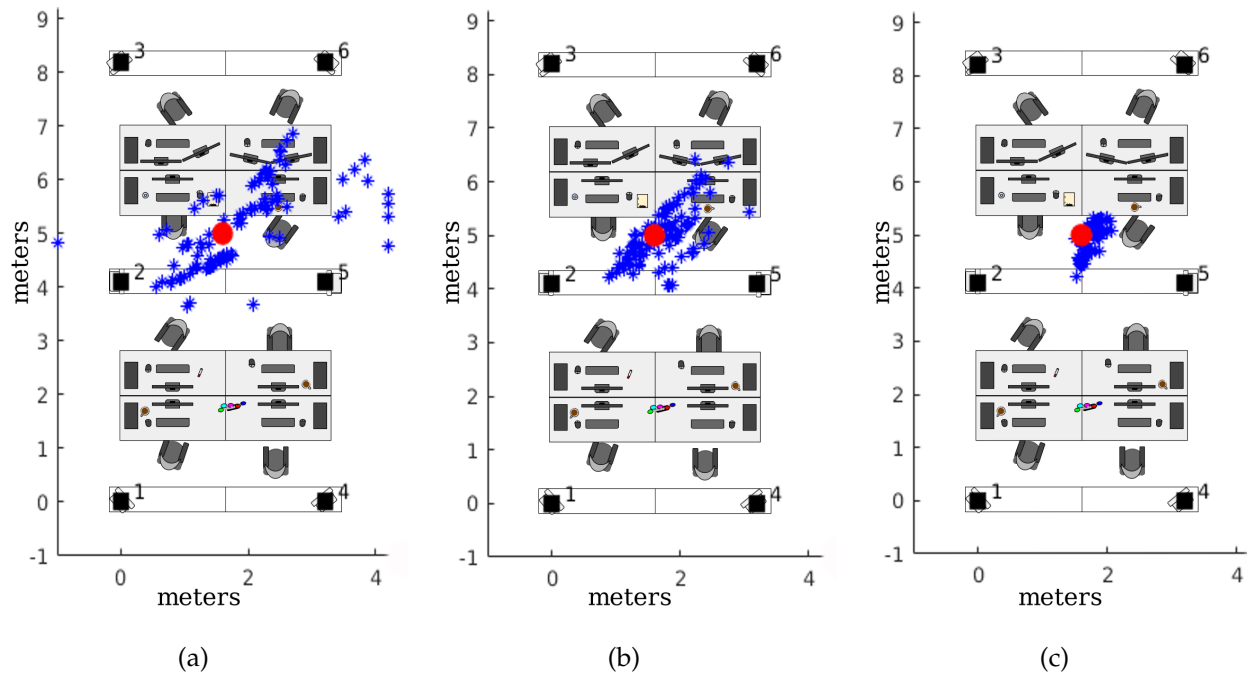


Figure 13: The results are placed on top of the image of the office and it shows the position of the sound source (the red dot), the speakers (the black boxes) and the calculated positions (the blue stars) for every experiment using Flank detection. Figure 13a shows the results from the system without using synchronisation. Figure 13b shows the result using GStreamer with the object "AudioInterleave" and Figure 13c shows the result after calculating the latencies using 10 sounds. Note that the last test actually contains many more sharp sounds (those 9 additional sounds made to calculate the latencies), however, only the first and relevant calculated sound source position is plotted.

4.4 Visualisation of the Performance

The equation system in (12) works best when the sound source is moving, see section 5.4 for further explanations. The last experiment is carried out with only one purpose, to give a visualisation of how well the system can follow a moving sound source which occasionally makes a sharp loud sound. Flank detection is used combined with adding the latencies to the equation system (12).

With the speakers placed on the shelves in Figure 11 in the same manner as in section 4.3, one person walks around the office between the tables and shelves. Two spoons are hit against each other repeatedly throughout the walk. The results projected on the office can be seen in Figure 14.

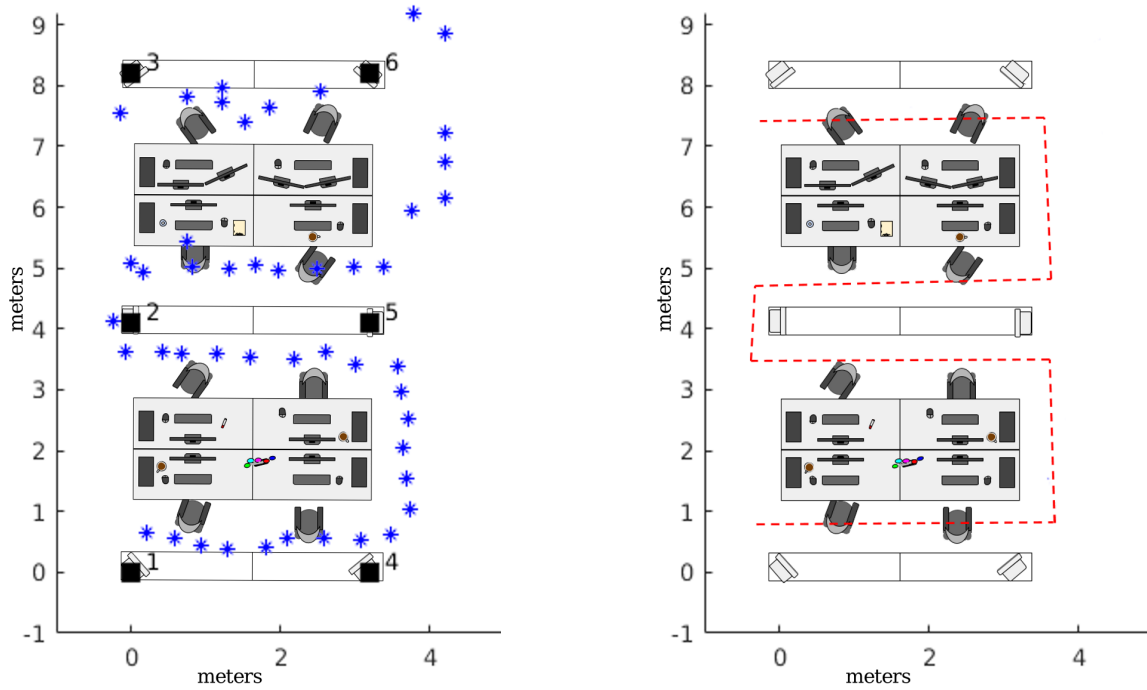


Figure 14: A walk through the office with regular spoon hits. The left image with the blue stars represent the calculated positions by the system and the black numbered boxes are the speakers. On the right image, a red dashed line shows the actual path.

5 Discussion and Conclusions

5.1 The Neural Networks

Looking at Table 4, all models for the classification task seems to perform quite well. However, one has to have in mind that the requirements for this system has a very low tolerance of error when it comes to detection of gunshots. Thus a model with close to 100% accuracy is needed. From that viewpoint, the models might not be adequate.

The best performing model on the validation set is the "Direct model". With the features from the base layers of a pretrained VGG16-network directly connected to the output node, this model has minimum additional weights. The performance is acceptable with test and validation accuracy of 95% and 100% respectively. It is quite surprising that the simplest model has the best accuracy; although, the VGG6-network itself is everything but simple. The base layers consists of 16 convolutional layers making it a deep network with millions of weights and nodes.

"FC2048" and "Diminution model" performed slightly worse on the validation data. Instead of an accuracy of 100%, both models missclassified one instance of the data each and landed on an accuracy of 98%. Table 5, 6, and 7 shows confusion matrices for respective models. Interestingly, all missclassifications were false-positives. The models tend to warn for gunshots with no gunshots presence rather than missing gunshots. The two missclassifications are classification from a running air-condition and an idling car-engine. In our opinion, the false positive errors are preferred as a missed gunshot can be quite devastating for users utilising the system. However, with the few data points tested, it is hard to draw any conclusions. One would need a bigger dataset in order to evaluate the models further (this is discussed in section 5.6, "Future Work").

The reasons for worse performance from more complex models can be due to the fact that they fit their parameters too extensively to the training data leading to less flexibility on new data never seen before. This is also called overfitting.

5.2 Information in z-axis Lost

All results from the positioning experiments only notes the calculated position in x- and y-axis of the coordinate system defined in section 4. Due to the fact that these speakers will be located on the same height in our thesis, the derivatives from (11) of the sound source position on the z-axis are always zero. If however, the speakers were located on different height information about the z-axis could be obtained. Let us motivate this by differentiating err_1 in equation (11) with respect to z_s .

$$\begin{aligned}
 err_1 &= TDOA_{12} - \left(\sqrt{(x_1 - x_s)^2 + (y_1 - y_s)^2 + (z_1 - z_s)^2} - \sqrt{(x_2 + x_s)^2 + (y_2 - y_s)^2 + (z_2 - z_s)^2} \right), \\
 \frac{\partial err_1}{\partial z_s} &= - \left(\frac{-2(z_1 - z_s)}{2\sqrt{(x_1 - x_s)^2 + (y_1 - y_s)^2 + (z_1 - z_s)^2}} - \frac{-2(z_2 - z_s)}{2\sqrt{(x_1 - x_s)^2 + (y_1 - y_s)^2 + (z_2 - z_s)^2}} \right), \\
 \Rightarrow \frac{\partial err_1}{\partial z_s} &= \frac{z_1 - z_2}{\sqrt{(x_1 - x_s)^2 + (y_1 - y_s)^2 + (z_2 - z_s)^2}}.
 \end{aligned} \tag{13}$$

From (13), one can see that $\frac{\partial err_1}{\partial z_s} = 0$ if $z_1 = z_2$ which is the case. This means that the optimisation algorithm Levenberg-Marquardt can not find a better position in z-axis than random guessing.

It is not very devastating to lose information about the sound source position in height as this mea-

sure is not of interest. The final system has a goal to detect and position gunshots. One can imagine that the height in which a person held a gun is irrelevant.

5.3 Choice of Method for Estimation of TDOA

The estimation of the TDOAs has been a major concern throughout this master's thesis. The different methods described in section 2.4 works for different kind of setups and sound sources, however the position of the sound source should not affect the result of the estimated position. Cross correlation and GCC-Phat are well established common methods to estimate TDOAs. However, looking at the results from Table 8, one can conclude that they are not the algorithms of choice for this setup and this can be due to several factors. Firstly, cross correlation and GCC-Phat are methods that performs better with longer sound files the reason being that longer sound files contains more data to correlate. Sharp sounds such as gunshots do not provide many data points to correlate, thus, returning a more inaccurate estimation. Secondly, the two methods are sensitive to echo paths which are typically present in an indoor environment.

It is quite obvious which method performs best. Flank detection has a mean error of 0.13 m; more than 10 times smaller than the other algorithms and it has the smallest standard deviation with 0.04 m. Also, it is most effective looking at the time it takes to locate 100 sharp sound events. When evaluating this method, one has to have in mind that the location of the speakers and sound source are measured with measuring tape manually. Of course, this also introduces an error and it can be argued that this measurement error might be greater than 0.13 m. Although flank detection shows good results for sharp sounds, one has to have in mind that it is a method designed for only this purpose. If continuous sounds were the sounds of interest, the flank detection algorithm would fail miserably as there are no flanks to detect.

Adaptive eigenvalue decomposition is a totally different model from the others. Taking the room reverberation into account in addition to time differences from source to receiver, it performs a pinch better than cross correlation but much worse than flank detection. Another concern with this algorithm is its complexity. With a conditional loop in the code, the running time for this method is thousand times slower than the other methods making it an impractical choice.

In Figure 12 the estimated positions tend to be closer to microphone 4. The reason for this is still unknown but it could be because of noise in that area or reverberation.

5.4 Synchronisation

Different methods of synchronising the hardware were tested and the results can be seen in Table 9. Both of the methods ("AudioInterleave" and "Calculate latencies") made an improvement from the system that was not synchronised. The numbers might be a bit off since there can be some measuring errors for the actual sound source. By just looking at the numbers, "Calculate latencies" seems to be the best method. The mean error is 0.3 m while "AudioInterleave" has a mean error of 0.6 m. "Calculate latencies" also has the smallest standard deviation. The difference in performance between them is not huge but Figure 13 shows that the positions calculated from "Calculate latencies" are all closer to the actual point while the positions from "AudioInterleave" are more spread out over the office. Yet it is still clear from both figures where the sound source came from.

The method "Calculating latencies" might seem better by just looking at the results. But as described in section 2.6, it needs more than one instance of sharp sounds in order to position them all at once

and the sounds needs to be scattered around the room. Otherwise the equation system (12) will be dependent. Using "AudioInterleave" to synchronise the system does not have this requirement and this can position a single sharp sound. The question is if it is reasonable to assume that there will be multiple gunshots during a situation. If not, "AudioInterleave" might be the choice although it performs slightly worse.

Another problem is that during longer recordings, the system can skip a frame containing 480 samples. This means that one of the speakers suddenly gets asynchronised with 10 milliseconds. Converted to distance for the positioning, this problem provides an additional error of around 3 m. It is hard to find the root for this problem and at present, we do not know why this phenomenon occurs. In order to avoid this trouble, one can restart the recording every hour.

All the calculated positions from the three synchronisation methods are spread out diagonally over the actual sound source. If the error only comes from the fact that the system is asynchronised, the calculated dots in Figure 13 should form a shape more like a circle instead. The reason for this aberration can be rooted in the speakers slightly different input gains. The algorithm for flank detection is very vulnerable to such variations as the signal's amplitude information is crucial (see section 2.2). The system is even more sensitive to sounds with some specific frequencies because these can cause standing waves inside the speakers.

5.5 The Final System

After acquiring all results from the experiments, a final system has been chosen. As both synchronisation methods has their own advantages and disadvantages, the final system contains both. It starts recording with "AudioInterleave" regardless and if only a single gunshot is found, latency parameters will not be added to the equation system (11). However, if multiple gunshots are detected, latencies will be added and the equation system transforms to (12). The system works best for a series of several sharp sounds, if two or more sounds occur at the same time the TDOAs can not be calculated since the system can not separate sounds from each other. The calculations will be done as if the sounds are from the same sound source. Detection of sound is done by flank detection and the classification task is sent to the neural network with output layer directly from the base layers of VGG16. To estimate the TDOA, the flank detection algorithm is used. An illustration of the final system can be seen in Figure 15.

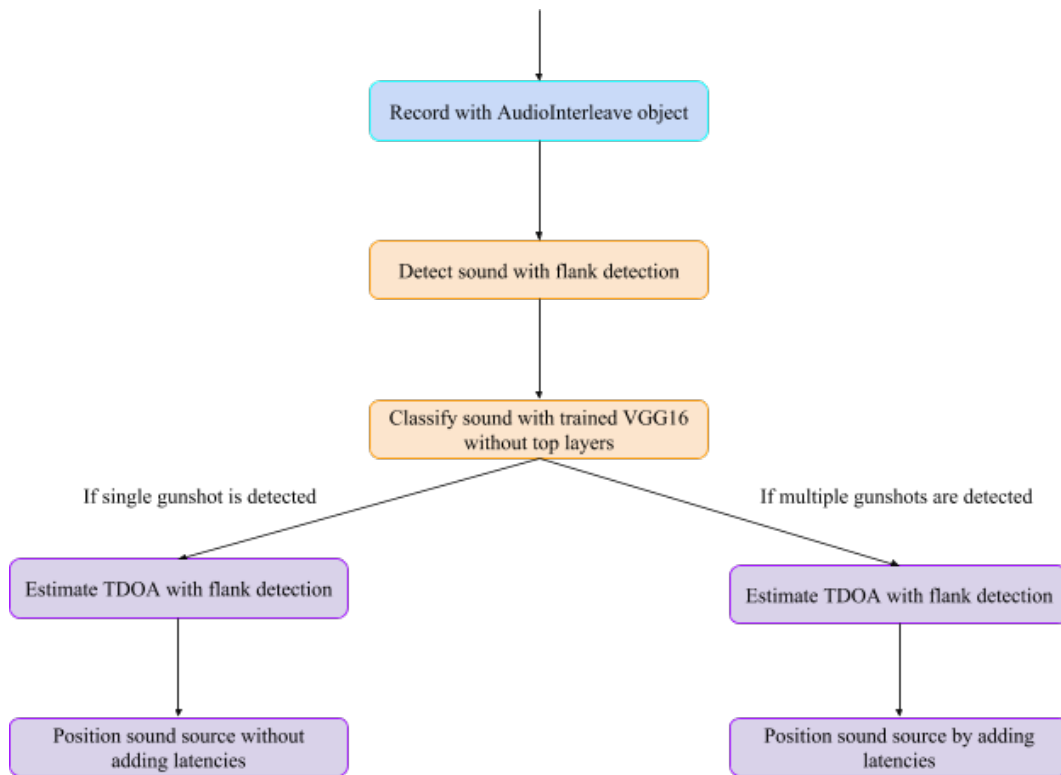


Figure 15: The final system with all chosen methods and algorithms. It is based on the results from all tests done throughout this master's thesis.

5.6 Future Work

In this project, like many other, there were ideas and problems that was never tested or solved due to lack of time and that some difficulties with the hardware were presence.

There are some changes and ideas that could make the system work even better in the future. In this master's thesis, only a pretrained VGG16-network is used. One can think of many different models that might be better up for the task. A critical issue with machine learning models are the data used for training. Generally, the more data presented to a model, the better performance is achieved. The data set "Urban Sounds 8k" contained 300 samples of gunshot data, 10 ms each. For a real system that needs to detect gunshot in sound files, a lot more data should be present.

The microphones inside the speakers did not always detect all sound sources, which became a problem in some applications. There was an idea to let one of the speakers play a tone and then let the rest of the speakers calibrate their signals after the speaker that played the tone. This would synchronise the speakers. It worked well in smaller rooms where all speakers could hear each other, but in larger rooms this method was not possible to test as speakers far from the speaker playing the tone could not hear it. A part of this idea was to let the speaker play a tone that humans can not hear, particularly a tone with frequency above 20 kHz. However, the microphones could not record this tone either. In the end, this synchronisation method was never implemented fully. In the future, when better microphones are added, we believe that this is a method that should be looked into again.

The method that worked best for estimating the TDOA can only position sharp sounds. Continuous sounds does not have a distinct flank for the flank detection algorithm and if the system should be able to position other interesting sounds, such as angry voices or screams, the method to calculate the TDOA needs to be changed.

The system has never been tested with background noise, instead the experiment with 100 sharp sounds took place in an empty and quiet office environment with soft music playing in the background. To fully test the methods for calculating the TDOA and get an estimation of performance, the experiments should be done again but with background noise. This background noise should contain music and people talking to better resemble the environment in which the speakers will be used.

References

Texts

- [2] J. Benesty. “Adaptive eigenvalue decomposition algorithm for passive acoustic source localization”. In: *The Journal Of The Acoustical Society Of America* (1999).
- [3] J. Benesty, Y. Huang, and J. Chen. *Acoustic MIMO Signal Processing (Signals and Communication Technology)*. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2006. ISBN: 3540376305.
- [4] J. Benesty, Y. Huang, and G. W. Elko. “Adaptive eigenvalue decomposition algorithm for real time acoustic source localization system”. In: *ICASSP*. 1999.
- [5] C. M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Berlin, Heidelberg: Springer-Verlag, 2006. ISBN: 0387310738.
- [6] L. Böiers. *Mathematical Methods of Optimization*. Studentlitteratur AB, Lund, 2015.
- [7] B. Van Den Broeck et al. “Time-domain generalized cross correlation phase transform sound source localization for small microphone arrays”. In: *2012 5th European DSP Education and Research Conference (EDERC)*. Sept. 2012, pp. 76–80. DOI: 10.1109/EDERC.2012.6532229.
- [8] A. Jansen D. Ellis S. Hershey and M. Plakal. *tensorflow/models/audioset*. <https://github.com/tensorflow/models/tree/master/research/audioset>. (Visited on 06/11/2018).
- [9] GStreamer documentation. *Basic tutorial 1: Hello world!* <https://gstreamer.freedesktop.org/documentation/tutorials/basic/hello-world.html>. (Visited on 06/11/2018).
- [10] GStreamer documentation. *RTP and RTSP support*. <https://gstreamer.freedesktop.org/documentation/rtp.html>. (Visited on 04/26/2018).
- [11] GStreamer documentation. *What is GStreamer?* <https://gstreamer.freedesktop.org/documentation/application-development/introduction/gstreamer.html>. (Visited on 04/26/2018).
- [12] G. Dreyfus. *Neural networks: methodology and application*. Springer, Berlin, Heidelberg, 2005.
- [13] M. Farzone and K. Smidje. “Embedded sound localization using multilateration in network camera systems”. In: *Lund University* (2013).
- [14] G. Flood, A. Heyden, and K. Åström. “Estimating Uncertainty in Time-difference and Doppler Estimates”. In: *7th International Conference on Pattern Recognition Applications and Methods*. 2018.
- [15] “Gauss–Newton Methods”. In: *From MathWorld—A Wolfram Web Resource* (). URL: <http://reference.wolfram.com/language/tutorial/UnconstrainedOptimizationGaussNewtonMethods.html>.
- [16] F. Gustafsson and F. Gunnarsson. “Positioning using time-difference of arrival measurements”. In: *Acoustics, Speech, and Signal Processing, 2003. Proceedings. (ICASSP '03). 2003 IEEE International Conference on*. Vol. 6. Apr. 2003, VI-553-6 vol.6. DOI: 10.1109/ICASSP.2003.1201741.
- [17] S. Hershey. “CNN Architectures for Large-Scale Audio Classification”. In: (2017).
- [18] D. P. Kingma and J. Ba. “Adam: A Method for Stochastic Optimization”. In: *CoRR abs/1412.6980* (2014).
- [19] C. Knapp and G. Carter. “The generalized correlation method for estimation of time delay”. In: *IEEE Transactions on Acoustics, Speech, and Signal Processing* 24.4 (Aug. 1976), pp. 320–327. ISSN: 0096-3518. DOI: 10.1109/TASSP.1976.1162830.
- [20] K. Krishnan. *The Industrial Information Technology Handbook*. SFWR 4C03: Computer Networks and Computer Security, 2004.

- [21] Y. Kuang and K. Åström. "Stratified Sensor Network Self-Calibration From TDOA Measurements". In: *21st European Signal Processing Conference 2013*. 2013.
- [23] Douglas O'Shaughnessy. *Speech communication: human and machine*. Addison-Wesley, 1987. ISBN: 978-0-201-16520-3.
- [24] O. Russakovsky et al. "ImageNet Large Scale Visual Recognition Challenge". In: *International Journal of Computer Vision (IJCV)* 115.3 (2015), pp. 211–252. DOI: 10.1007/s11263-015-0816-y.
- [25] J. Salamon, C. Jacoby, and J. P. Bello. "A Dataset and Taxonomy for Urban Sound Research". In: (Nov. 2014), pp. 1041–1044.
- [26] SST ShotSpotter. "Gunshot Detection Technology". In: 2014.
- [27] K. Simonyan and A. Zisserman. "Very deep convolutional networks for large-scale image recognition". In: (2014).
- [29] W. Sun and Y. X. Yuan. *Optimization Theory and Methods: Nonlinear Programming*. Springer Optimization and Its Applications. Springer US, 2006. ISBN: 9780387249766. URL: <https://books.google.se/books?id=o0BYHLhhPJMC>.
- [31] E. W.Weisstein. "Discrete Fourier Transform". In: *From MathWorld—A Wolfram Web Resource* (). URL: <http://mathworld.wolfram.com/DiscreteFourierTransform.html>.
- [32] E. W.Weisstein. "Hyperbola". In: *From MathWorld—A Wolfram Web Resource* (). URL: <http://mathworld.wolfram.com/Hyperbola.html>.
- [33] Guanghan Xu et al. "A least-squares approach to blind channel identification". In: *IEEE Transactions on Signal Processing* 43.12 (Dec. 1995), pp. 2982–2993. ISSN: 1053-587X. DOI: 10.1109/78.476442.
- [34] R. Zurawski. *User Datagram Protocol (UDP), Lecture notes*. CRC Press, Florida, 2004.

Images

- [1] *AXIS C1004-E Network Cabinet Speaker*. <https://www.axis.com/sv-se/products/axis-c1004-e>. (Visited on 04/26/2018).
- [22] *Netgear GS108PE 8-Port Gigabit Switch - 4 PoE Ports*. <https://www.ipphone-warehouse.com/Netgear-GS108PE-p/gs108pe-300nas.htm>. (Visited on 04/26/2018).
- [28] *Sony ECM-VG1 Electret Condenser Shotgun Microphone*. https://www.bhphotovideo.com/c/product/758139-REG/Sony_ECM_VG1_ECM_VG1_Electret_Condenser_Microphone.html. (Visited on 05/14/2018).
- [30] *Tascam DR680 8-track Portable Digital Field Audio Recorder*. <https://www.amazon.com/Tascam-8-track-Portable-Digital-Recorder/dp/B0036VC3I2>. (Visited on 05/14/2018).

Master's Theses in Mathematical Sciences 2018:E30

ISSN 1404-6342

LUTFMA-3350-2018

Mathematics

Centre for Mathematical Sciences

Lund University

Box 118, SE-221 00 Lund, Sweden

<http://www.maths.lth.se/>