Hear-and-avoid for UAVs

using convolutional neural networks

Dirk Wijnker

TU Delft

Technische Universiteit Delft

Cover: https://pxhere.com/

Hear-and-avoid for UAVs

using convolutional neural networks

by

Dirk Wijnker

to obtain the degree of Master of Science
at the Delft University of Technology,
to be defended on Tuesday November 6th, 2018 at 9:30 AM.

Student number:	4280598	
Project duration:	November, 2017 – November, 2018	
Thesis committee:	Dr. Ir. G.C.H.E. de Croon,	TU Delft, supervisor
	Ir. C. de Wagter,	TU Delft, supervisor
	Ir. T. van Dijk,	TU Delft, supervisor
	Dr. Ir. M. Snellen,	TU Delft, external member

An electronic version of this thesis is available at `http://repository.tudelft.nl/`.

TU Delft

Contents

List of Figures

List of Tables

List of Abbreviations

AD Amplitude descriptor

ANCE Aircraft Noise and Climate Effects

ANN Artificial Neural Network

ASR Automatic Speech Recognition

AUC Area Under the Curve

AVS Acoustic Vector Sensor

AW MPEG-7 audio waveform

CELP Code Exited Linear Prediction

CNN Convolutional Neural Network

ConvNet Convolutional Neural Network

CRNN Convolutional Recurrent Neural Network

DAQ Data Acquisition Box

DBN Deep Belief Network

DCASE Detection and Classification of Acoustic Scenes and Events

DCT Discrete Cosine Transform

DFT Discrete Fourier Transform

DNN Deep Neural Network

DOA Direction of Arrival

DTW Dynamic Time Warping

EM Expectation-Maximization

EPA Environment Protection Authority

FFT Fast Fourier Transform

FN False Negative

FP False Positive

FPR False Positive Rate

FT Fourier Transform

GA General Aviation

GCC Generalized Cross Correlation

GMM Gaussian Mixture Models

HMM Hidden Markov Model

HRTF Head Related Transfer Function

kNN k-Nearest Neighbours

LAT Log Attack Time

LBP Local Binary Pattern

LPCC Linear Prediction Cepstrum Coefficients

LSP Linear Spectral Pair

LSTM Long Short-Term Memory

MFCC Mel Frequency Cepstral Coefficients

MIR Music Information Retrieval

MUSIC Multiple Signal Classification

PSD Power Spectral Density

ReLU Rectified Linear Unit

RNN Recurrent Neural Network

ROC Receiver Operating Characteristic

RPM Revolutions Per Minute

SAI Stabilized Auditory Image

SC Spectral Centroid

SER Sound Event Recognition

SF Spectral Flux

SGD Stochastic Gradient Descent

SIF Spectrogram Image Features

SNR Signal-to-Noise Ratio

SPL Sound Pressure Level

SSF Subband Spectral Flux

STE Short-Time Energy

STFT Short-Time Fourier Transform

SVM Support Vector Machines

TDOA Time Difference of Arrival

TP True Positive

TPR True positive rate

UAS Unmanned Aircraft System

ZCR Zero-Crossing Rate

I

Scientific Paper

Hear-and-avoid for UAVs using convolutional neural networks

Dirk Wijnker, student, Tom van Dijk, daily supervisor, Guido de Croon, supervisor, Christophe de Wagter, supervisor

Abstract—We investigate how an Unmanned Air Vehicle (UAV) can detect manned aircraft with a single microphone. In particular, we create an audio data set in which UAV ego-sound and recorded aircraft sound can be mixed together, and apply convolutional neural networks to the task of air traffic detection. Due to restrictions on flying UAVs close to aircraft, the data set has to be artificially produced, so the UAV sound is captured separately from the aircraft sound. The aircraft data set is collected at Lelystad airport by capturing flyovers with a microphone array. It is mixed with UAV recordings, during which labels are given indicating whether the mixed recording contains aircraft audio or not. The mixed recordings are the input for a model that determines whether an aircraft is present or not. The model is a CNN which uses the features MFCC, spectrogram or Mel spectrogram as input. For each feature the effect of UAV/aircraft amplitude ratio, the type of labeling, the window length and the addition of third party aircraft sound database recordings is explored. The results show that the best performance is achieved using the Mel spectrogram feature. The performance increases when the UAV/aircraft amplitude ratio is decreased, when the time window is increased or when the data set is extended with aircraft audio recordings from a third party sound database. It is not desirable to train the model on distant approaches and test them on nearby approaches as the performance then drops. The results also prove that the performance increases the closer the aircraft is. Although the currently presented approach has a number of false positives and false negatives, that is still too high for real-world application, this study indicates multiple paths forward that can lead to an interesting performance. In addition, the data set is provided as open access, allowing the community to contribute to the improvement of the detection task.

Index Terms—Hear-and-avoid, Convolutional Neural Network, MFCC, Spectrogram, Mel, UAV, TU Delft.

1 Introduction

More and more UAVs are entering the air every day, both for professional as well as for recreational purposes. Safety and regulations are subjects undergoing intense study nowadays in the UAV industry, as UAVs form a hazard for people, other (air) traffic, buildings, etc. For this research, the focus is on the collisions between UAV and air traffic, which are still possible to occur. For example, emergency helicopters sometimes fly low in UAV-permitted airspace. Part of this problem can be solved by establishing (and following) good rules and laws, but also technology can help out. Technology becomes even more important when UAVs have to operate fully autonomously, as required by many future applications. A project initiated by Single European Sky ATM Research (SESAR) that aims to increase air traffic safety regarding to UAVs is called Percevite[1]. Using multiple lightweight, energy-efficient sensors obstacles should be avoided to protect UAVs and their environment. One such a sensor is a microphone, which fulfills the task of 'hear-and-avoid', meaning that it should detect and avoid air traffic by sound. The goal of this research is to create a safer airspace by creating this hear-and-avoid algorithm.

The first feasibility study for hear-and-avoid has been performed by Tijs et al [1]. In this research an acoustic vector sensor is used to detect other flying sound sources. Two co-authors, De Bree and De Croon [2], have used an acoustic vector sensor in order to detect sound recorded on a UAV for military purposes. However, neither works have used deep artificial neural networks to separate aircraft and UAV sounds. Moreover, there are two research groups that have tried to identify the position of other UAVs using sound recorded from a UAV. Basiri et al. [3], [4], [5], [6] try to determine the position of a UAV in a swarm of UAVs. The transmitting UAV sends a chirp sound in the air that has frequencies different than the UAVs ego-sound, which can be picked up quite well while flying. Also, they do tests with engines of the receiving UAV turned off and the transmitting UAVs not transmitting the chirp anymore. Also here, based on the engine sounds of the transmitting UAV its location can be determined. The hear-and-avoid algorithm can be seen as a follow up of these researches, as they have not managed to identify other air traffic by its original sound while also having the engines turned on. Harvey and O'Young [7] show that with two microphones, the detection of another UAV can be performed at such a distance that is double the distance to prevent head-on collision. Furthermore, research is performed focusing only on the UAV sound by Marmaroli et al. [8]. They have created an algorithm that is able to denoise the ego-sound of the UAV based on the knowledge about the propellers' revolutions per minute (RPM).

One of the reasons that there is not a large amount of researches performed on audio analysis for UAVs is that there are alternatives that provide traffic information, such as ADS-B, GPS, vision, etc. However, all alternatives have their disadvantages and do not fully eliminate the chance of a collision. For example, ADS-B requires a system in an aircraft that is not always present or turned on. For vision based sense-and-avoid its images can be disturbed due to speed, rain, fog, darkness, objects, etc. Sound, on the other hand, is inevitable for motorized aircraft, so it is a

1. www.percevite.org

promising method. Moreover, microphones are lightweight, easy to use, omnidirectional and only weakly influenced by weather. The challenge that sound brings in this application is that many different (loud) sounds are present, such as the UAV's ego-noise, wind, air traffic and environmental sounds.

In this research the following situation is studied: a UAV, which is carrying a single microphone, flies around and should detect incoming or passing aircraft based on sound, after which it should perform a avoidance maneuver. The avoidance maneuver is not elaborated on in this research, it is assumed that the UAV would either warn the human operator (if there is one), or autonomously descend or even land if it detects an aircraft. The detection of aircraft will be realized by means of a convolutional neural network (CNN) due to their promising performance on sound in [9], [10] and [11]. The representative data set that is needed, which consists of audio recordings taken on a UAV including aircraft sound, does not exist yet and therefore needs to be artificially created. The CNN uses three audio features as input: Mel Frequency Cepstral Coefficients (MFCCs), spectrograms and Mel spectrograms. Four variables are changed in the data sets to discover their influence: the window length, the amplitude ratio UAV/aircraft, the type of labeling and the use of third party database recordings.

The remainder of the article is structured as follows. The generation of the data set is explained in section 2, including how the individual sound recordings are obtained, how those are processed and mixed to recordings that include both UAV and aircraft sound. Secondly, the features and the model are described in section 3. The results for each of the models are shown in section 4 and discussed in section 5.

2 Audio Acquisition

This research needs a database that contains audio recordings, recorded on UAVs, of the UAV's ego-sound and closely approaching aircraft. Such a database does not exist yet and therefore it is created for this purpose. The database consists of (preprocessed) sound recordings (of UAVs, aircraft and rotorcraft) and labels, which indicate whether only UAV sound is present or UAV and aircraft sound are present. The data set is provided as open access.

2.1 Sound recordings

The laws on UAVs prevent the UAV to come in the vicinity of an aircraft. It is therefore - under normal circumstances - legally not possible to make recordings obtained from a UAV including aircraft sounds. In order to still have a representative database of UAV sounds that include passing aircraft, the UAV sounds and aircraft sounds are recorded separately and mixed afterwards. Three types of recordings have been used: self-made recordings using a microphone on a UAV, general aviation aircraft recordings using a microphone array and aircraft recordings obtained from a third party audio database.

2.1.1 Recordings of the UAV sounds

The UAV sounds are recorded in the Cyberzoo of the TU Delft. This is a protected area for UAVs to be safely and legally flown at the university. An 808 micro camera[2] is placed under a Parrot Bebop UAV, so that its body already blocks part of the UAV's ego-sound. Between the UAV and the microphone, foam is used to absorb the mechanical vibrations. During the recordings, the UAV performed all the possible rotations and movements around its pitch, roll and yaw axes at different speeds. In this manner the most common frequency and sound pressure levels are included in the recordings. After recording, the data is cropped to remove the silences in the beginning and at the end. These recordings are complemented with audio recordings from a mobile phone that filmed the UAV from a close distance. Effectively a total of 20 minutes of UAV recordings are used.

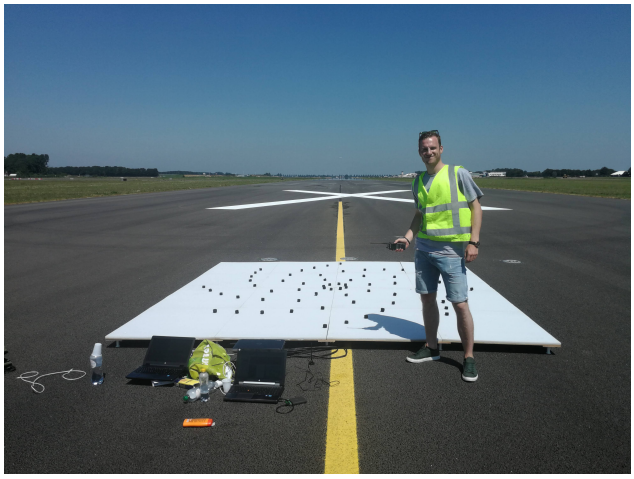


Fig. 1: The acoustic camera on the runway of Lelystad Airport.

2.1.2 Recordings of general aviation flights

Since the most probable group to come in contact with UAVs is general aviation (GA) rotor- and aircraft, flyover data has been obtained at the biggest GA airfield of the Netherlands, Lelystad Airport, in collaboration with the Aircraft Noise and Climate Effects (ANCE) section of the TU Delft.

As Lelystad airport is expanding to a larger airfield, the runway is extended, but the new part is not in use yet. This part of the runway is therefore a perfect place to obtain recordings as the aircraft would fly straight over the so-called "acoustic camera".

The acoustic camera, designed and built by the TU Delft [12], consists of an array with 8 bundles of 8 microphones[3]. The bundles are arranged in a spiral shape for optimal beamforming purposes. The microphones are covered in a foam layer to decrease the noise due to wind. Moreover, the array is covered in foam in order to absorb ground reflections. All the bundles are connected to a Data Acquisition Box (DAQ) which samples the data at 50 kHz and sends it to the connected computer. Not only the DAQ is connected to the computer, but also an ADS-B receiver in order to receive aircraft position information. However, the ADS-B did not produce useful information as none of the GA aircraft broadcast ADS-B information. Moreover, a

2. http://www.chucklohr.com/808/
3. Model: PUI AUDIO 665-POM-2735P-R

mobile phone camera is placed in the center of the array to capture the flyover on video, but this data is not used for this research. The setup of the acoustic camera is shown in Figure 1.

In total 75 recordings are obtained, which consist of background noise recordings and flyovers. One recording sometimes consists of more than one flyover. Effectively, 75 GA aircraft and 9 helicopter flyovers are captured. The background noise consists of microphone noise, noise due to wind, distant traffic and a distant motor race track.

For this research only the recording of one microphone is necessary, so from only one microphone the recordings are extracted. Every microphone is checked to make sure it worked correctly. One of the 64 microphones is faulty, so its data is not used.

2.1.3 Recordings obtained from a third party audio database

With regard to creating a data set that is representative for the possible air traffic sounds that a UAV could encounter, it had to consist of more than only flyover data. For example, other background noise could influence the detection performance. Therefore also a (free) audio database[4] is consulted to obtain helicopter and (propeller) aircraft sounds. Only the sound samples that are of sufficient quality and which are not mixed with (too much) other background noise are selected.

2.2 Data preprocessing

All the separate recordings are manually modified before adding them together. Some UAV recordings contained heavy vibrations of the tape that held the microphone. Those recordings are removed from the data set. For both the UAV recordings and the third party database recordings the silent/fading start and end are cut out. The recordings obtained at Lelystad airport do not require this as the parts that do not include aircraft sound are used as background noise. Instead, we manually labelled every second in the recording, indicating whether it consists of only background noise or include aircraft sound. The recordings from Lelystad Airport include noise introduced by the microphones and the wind. A first order Butterworth low-pass filter is used to remove most of the noise. Most of the time the aircraft sound information is in the frequency region lower than 100 Hz. Only during a flyover aircraft sound information comes above this value. In order to capture the higher frequency content during a flyover but also remove much of the noise during the rest of the time, the cut-off frequency is set on 2.5 kHz.

All the recordings are resampled to a sample rate of 8 kHz as there is no important information present above the Nyquist frequency of 4 kHz and it decreases the size of the data set significantly, which shortens the computational time. Secondly, the sound recordings are normalized by scaling the amplitude between -1 and 1, so that the amplitude of two recordings is similar. Before mixing aircraft and UAV sounds, also data augmentation is applied to all the separate aircraft and UAV recordings in order to increase the size of the data set, which increases the performance of the model. Three types of data augmentation are applied: addition of white noise, increase in pitch and decrease in pitch. The white noise is a randomly generated Gaussian distribution with mean 0 and a variance of 0.005. The pitch is increased and decreased by two semitones on the 12-tone. An increase of two semitones relates to $\sqrt[12]{2^2} \approx 1.12$ times the original frequency. After augmentation, the data set is four times its original size, one original data set plus three augmented data sets.

2.3 Mixing the recordings

In order to get sound samples that include both aircraft and UAV sound, the following mixing procedure is used.

First, the whole data set is split up in a test set and in a training set. All the augmented versions of a sound sample are always in the same set as their original sound sample to ensure that the two sets are uncorrelated.

Secondly, each recording from Lelystad airport is combined with a randomly selected UAV recording of the same set. In some (part of the) recordings only background noise is present. This background noise is necessary since without the noise, the model might classify every sound which is not UAV sound as aircraft sound. Mixing consists of adding a segment of the Lelystad airport sound sample, which has a random length, to one of the UAV recordings on a random starting position. If the starting position plus the length of the segment is longer than the length of the UAV sound sample, the added segment is cut off at the end of the UAV sound sample. The mixed sample therefore never exists of only aircraft sound. The total length of each mixed sample is equal to the length of the UAV recording, which is different for each recording.

Mixing the third party database recordings is done slightly different than the method described for the Lelystad recordings because the third party database recordings always exist fully of aircraft sound. The difference between the two mixing methods is that not only a part of the recording is added to the UAV sound sample, but the whole recording is added instead (at a random starting position).

The detection model in this paper requires the inputs to be of equal length (more on this in subsection 3.2). As this is not the case for the combined samples, the third step is to cut the combined samples to equal lengths. To maximize the amount of data in the sets, the cutting length is set on 51 seconds, which is equal to the length of the shortest combined sound sample.

The amplitude ratio when mixing the UAV and aircraft sound is not always 1:1. In this work, four UAV/aircraft amplitude ratios will be used, namely 0:1 (which means no UAV sound), 1:1 (equal amplitudes), 1:4 (aircraft sound amplitude is four times larger) and 1:8 (aircraft sound amplitude is eight times larger). Most of the time, a ratio of 1:4 is used. This ratio is obtained as follows. Assuming the average Sound Pressure Level (SPL) of a UAV at one meter distance is 76 dB[5] and that of an aircraft at 300 meters distance is 88 dB[6], the difference between the SPLs of the two sounds is 12 dB. Equation 1 shows how the SPL is

4. https://freesound.org/

5. https://www.youtube.com/watch?v=uprXhH6-FNI

6. http://airportnoiselaw.org/dblevels.html

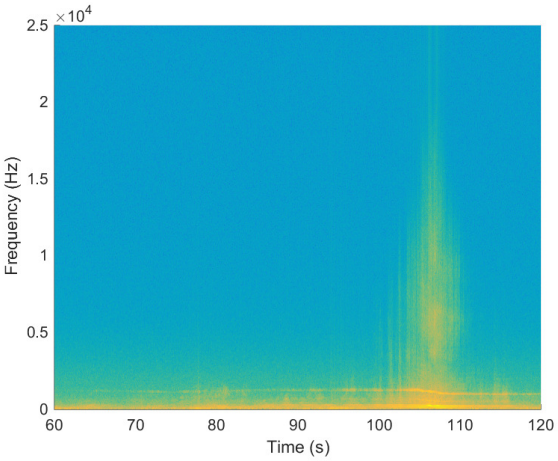


Fig. 2: Spectrogram of a flyover recording. The exact flyover is between 100 and 110 seconds, which can be recognized by a yellow peak and a Doppler shift around 100 Hz. Also before and after the peak the aircraft sound is present, which is visible by the horizontal line around 100 Hz.

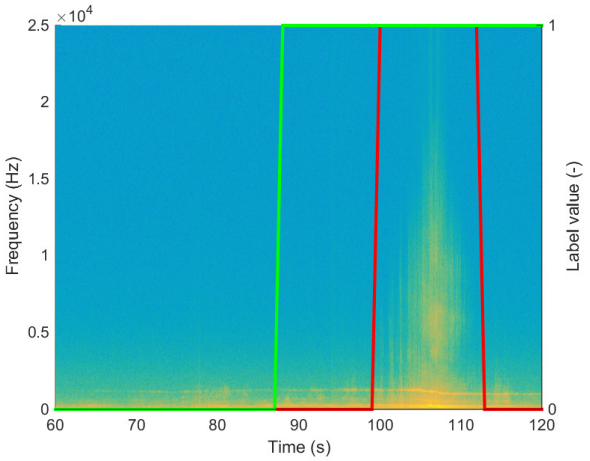


Fig. 3: Spectrogram showing nearby detection labeling (red) and distant detection labeling (green).

calculated from the pressure p_1 (which is the amplitude in the waveform) of a sound and a reference pressure p_0. Taking the amplitude of the UAV waveform as reference pressure and the aircraft waveform as p_1, an SPL of 12 is obtained when the aircraft waveform is 4 times larger. If the ratio 1:4 is corresponding to an airplane on 300 meters distance, 1:1 corresponds to a distance of 1200 meters and 1:8 to a distance of 150 meters, following Equation 2. In this equation, r_2 is the distance of interest, r_1 the original distance, SPL_1 the SPL at r_1 and SPL_2 the SPL at r_2.

$$SPL = 20 \log \frac{p_1}{p_0} \tag{1}$$

$$r_2 = r_1 \cdot 10^{\frac{|SPL_1 - SPL_2|}{20}} \tag{2}$$

2.4 Labels

Each second of a mixed sample is given a binary label, indicating whether there is other aircraft sound present (1) or not (0). The recordings from Lelystad airport are labeled manually before mixing. There are two types of labeling, called nearby detection labeling and distant detection labeling. Nearby detection labeling is partly based on listening to the sound, and partly on looking at the spectrogram. The spectrogram, which is shown in Figure 2 and elaborated on in subsubsection 3.1.2, shows the amount of frequency content over time. Nearby detection labeling gives label 1 when a peak is visible in the spectrogram. By ear this is noticeable as more high frequency content is heard.

Distant detection labeling is purely based on hearing. The frames in which a human is able to separate noise from aircraft sounds are labeled 1. This time it cannot be based on the spectrogram as the aircraft sound is either not visible on the spectrogram (when it is blended in too much with the background noise) or it is visible (as a line on a single frequency caused by the propeller's rotational speed) but the background noise is louder than the aircraft sound. An example of the latter is shown in Figure 3, at which the horizontal line around 100 Hz is also present when no label is given.

The time instances that are not labeled one are labeled zero, so also the background noise from the Lelystad recordings is given the same label as when there is no other aircraft sound present. In Figure 3, the areas in the spectrogram that are labeled as 1 are indicated in red for nearby detection labeling and green for distant detection labeling.

For the third party sound database, the whole aircraft recording is always labeled as a one, as each of the sound samples is selected on only having aircraft sounds. Again, all the time instances in the mixed recording that are not one are labeled zero.

3 Aircraft Audio Event Recognition

The aircraft sound will be detected by a framework that exists of a feature extractor and a classifier. The features capture important sound information and reduce the dimensionality of the data. They are the inputs for the classifier. Thereafter the classifier determines whether the sound sample contains aircraft sound or not.

3.1 Feature extraction

Three features are extracted from the combined sound samples using Python library Librosa [13]. First there are the Mel Frequency Cepstral Coefficients (MFCCs) [14], which are chosen because of their popularity in one of the biggest domains in machine hearing, Automatic Speech Recognition (ASR). The two other features, the spectrogram and Mel spectrogram, are visual representations of the sound samples. Content-based analysis of images is already quite developed [15], therefore the image of a sound might be a good starting point.

For every feature, each frame in the time dimension has a length of one second. One second is a rather large frame but it chosen to reduce in dimensionality. The window moves over the sound sample with a step of one second. All the sound samples are 51 seconds long, thus from each

sound sample 51 separate frames are obtained in the time dimension.

3.1.1 MFCC

The cepstrum is a domain which represents the rate of change in multiple frequency bands. MFCCs are the coefficients of which the cepstrum is composed. It has the ability to separate convoluted signals in the time domain[7]. This domain is therefore often used in speech recognition, to separate the vocal pitch and the vocal tract. The coefficients are obtained by taking the logarithm of the amplitude spectrum, converting this to the Mel scale and taking the Discrete Cosine Transform (DCT). The Mel scale, which is expressed as a function of frequency (f) in Equation 3, is a scale that approximates the human perception of frequency. This scale emphasizes the low frequencies (<1 kHz), which is also the frequency range in which most of the UAV/aircraft sound information is present. The full transformation from time domain signal to MFCC is shown in Equation 4 [16].

$$M(f) = 2595 \log\left(1 + \frac{f}{700}\right) \tag{3}$$

$$MFCC(d) = \sum_{k=1}^{K} (\log X_k) \cos\left[d\left(k - \frac{1}{2}\right)\frac{\pi}{k}\right] \quad \text{for} \quad d = 0, 1, ..., D \tag{4}$$

In this equation X_k is the Discrete Fourier Transform (DFT) obtained in Equation 5 of which the frequency belonging to each k is warped to the Mel scale by Equation 3. D is the total number of coefficients and N the number of data point in the time frame. The number of coefficients used in this research is 20.

$$X_k = \sum_{n=0}^{N-1} X_n e^{-\frac{2\pi i}{N} kn} \quad \text{for} \quad k = 1, 2, ..., N \tag{5}$$

3.1.2 Spectrogram

Spectrograms are visual representations of the energy per frequency plotted against time, of which the Mel spectrogram uses the Mel scale of Equation 3 on the frequency axis. A typical flyover spectrogram (without UAV sound), is shown in Figure 2. In this figure the point where the aircraft is passing the array is between 100 and 110 seconds, which is visible with the large yellow peak and a Doppler shift (the sigmoid-shaped line around 1 kHz). It also shows that when the aircraft is further away, it lacks in high frequency content (due to atmospheric attenuation). That means most of the time only the aircraft's low frequency content is heard by the UAV in combination with low frequency noise.

The spectrograms are calculated following Equation 6, which is the magnitude to the power p of the Short-Time Fourier Transform (STFT). Usually the Power Spectral Density (PSD) is chosen, for which $p = 2$. It uses a window function $w[n]$, in this case the Hann window of one second, of which m is the index of the position in the window function with length N, discrete frequency k, signal $x[n]$ at time n.

$$Spectrogram = \left| \sum_{n=-\infty}^{\infty} x[n] w[n-m] e^{\frac{-i2\pi kn}{N}} \right|^p \tag{6}$$

7. http://research.cs.tamu.edu/prism/lectures/sp/l9.pdf

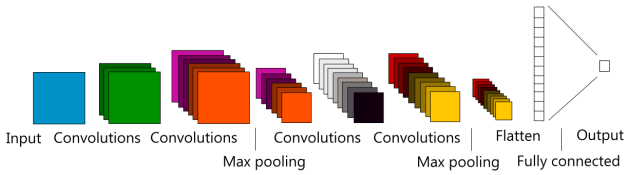


Fig. 4: Architecture of the CNN. The input is a moving time window over the spectrogram, Mel spectrogram or MFCC. The output a binary value indicating whether aircraft sound is present or not.

TABLE 1: Model parameters of the CNN from Figure 4.

Parameter	CNN
Convolution units first set	32
Convolution units second set	64
Kernel size	3x3
Pooling size	2x2
Dropout probability 1	0.25
Dropout probability 2	0.5

3.2 Model

The previously described features are the input for a deep artificial neural network: the convolutional neural network (CNN). It has shown best performance for sound event recognition tasks in [9], [10] and [11]. The basic CNN used in this research is shown in Figure 4. The network is created with the Python libraries Keras [17] and Tensorflow [18].

Even though the features consist of 51 second of UAV/aircraft sound, the input for the CNN is a smaller time window which slides over the time axis. The smaller time window is used as otherwise the detection output of a frame could be depended on data from later frames, due to the fully connected layer. Multiple window lengths are used, as shown in section 4. In the basis, however, the window size is three seconds. This window slides over the feature's time axis with a step of one second.

The first layers of the CNN are convolutional layers. There are two subsequent sets of layers, each consisting of two convolutional layers, followed by a max pooling layer. The convolutional layers use the Rectified Linear Unit (ReLU) as activation function and it applies zero padding to the input. After the two sets, the output is flattened in order to be able to connect it with the output layer, a fully connected layer. For the output, a sigmoid activation function is used, which scales the output (as a float) between 0 and 1. The binary discrimination threshold determines whether this output becomes a 1 or a 0, so whether an aircraft is present or not, respectively. The network is based on [11] and its parameters are modified based on preliminary test results.

Training the network is performed by means of a binary cross-entropy loss function and the Adam optimizer [19].

TABLE 2: Overview of the variables that are changed for each run, including their corresponding values and the values of the standard case, the basis run.

Variables	Basis values	Variations
UAV/Aircraft ratio	1:4	0:1 1:1 1:8
Third party database used	No	Yes
Labeling type	Nearby detection labeling	Distant detection labeling
Window length (s)	3	10 15 20

TABLE 3: The number of each run with their corresponding changed variable and the corresponding value.

Run #	Variation
1	UAV/Aircraft ratio: 0:1
2	UAV/Aircraft ratio: 1:1
3	Basis run
4	UAV/Aircraft ratio: 1:8
5	Database used: Yes
6	Distant detection labeling
7	Window length: 10
8	Window length: 15
9	Window length: 20

The Adam optimizer parameters are the same as in the original paper, so a learning rate of 0.001, $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 10^{-8}$, and no decay. After each pooling layer, dropout is used in order to prevent overfitting of the training data. The parameters for the CNN are shown in Table 1.

4 Results

Each feature is combined with the CNN, so in total three models are tested. They are trained and tested on multiple data sets, which are listed in Table 2. To check the influence of certain parameters in the data set or in the model, four parameters are altered during the runs: the window length, the labeling type, the ratio in amplitude between the UAV and aircraft sound and whether third party database recordings and Lelystad airport recordings are used or only the Lelystad airport recordings.

There is one basis run, for which the window length is 3 seconds, the labeling is nearby detection labeling, the UAV/aircraft ratio is 1:4 and there are no third party database recordings involved. For all the other runs, only one variable of the basis run is changed each time.

The window length is either 3, 10, 15 or 20 seconds. The Lelystad airport recordings are labeled manually, in two manners, as explained in subsection 2.4. For distant detection labeling the training is performed with distant detection labeling and the testing is performed with nearby detection labeling. The idea behind this method is that the model could learn aircraft sound when it is not so obviously present, so that detection when the aircraft is obviously present is outstanding. The amplitude ratio between the UAV and the aircraft is tested when no UAV sound is present, and for the ratios 1:1, 1:4 and 1:8. Lastly, the third party database sounds are either added to the data set or omitted.

From here on, each specific run is indicated by the number of the run given in Table 3. The performance of the models is compared for each of the variables (window length, label type, etc.). This comparison is based on the Receiver Operating Characteristic (ROC) curve. The ROC curve shows the True Positive Rate (TPR) against the False Positive Rate (FPR) for all possible binary discrimination thresholds. The area under the curve (AUC) is a measure of accuracy of the binary classifier. In this research specifically, especially the region of low FPR is important, as it shows how many times the UAV would falsely decide to warn the operator or descend. For each point on the ROC curve the desirable discrimination threshold can be extracted, which determines whether the output from the model is classified with label 1 or label 0.

4.1 Influence of the UAV/aircraft ratio

Runs 1, 2, 3 and 4 are simultaneously plotted for the CNNs in Figure 5. In general, the best performance is achieved for the cases where there is no UAV sound present (run 1). If the UAV's ego-sound is added to the aircraft sound with an amplitude ratio of 1:1 (run 2), the performance is the worst in all cases. The figures show that amplifying the aircraft sound increases performance, however, there is little increase between the ratio 1:4 and 1:8. The expected result is that the less UAV content is present, the more the performance would converge to the result of run 1. Only for the MFCC and Mel spectrogram this trend is visible in the lower FPR region. Looking at the AUC, the MFCC and the spectrogram show no convergence to the ratio of 0:1. In the case of the Mel spectrogram, there is only a difference visible between the ratio of 1:1 and the others.

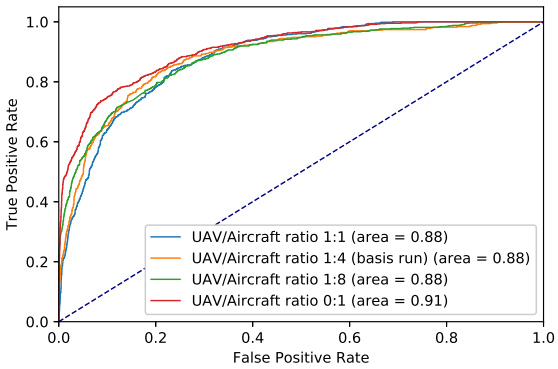
4.2 Influence of the third party database recordings

In the basis run, only the recordings from Lelystad airport are used. This means that all the recordings have (fairly) the same background noise and types of airplanes and they use the same recording equipment. In order to check how much the models rely on these characteristics, they are trained and tested with the third party database recordings as well for this run.

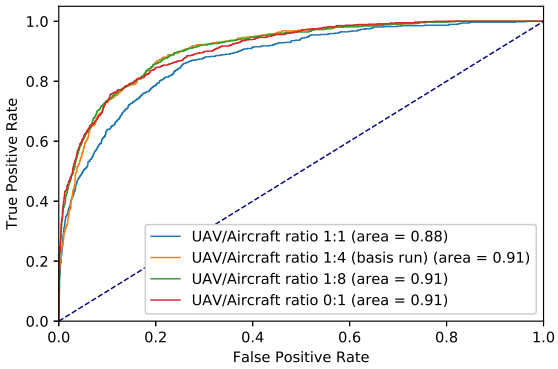
Figure 6 shows that for all the models, the addition of the third party database recordings improves the performance of the model. Only for the very low FPR (< 0.01), the basis run performs better for the MFCC-CNN and the Mel spectrogram-CNN.

4.3 Influence of labeling

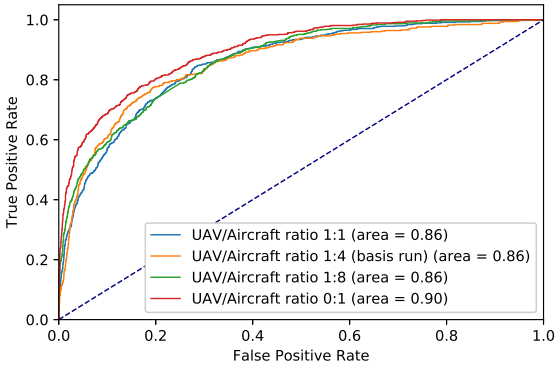
The third type of modification made in the data set relates to which labels are used for training. For all cases the nearby detection labeling is used for testing. For training, however, one run uses distant detection labeling and one run uses nearby detection labeling. When an aircraft is approaching, the lower frequencies of its generated sound reach the ear first. This low frequency content is in the same range as the background noise. It is therefore expected that for distant detection labeling a better separation is found in the model between drone and aircraft and therefore would also better perform for the nearby cases. Figure 7, however, does not prove this hypothesis. This time, for all features, the performance deteriorates when distant detection labeling is used.



(a) MFCC-CNN for different UAV/aircraft amplitude ratios.

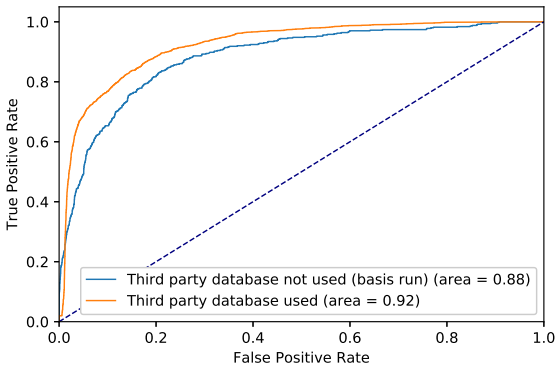


(b) Mel spectrogram-CNN for different UAV/aircraft amplitude ratios.

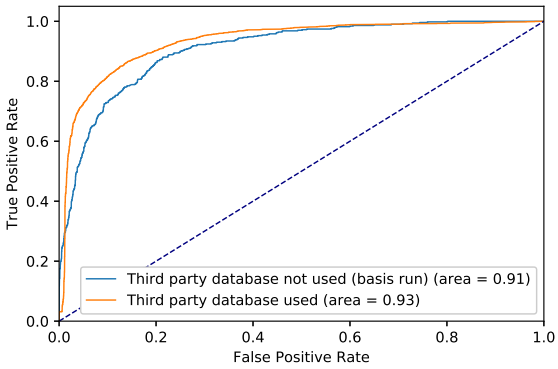


(c) Spectrogram-CNN for different UAV/aircraft amplitude ratios.

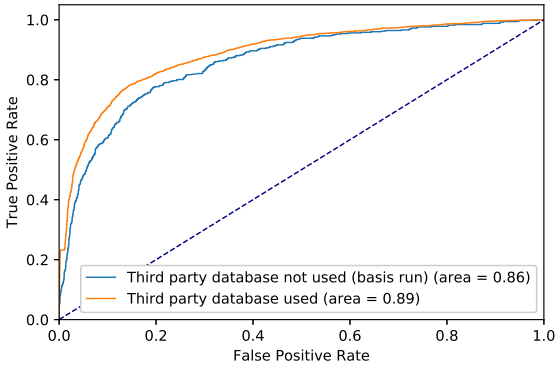
Fig. 5: ROC curves showing the influence of the UAV/aircraft ratio for each feature. Best accuracy is achieved for the ratio 0:1 (no UAV sound present). The more UAV content is added, the worse the performance.



(a) MFCC-CNN with and without third party database recordings.

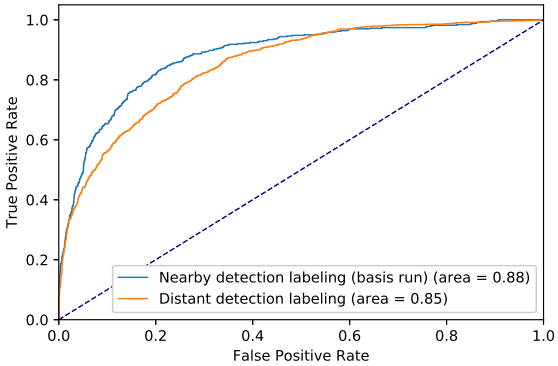


(b) Mel spectrogram-CNN with and without third party database recordings.

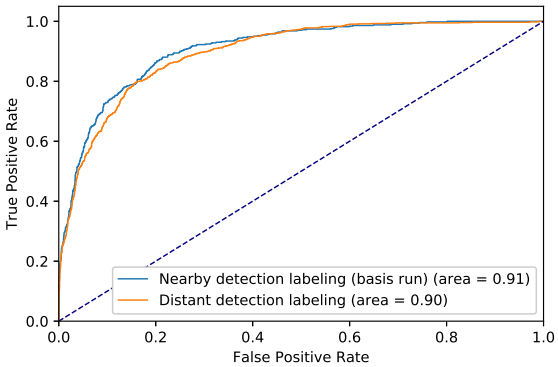


(c) Spectrogram-CNN with and without third party database recordings.

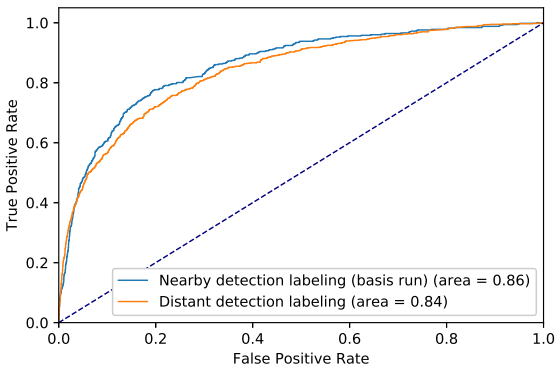
Fig. 6: ROC curves showing the influence of the third party database recordings for each of the features. For all features, the performance increases using the third party database recordings.



(a) MFCC-CNN comparing the performance for different label types.



(b) Mel spectrogram-CNN comparing the performance for different label types.



(c) Spectrogram-CNN comparing the performance for different label types.

Fig. 7: ROC curves showing the influence of labeling type for each of the feature. Each run is tested with nearby detection labeling. One run is using the nearby detection labeling for training as well and the other one uses the distant detection labeling during training.

4.4 Influence of the window length

The window length of the CNN determines how many seconds of history are used to determine whether the sound contains aircraft sound or only UAV sound. The more history the sound contains, the better the development of (possible) aircraft sound can be captured. It is therefore expected that with a larger window length a better performance is achieved. However, eventually the performance of longer time windows are expected to converge as history from long ago does not give useful information in detecting aircraft sound in the present.

This hypothesis is confirmed for the CNNs using Mel spectrogram, spectrogram and MFCC in Figure 8. Improvement in AUC between a three second window and a ten second window is shown in each of the subfigures. For window lengths of more than ten seconds, the AUC hardly changes. For the spectrogram-CNN there is a clear difference in the low FPR region between the 10 and 15 seconds.

4.5 Comparison of the features

So far, the results are only shown per feature. In order to show which feature works best, the features have been compared for the basis run in Figure 9. The results show that the Mel spectrogram performs best, followed by the MFCC. The spectrogram performs worst compared to the other two.

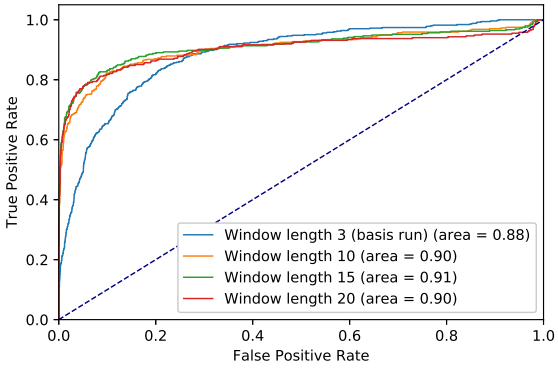
Even though the results are only set out for one run, this is true in general for the other runs. For the runs with a UAV/aircraft ratio of 0:1, 1:1, and distant detection labeling (run 1, 2 and 6) the MFCC is equally accurate as the Mel spectrogram. For the runs with an increase window size (run 7,8 and 9), the spectrogram is slightly better then the MFCC.

Moreover, a ROC curve with the binary discrimination threshold based on the pure energy of the signal is shown in Figure 9. This curve is used to see whether the model just checks the amount of energy in the signal or if it uses more elaborate features. The AUC gives away directly that the performance is significantly worse than the CNNs, so the model does not base its outputs simply on the amount of energy in the signal. Especially in the low FPR region ($<$ 0.1) the TPR is significantly lower than for the CNNs.

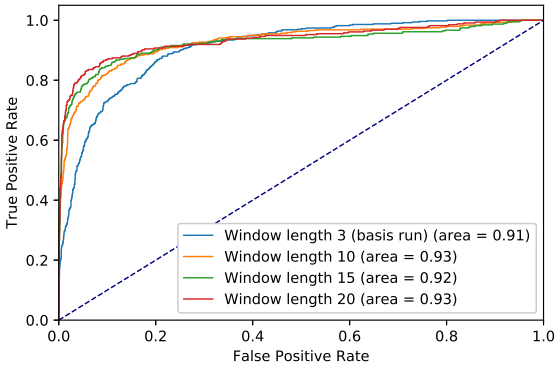
4.6 Visualization of the output

In order to clarify the output of the model, one of the runs is used to visualize the outputs. In Figure 10, the spectrogram of one sample of the basis run test set is shown, along with the expected label (in red), the output of the network (in black) and the binary discrimination threshold belonging to a FPR of 0.1 (in purple). This example shows a decent detection result in which the results in the time window for which the label is 1 (between 28 and 40 seconds) is correctly above the threshold (except for the first second). The rest of the output is always under the threshold and therefore not detected as an aircraft.

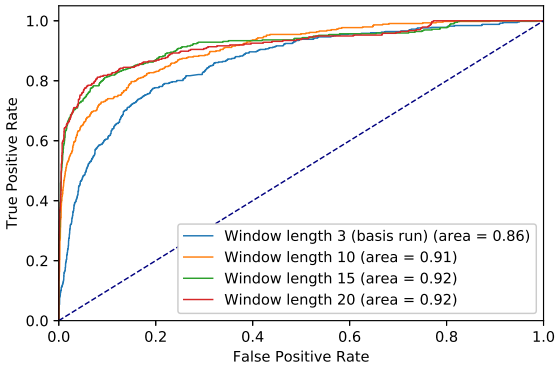
The correctness of the result of Figure 10, however, is not observed for all cases of the test set. False positives and false negatives are appearing as well, such as shown in Figure 11. In this figure the time span between 30 and 45 seconds



(a) MFCC-CNN for different window lengths.



(b) Melspectrogram-CNN for different window lengths.



(c) Spectrogram-CNN for different window lengths.

Fig. 8: ROC curves showing the influence of the window lengths for each feature. In general, the increase in window length increases the performance, but it converges to the performance of a window length of 20 seconds.

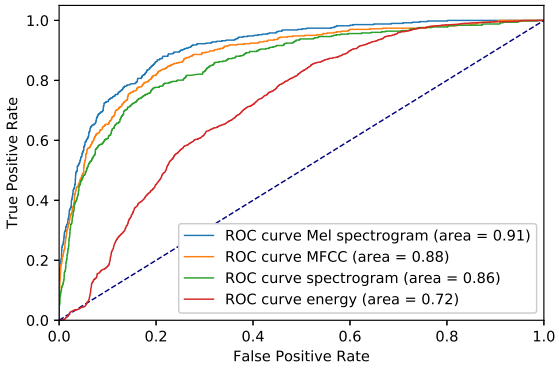


Fig. 9: ROC curves of each feature for the basis run. Also the energy of the signal is used as an input for the ROC curve to show that the model does not base its output only on the energy in the signal. The Mel spectrogram is the best performing feature, MFCC second best, the spectrogram is the worst feature and energy performs significantly worse than all features.

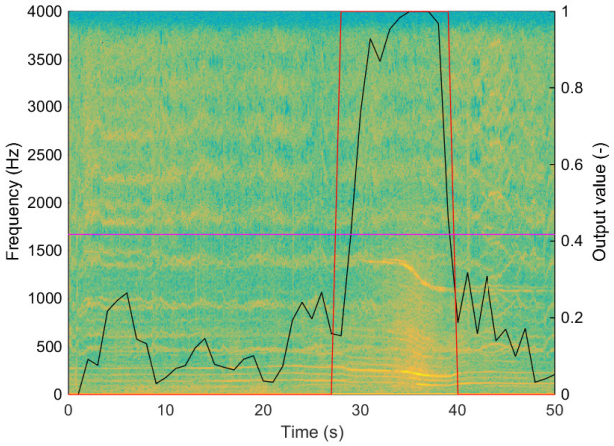


Fig. 10: Correct classification example of a sound sample. In red is the expected label, in black the given output and in purple the discrimination threshold. The left axis belongs to the spectrogram only, the right axis belongs to the output, the label and the threshold lines. As the output is always under the purple line when the label is 0 and above the purple line when the label is 1 (except for 1 second), this sample is accurately classified.

should be given a label of 1, but but the model output is still under the threshold, except for 1 second. Also, the point at second 3 is just above the threshold, whereas it should be labeled 0. On the other hand, also for the human eye the presence of an aircraft is better visible in the spectrogram of Figure 10 than in the spectrogram of Figure 11, due to the Doppler shift and the increase in energy (which can be seen by the increase of the yellow content) in Figure 10.

In order to confirm that the model can recognize Closest Point of Approach (CPA) such as shown in the spectrogram, all the audio samples of the test set of the basis run are

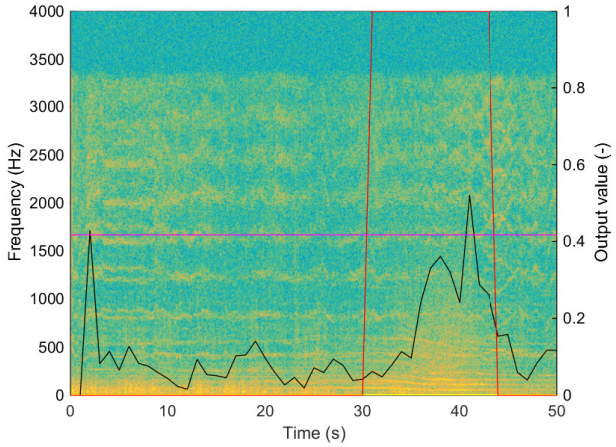


Fig. 11: Partly wrong classification example of a sound sample. In red is the expected label, in black the given output and in purple the discrimination threshold. The left axis belongs to the spectrogram only, the right axis belongs to the output, the label and the threshold lines. A false positive is shown at 3 seconds and false negatives between 30 and 45 seconds (except second 40).

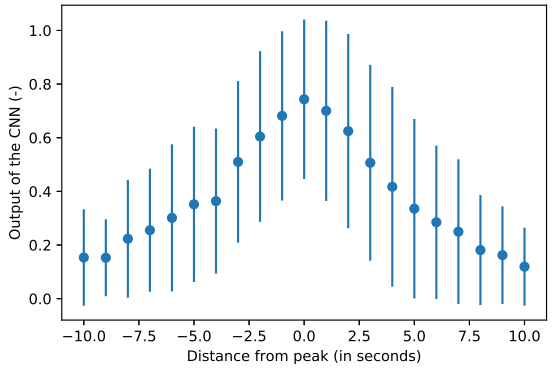


Fig. 12: Means (dots) and standard deviations (bars) per time distance from the center of a CPA in the spectrogram. It shows that the closer the aircraft is, the better the detection performance.

centered around the CPA (if any). For each second in the range of 10 seconds before the CPA and 10 seconds after the CPA, the mean values and standard deviation of the model output are taken. Those values are shown in Figure 12. Each dot represents the value of the mean, each bar the standard deviation from the mean. This figure shows that at the CPA, the output value is usually the highest. Furthermore, the larger the time distance from the CPA, the lower the mean and standard deviation. There is, however, relatively much spread in the output of the network.

4.7 Precision and recall

The AUC gives a good overall indication for the accuracy of the model. However, in order to see how well the model performs per point on the ROC curve, precision and recall is used. Precision is defined in Equation 7, in which FP is the number of false positives and TP is the number of true positives. For recall, also the false negatives FN are used, such as shown in Equation 8.

$$Precision = \frac{TP}{TP + FP} \tag{7}$$

$$Recall = \frac{TP}{TP + FN} \tag{8}$$

In this research, an important value is $1 - recall$ for the label 0. This value shows how many false positives are present, so how often the UAV would falsely perform an avoidance maneuver. The recall for the label 1 is the second most important. It shows how well the aircraft is detected when it is present. The reason that it is less important than the $1 - recall$ for label 0 is because this value does not say when the false negatives appear. It is expected that the closer the aircraft gets, the better the detection performance. Figure 12 shows that this is actually the case for this model. So if the model does not detect the aircraft it is probably not too close, so it would not directly lead to a critical situation. Precision shows how many of the predicted labels are relevant, which is less important for this application than the recall.

An example of the precision and recall and the confusion matrix for the Mel spectrogram-CNN with the window length 20 are shown in Tables 4 and 5 respectively. As a very low FPR beneficial, but still aircraft should detected, the point on the curve for which the ROC curve just separates from the Y-axis is chosen (which is around an FPR of 0.01 and a TPR of 0.7).

TABLE 4: Precision and recall of the Mel spectrogram-CNN using window length 20.

	Precision	**Recall**
0	0.97	0.99
1	0.85	0.70

TABLE 5: Confusion matrix of the Mel spectrogram-CNN using window length 20.

		Predicted class	
		0	1
Actual class	0	2823	42
	1	101	234

5 Discussion

The results shown in section 4 are further discussed in this section. Starting off with the different UAV/aircraft amplitude ratios, Figure 5 shows in the lower FPR region an expected trend, which is that the lower the UAV amplitude is compared to the aircraft amplitude, the better the aircraft is detected. That means, in order to use this model for real-world application, it is best to diminish the UAV's ego-sound as much as possible, for example by means of the method of Marmaroli et al. [8].

The addition of third party database recordings also improves the performance, such as shown in Figure 6. Those recordings consist of different background noise, which

could be easier for the model to distinguish from the typical background noise from the Lelystad recordings. The basis run performed better in the very low FPR (< 0.01), but the corresponding TPR is to low to be a good detector.

The fact that the different type of labeling performs worse, which is shown in Figure 7, is unexpected. The labels that are 1 for the distant detection labeling consist of the ones from nearby detection labeling plus some extra ones before and after. In other words, the nearby detection labels are a part of the distant detection labels. As the distant detection labeling includes the nearby detection labels, it is expected that training with distant detection labeling at least performs the same as training with nearby detection labeling. However, the model performs worse (or equal, for any FPR lower than 0.05) which means that there is no benefit in using the distant detection labeling. The consequence of using nearby detection labeling over distant detection labeling is that the aircraft is closer to the UAV when it is detected.

The trends shown in Figure 8, at which the window length is increased, are not unexpected. The longer the window length, the more information the model uses to make a decision and therefore the performance is better. This only works up to a certain amount since sound information to far in the past can have nothing to do with the present sound. Based on the presented experiments, a window length between 15 and 20 seconds should be used to be as accurate as possible. Choosing a value above 20 seconds will not increase the performance and makes it computationally more expensive. Of course, also other forms of memory can be explored, such as Long Short Term Memory [20] or GRU [21].

In the ideal situation, no false positives or false negatives are present in the output of the detector. Since the ROC curves in Figures 5, 6, 7 and 8 never have an AUC of 1, this is not possible. Therefore, we aim to have as little false positives and false negatives. In Table 4 and Table 5 a limit of one false positive in 100 seconds is set. If after a false positive a warning is send to the operator, once in 100 seconds he/she has to check whether there is really other air traffic present, which is not increasing the workload to much and therefore once in a 100 seconds is a reasonable limit. If the UAV has to descend (or even land) after a detection, a false positive once in a 100 seconds is too much, so for those cases a filter should be applied, which checks whether multiple positive detections are found in a short-time frame. The percentage of missed detections corresponding to a once in a 100 FPR, is 30%. Luckily, Figure 12 shows that the closer the aircraft is the better the accuracy, so the missed detections will be mostly appear in the early stages of the detection.

Alongside the conclusions drawn from the results, there are a few general comments to be made concerning the research method.

Firstly, the data set should be extended. The data set used in the basis case (run 3) only contains the recordings from Lelystad airport. This data set has in total 84 flyovers. The data augmentation increases the data set times four, so 336 flyovers are available for the data set. This is considered a relatively small data set for machine learning purposes such as this research. For comparison, ImageNet[8], a famous data set for image recognition, has 15 million examples in total. In addition, the ratio of the data set that includes aircraft sound and that only includes background noise is not 50/50, due to the fact that the cut-outs from the recordings are random. The ratio aircraft/background in this data set is approximately 20/80. The problem with this ratio is that the model could classify all the sound samples as background noise and still would have an accuracy of 80%. Another comment about the data set is that it is artificially mixed, so the UAV and aircraft sound are individually recorded. In the spectrogram, it is visible where the aircraft sound is added to the UAV sound by vertical lines at the stop and start. An example is shown in Figure 13, at which the aircraft recording part stops at 30 seconds. In order to avoid this effect, recordings should be taken on a UAV, which flies close to flying aircraft.

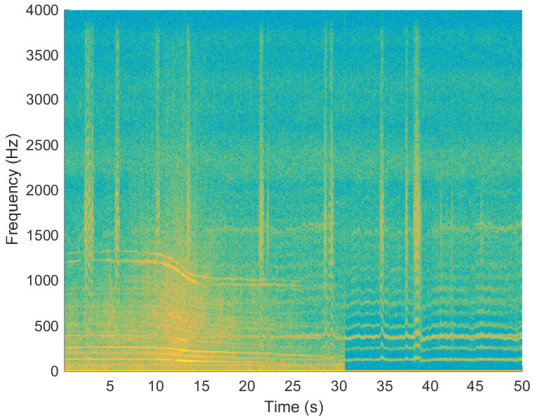


Fig. 13: Spectrogram of a mix of UAV and aircraft sound. The end of the aircraft sound recording is visible on the spectrogram at 30 seconds by the vertical line (which is the sudden decrease in energy).

So far, the only different scale used is the Mel scale. Two features use this scale which mimics the way humans perceive frequency. The comparison of the Mel spectrogram and the spectrogram in Figure 9 shows that stretching the lower frequencies works well in combination with the CNN. One idea is to make a scale that stretches the lower frequencies even more. As most of the distant aircraft sound lies in the low frequency region, further stretching the lower frequencies could show more important low frequency sound information for the CNN.

What is more, is that there is not much difference in type of background noise. Only two types of microphones are used, the 808 micro camera microphone and the microphone from the array. Different microphones could show different noise content. Further research in the quality of the microphones is demanded. Also, the background noise is pretty constant during the recordings, whereas on a flying UAV this could differ considerably. Other background noise, such as cars, trains, lawnmowers, etc., is not added.

Not only is there one composition of background noise, but also only one type of UAV sound has been used. In order

8. http://www.image-net.org/

to make a model for versatile applications, multiple UAV sounds should be included in the data set. If the model is applied to only one UAV, it is useful to use its specific model in training the detection network. In this process it is also important to check whether the ego-noise of the UAV is in the same order of loudness as the Parrot Bebop used in this research.

6 Conclusion

Detection of air traffic sounds on a UAVs could increase the safety of the airspace. This paper builds on existing sound features and classification methods, but this time applied to combined UAV and aircraft sound.

The three features used are the MFCC, spectrogram and Mel spectrogram, which are the input to a CNN classifier. The best performance of the model is obtained using the Mel spectrogram, which moves over the sound recording with a 20-second window length. The detection performance increases when the aircraft is closer to the UAV. Longer time windows give better performance up until a certain window length, but also decrease the potential reaction time for an avoidance maneuver. Secondly, the model works best if as little UAV sound is present as possible. Thirdly, the current method still gives too many false positives for real-world application. Improvements may be expected from a better filtering over time (ignoring solitary peaks of the network's output), a more extensive data set, and potentially additional information such as the commanded RPMs of the UAV's propeller(s). Finally, a more realistic data set should include sound recordings of aircraft taken from a (moving) UAV.

Acknowledgments

I would like to thank my supervisors Tom van Dijk, Guido the Croon and Christophe de Wagter for giving me the opportunity to perform this research and support me when needed. Secondly, special thanks to the Aircraft Noise and Climate Effects (ANCE) section of the TU Delft for providing me the recording material for the recordings from Lelystad airport, in particular to Mirjam Snellen, Salil Luesutthiviboon, Ana Alves Vieira and Anwar Malgoezar.

References

[1] E. Tijs, G. de Croon, J. Wind, B. Remes, C. De Wagter, H. de Bree, and R. Ruijsink, "Hear-and-avoid for micro air vehicles," in *Proceedings of the International Micro Air Vehicle Conference and Competitions (IMAV), Braunschweig, Germany*, vol. 69, 2010.

[2] H.-E. De Bree and G. De Croon, "Acoustic vector sensors on small unmanned air vehicles," *the SMi Unmanned Aircraft Systems, UK*, 2011.

[3] M. Basiri, F. Schill, P. U. Lima, and D. Floreano, "Robust acoustic source localization of emergency signals from Micro Air Vehicles," *IEEE International Conference on Intelligent Robots and Systems*, pp. 4737–4742, 2012.

[4] M. Basiri and F. Schill, "Audio-based Relative Positioning System for Multiple Micro Air Vehicle Systems." *Robotics: Science and Systems*, no. 266470, 2013. [Online]. Available: http://www.roboticsproceedings.org/rss09/p02.pdf

[5] M. Basiri, F. Schill, D. Floreano, and P. U. Lima, "Audio-based localization for swarms of micro air vehicles," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, may 2014, pp. 4729–4734. [Online]. Available: http://ieeexplore.ieee.org/document/6907551/

[6] M. Basiri, F. Schill, P. Lima, and D. Floreano, "On-Board Relative Bearing Estimation for Teams of Drones Using Sound," *IEEE Robotics and Automation Letters*, vol. 1, no. 2, pp. 820–827, 2016.

[7] B. Harvey and S. O'Young, "Acoustic Detection of a Fixed-Wing UAV," *Drones*, vol. 2, no. 1, p. 4, jan 2018. [Online]. Available: http://www.mdpi.com/2504-446X/2/1/4

[8] P. Marmaroli, X. Falourd, and H. Lissek, "A uav motor denoising technique to improve localization of surrounding noisy aircrafts: proof of concept for anti-collision systems," in *Acoustics 2012*, 2012.

[9] H. Phan, L. Hertel, M. Maass, and A. Mertins, "Robust audio event recognition with 1-max pooling convolutional neural networks," *Proceedings of the Annual Conference of the International Speech Communication Association, INTERSPEECH*, vol. 08-12-Sept, no. 1, pp. 3653–3657, 2016.

[10] I. McLoughlin, H. Zhang, Z. Xie, Y. Song, W. Xiao, and H. Phan, "Continuous robust sound event classification using time-frequency features and deep learning," *PLoS ONE*, vol. 12, no. 9, p. e0182309, sep 2017. [Online]. Available: http://dx.plos.org/10.1371/journal.pone.0182309

[11] H. Zhang, I. McLoughlin, and Y. Song, "Robust sound event recognition using convolutional neural networks," in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, vol. 2015-Augus. IEEE, apr 2015, pp. 559–563. [Online]. Available: http://ieeexplore.ieee.org/document/7178031/

[12] S. Doljé, "Quantifying microphone array directivity," Master's thesis, Delft University of Technology, dec 2017.

[13] B. McFee, C. Raffel, D. Liang, D. P. Ellis, M. McVicar, E. Battenberg, and O. Nieto, "librosa: Audio and music signal analysis in python," in *Proceedings of the 14th python in science conference*, 2015, pp. 18–25.

[14] B. Logan, "Mel frequency cepstral coefficients for music modeling." in *ISMIR*, vol. 270, 2000, pp. 1–11.

[15] R. F. Lyon, "Machine hearing: An emerging field," *IEEE Signal Processing Magazine*, vol. 27, no. 5, pp. 131–136, 2010.

[16] F. Zheng, G. Zhang, and Z. Song, "Comparison of different implementations of mfcc," *Journal of Computer science and Technology*, vol. 16, no. 6, pp. 582–589, 2001.

[17] F. Chollet, "Keras," https://keras.io, 2015.

[18] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard *et al.*, "Tensorflow: a system for large-scale machine learning." in *OSDI*, vol. 16, 2016, pp. 265–283.

[19] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[20] T. N. Sainath, O. Vinyals, A. Senior, and H. Sak, "Convolutional, Long Short-Term Memory, fully connected Deep Neural Networks," in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, vol. 2015-Augus. IEEE, apr 2015, pp. 4580–4584. [Online]. Available: http://ieeexplore.ieee.org/document/7178838/

[21] E. Cakir, G. Parascandolo, T. Heittola, H. Huttunen, and T. Virtanen, "Convolutional Recurrent Neural Networks for Polyphonic Sound Event Detection," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 25, no. 6, pp. 1291–1303, jun 2017. [Online]. Available: http://arxiv.org/abs/1702.06286http://dx.doi.org/10.1109/TASLP.2017.2690575http://ieeexplore.ieee.org/document/7933050/

II

Scientific Paper Appendices

A

List of third party database recordings

As explained in the paper, third party database recordings are used in order to create a larger dataset for the research. In Table A.1, the names of the recordings that are used, the username of the owner of the recording, and the number by which the recording can be found on `https://freesound.org/` are listed.

Table A.1: The names, corresponding usernames and search numbers for the third party database recordings used.

Name	Username	Search Number
police_helicopter.wav	reinsamba	36759
Robinson flying loop.wav	Figowitz	69607
helicopter02.wav	gezortenplotz	77312
Helicopter Landing	Mings	150976
heli3.wav	nicobaba	159778
blackhwak_flyby.wav	primeval_polypod	160613
helicopter3.wav	Zabuhailo	173431
helicopter1.wav	dwareing	181071
Apache helicopter fly over 01.wav	klankbeeld	191429
helicopter over neighborhood.wav	hachaduryan	203355
Helicopter 002.wav	GiovanniProvenzale	237183
Helicopter 001.wav	GiovanniProvenzale	237184
HELICOPER	towers4223	253643
Helicopter in Mountains	DallasBass	258007
Cobra_Helicopter_Flyover_Ext.wav	Cell31_Sound_Productions	263458
helicopter.wav	UncleSigmund	271214
Helicopter 1.WAV	ElementRS2	343685
HELICOPTER OVERHEAD V#1 CLEANFINAL.WAV	metrostock99	345084
helicopter.L.wav	nhaudio	346372
Sea Hawk Helicopter NH60-R.wav	granconill	380079
Helicopter_Passby_1.wav	pan14	388287
Helicopter 2.wav	BeeProductive	395594
Helicopter.wav	BeeProductive	395601
A helicopter flying low above the ground	seenms	421614
plane.wav	UncleSigmund	36929
00489 aircraft run2.wav	Robinhood76	62049
retro airplane.wav	Inplano	81056
OverpassingPlane.wav	adamlhumphrey	115387
Biplane fly-by.wav	debsound	117270
Plane Cessna start stop.wav	Cheeseheadburger	141519
Two Spitfires Flyby	Flight2FlyPhoto	142901
Bf-109 Flyby	Flight2FlyPhoto	143558
Low Airplane Fly By	Snipperbes	145365
porpplaneflyover.wav	Laers	169129
Spitfire_slow_OH.wav	beerbelly38	276550
AIRPLANE - small airplane flying by (SFX)	Anika11	325683
PlaneFlyoverDistant.wav	kingsrow	348595
Small propeller plane	jay_rope	361434
A small propeller plane.wav	straget	403316

B

Lelystad airport recording procedure

As the most probable group to come in contact with UAVs is general aviation (GA) rotor- and aircraft, flyover data has been obtained at the biggest GA airfield of the Netherlands, Lelystad Airport, in collaboration with the Aircraft Noise and Climate Effects (ANCE) section of the TU Delft. In order to get permission to record at the airstrip, the operations manager Edward de Kruijf is consulted by `info@lelystad-airport.nl`.

As Lelystad airport is expending to a larger airfield, the runway is extended, but the new part is not in use yet. This part of the runway is therefore a perfect location to obtain recordings as the aircraft would fly straight over the so-called acoustic camera. This location is indicated in Figure B.1 by a red arrow. The aircraft land (and depart) from south-west to north-east, so the landing aircraft flyovers are obtained. The departures were to far away to be captured. The trajectories of aircraft and helicopters are indicated by the orange and blue lines respectively.

The acoustic camera, designed and build by the TU Delft [14], consist of an array with 8 bundles of 8 microphones. The bundles are arranged in a spiral shape for optimal beamforming purposes. The microphones are covered in a foam layer to decrease the noise due to wind. Moreover, the array is covered in foam in order to absorb ground reflections. All the bundles are connected to a Data Acquisition Box (DAQ) which samples the data at 50 kHz and sends it to the connected computer. On the computer, Labview is used to visualize and store the data. Not only the DAQ is connected to the computer, but also an ADS-B receiver in order to receive aircraft position information and an optical camera to capture the flyover. This camera was positioned in the center of the array. However, the ADS-B did not produce useful information as none of the GA aircraft send out ADS-B information and the optical camera failed to work. In order to solve this a mobile phone camera is placed in the center of the array, but is not synced with the DAQ. However, for this research the video was not needed anyway. The complete the acoustic camera is shown in Figure B.2 and the setup of the plates and microphones are presented in Figure B.3. For more information about the array, the file 'manual camera.docx' and [14] should be consulted. The manual, from which the set up of the different plates of the array with the corresponding positions for the microphone are taken, includes the list of materials that are needed for using the array, as well as specifications for all parts of the equipment and the use of different .mat files.

The recordings are made on the 29th of June, 2018. The this day was a warm summer day with perfect blue skies, which is shown per hour in Table B.1. Before and after the time range used in this table, no recordings were made.

In total 75 recordings are obtained, which consist of background noise recordings and flyovers. One recording sometimes consist of more than one flyover. Effectively, 75 GA aircraft and 9 helicopter flyovers are captured. The background noise consisted of microphone noise, noise due to wind, distant traffic and a distant motor race track. In the folder of the recordings is a file present called 'data matrix 29062018 v2.xlsx'. This file shows which flyovers are present in which recordings. It shows the time of the start of the recording, the aircraft's registration, the corresponding video number (if any), the aircraft types and some remarks. If a recording only consist of background noise this is indicated in the column 'registration' by 'bgn'.

For this research only the recording of one microphone is necessary, so from only one microphone the recordings are extracted. Every microphone is checked to make sure it worked correctly. One of the 64 microphones is faulty, so its data is not used.

The recordings from Lelystad airport are made using Labview, which saves the recording as a binary file. The recordings have to be preprocessed (for calibration purposes), which normally is performed with Labview

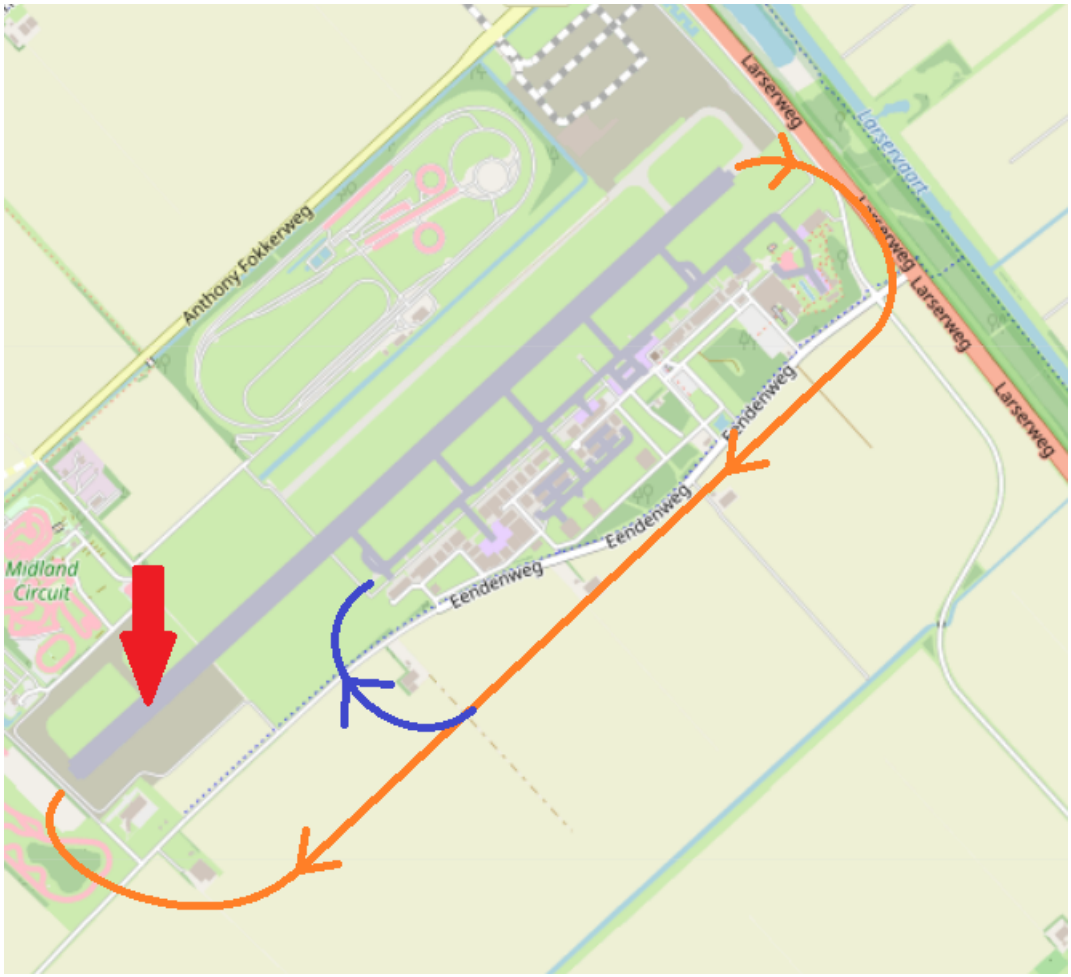


Figure B.1: Map of Lelystad airport. The red arrow indicates the position of the array for the recordings, the orange line shows the most common aircraft approaches/departures and the blue lines show the helicopter trajectory.

as well. However, the recordings from Lelystad were sometimes too long, therefore the files were too big to be preprocessed in Labview. So before the files could be preprocessed, they are cut using the 'cut_data.m' file. It cuts the original file in separate files existing of 80 seconds of recordings. Secondly, the trimmed files are read out using 'ReadAll.m', which depends on 'Read_data.m'. It scrolls through all folders and converts the trimmed files to .mat files. In order for this program to work, the 'ReadAll.m' file, the 'Read_data.m' file and the Calibration folder should be in the same folder. Lastly, the .mat files are converted to .wav files using 'SaveOneMic.m'. In this file, also the low-pass filter is applied and saved in a separate .wav file. For all the scripts, only the paths should be changed to the corresponding folder in order to work.

Figure B.2: Acoustic camera on the runway of Lelystad Airport.

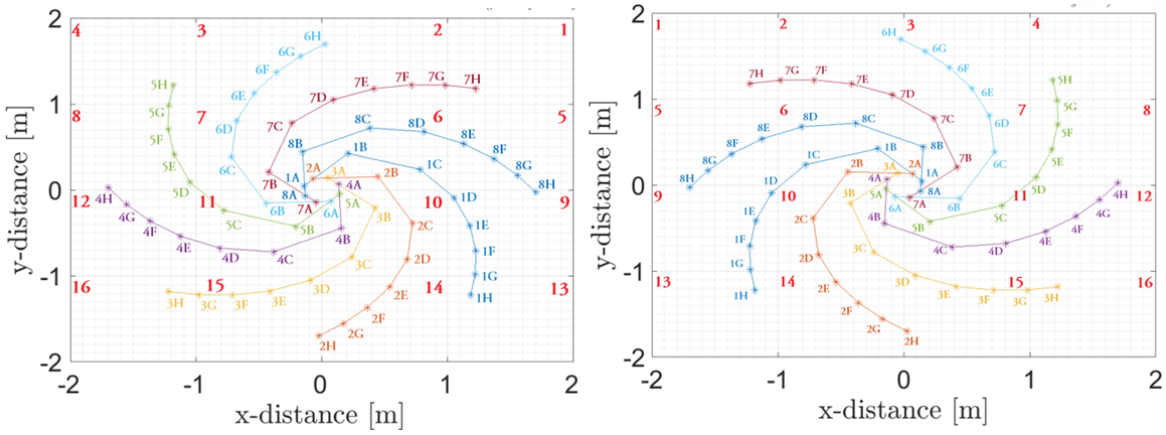


Figure B.3: Microphone configuration from the front (left) and the back (right).

Table B.1: Weather data of Lelystad airport on the 29th of June, 2018

Hour	Mean wind direction (in degrees)	Hourly mean wind speed (m/s)	Temperature (in degrees)	Sunshine duration (in hours)	Global radiation (J/cm2)	Precipitation duration (in hours)	Hourly precipitation amount (mm)	Air pressure (hPa)	Cloud cover (in octants)	Relative atmospheric humidity (%)
10	360	3.0	22.0	1	274	0	0	1020.8	0	59
11	350	4.0	22.6	1	298	0	0	1020.4	0	57
12	350	3.0	23.4	1	309	0	0	1020.1	0	52
13	340	4.0	23.1	1	303	0	0	1019.7	0	65
14	350	6.0	22.7	1	288	0	0	1019.3	0	65
15	350	7.0	21.9	1	259	0	0	1019.1	0	63
16	350	6.0	20.8	1	216	0	0	1018.7	0	70
17	340	6.0	19.6	1	164	0	0	1018.4	0	74

C

Overview of the Python scripts

Most of the code written in this research was performed using Python 3.6. For some additional purposes, Matlab R2014b was used. The Matlab files are explained in Appendix B and in 'manaul camera.docx'. One general note for all of the scripts is that all the paths should be modified to the corresponding folders. More information about the code will be present as comments in the code as well.

C.1. Main.py

This file is used for mixing the sounds, store them and get the features of them, which are also stored. There are only 7 parameters to be checked before running the code.

First of all, the sample frequency 'fs' has to be set correctly. In this research it is set on 8000 Hz in order to have computationally low cost. However, it also means that a lot of sound information is 'thrown away' as the .wav files were usually sampled at either 22500 Hz, 44000 Hz or 50000 Hz. Secondly, the 'hop_length' should be chosen properly. So far, it has been set equal to the 'fs', which results in spectrograms and an MFCC having one frame per second. In order to change the amount of frames in the time axis for the spectrograms and MFCC, the 'hop_length' should be altered. Thirdly, the 'min_sound_length' is an important variable as it decides whether it is needed to cut mixed recordings. If the variable 'sound_generation' (which will be explained below) is set on 'True', 'min_sound_length' needs to be set on 'float('inf')' (after which the right value will be computed automatically). If 'sound_generation' is 'False', 'min_sound_length' should be have the correct value, so it is important to write this number down after a run with 'sound_generation = True' is performed. The fourth variable is 'ratio', which indicates the amount of aircraft content in the amplitude ratio UAV/aircraft. For example, if it is set on 4, the ratio is 1:4. Finally there are three booleans, 'sound_generation', 'melspectrogram_generation' and 'aircraftANDdrone'. The 'sound_generation' indicates if the mixing procedure should be performed again, which need to happen after altering one of the previously described variables. It automatically deletes the previous data if it is generating new data. Then, 'melspectrogram_generation' indicates whether the features should be computed again. Finally, if 'aircraftANDdrone' is 'False', it generates the data for the case in which no UAV sound is added.

Apart from those variables, there are two variables in the code that also could be altered. They can be found in the function 'mix_recordings'. The variables of interest are 'noise' and 'label_path'. The first one indicates whether the original recordings should be used, or the ones which are low-pass filtered. If 'noise' is '0', the low-pass filtered recordings are used, if not, then the original recordings are used. In order to determine what type of labeling should be performed, 'label_path' is set on 'label.txt.txt' for distant detection labeling or on 'label2.txt.txt' for nearby detection labeling. If distant detection labeling is used, the file 'ChangeLabel.py' is needed afterwards, which is explained in section C.2.

C.2. ChangeLabel.py

This script is only needed when distant detection labeling is used in 'Main.py'. When 'Main.py' is run with the distant detection labeling, also the distant detection labels will be present in test set, which is not the intention of the run. Therefore, only the labels in the test set needs to be changed with the corresponding nearby detection labels, which is performed by 'ChangeLabel.py'. It is important that after 'ChangeLabel.py'

is run, also 'Main.py' should be run again with the following settings: 'sound_generation' is 'False' and 'mel-spectrogram_generation' is 'True. This ensures that now the nearby detection labels will be used in the test set.

C.3. CNN-Mel.py

The information that is described in this section, is also valid for the files 'CNN-MFCC' and 'CNN-Spec' (as well as the SIF and CRNN files). In these files the CNN is build, trained, tested and saved. The variable 'batch_len' can be changed in order to speed up the training, however, this value should always be such that it is a multiple of the amount of sound samples in the test set and a multiple of the amount of sound samples in the training set, otherwise it skips sound samples. In order to change the length of the history used, 'width' should be modified.

C.4. GenerateResults.py

This section is also valid for 'GenerateResultsCRNN.py'. Even though the files from section C.3 generate results, they are not saved. 'GenerateResults.py' loads the models, get the test results again and saves them. Not only are they saved for Python purposes (as .npy files), but also for Matlab purposes (as .m files). In this research Matlab was used for the generation of the spectrograms, so also the spectrograms over which the output was plotted. Furthermore, this file generates the error bar such as shown in the paper.

In order to use this file, the right settings should be entered. The first variable is 'results', which is a string that indicates the name of the folder in which the model (and the data set) is stored. Secondly, the variable 'width', just like in the 'CNN-Mel.py', indicates the time history used in the model. Also 'batch_len' has the same purpose (and the same requirements) as in 'CNN-Mel.py'.

C.5. GeneratePlots.py

The plots such as shown in the paper are generated by 'GeneratePlots.py'. There are two options: plotting a selection of runs per feature, or plotting one run for all features. For the first one, the feature should be chosen using the variable 'feature'. Then, the required folders that contain the data that you want to plot are inserted in 'selection'. For plotting all the features, the lines of 'feature' up until the for loop should be commented. Then the (commented) lines starting from 'run' up until the for loop should be uncommented. Secondly, in the variable 'run' the folder name of the correct run should be chosen. Running this file plots the requested runs/features and saves them with a unique name per run/feature combination.

D

Spectrogram Image Feature

The most modern feature used in this research is the Spectrogram Image Feature (SIF). It already has proved itself a reliable feature for sound event recognition, even in noisy conditions [9]. However, due to the strange trends observed in the results of the SIF-CNN, it has not been included in the paper.

The procedure to obtain the SIF is taken from Dennis, 2011[15]. The first steps are to compute a spectrogram image and convert it to a greyscale image. The spectrogram is obtained following the method described in the paper and then normalized to scale the values between [0, 1] and thus creating the greyscale image. The normalization process is explained in Equation D.1, in which $S(k,t)$ is the original spectrogram and $G(k,t)$ the greyscale version.

$$G(k,t) = \frac{S(k,t) - \min(S)}{\max(S) - \min(S)} \tag{D.1}$$

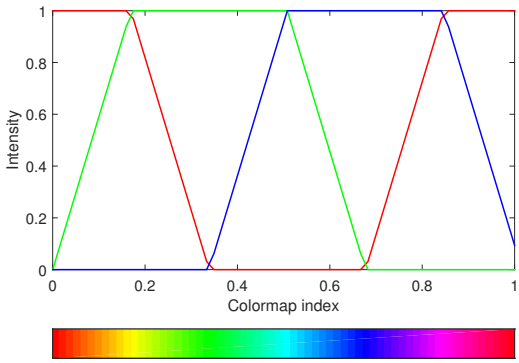


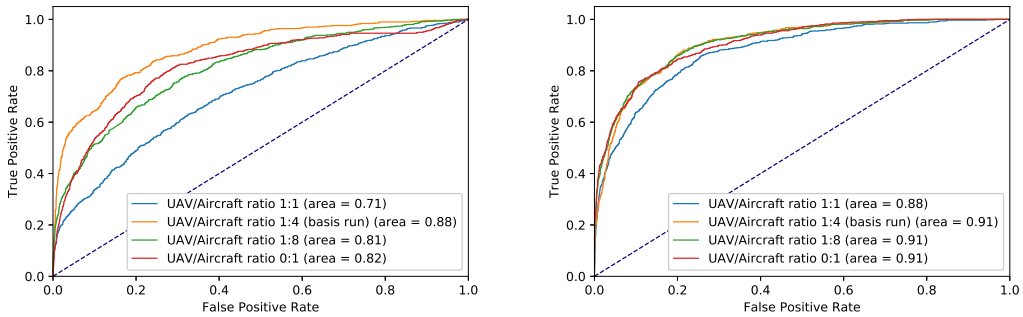
Figure D.1: HSV colormap

Thirdly, the greyscale image is quantized into three monochrome image. Red, green and blue (RGB) are the colors used for this (nonlinear) mapping by means of a HSV colormap. This colormap is presented in Figure D.1. The (normalized) pixel value takes for each monochrome image the intensity belonging to the pixel value. The three monochrome images are subjected to color distribution statistics as it can describe the sound intensity variation in frequency and time. In order to do so, first the the images are split up in 51x51 blocks, of which for each block the two central moments of Equation D.2, $k = 1$ and $k = 2$, are calculated. E is the expectation operator, X the distribution per block and μ_k the k^{th} moment. These SIF comprises of the central moments for each monochrome image and is therefore a (51x51x3x2)-dimensional vector.

$$\mu_k = E\left[(X - E[X])^k\right] \tag{D.2}$$

The CNN is modified for the SIF compared to the CNN for the MFCC, spectrogram or Mel spectrogram, as there is a vector input instead of a 2D array. Therefore, the convolutional layers are changed to 1D layers. The other parameters and the architecture of the model is kept the same.

The results of the SIF-CNN are shown in Figures D.2, D.3, D.4 and D.5. For comparison, the results of the Mel spectrogram-CNN are shown on the right side. For the different UAV/aircraft ratios, it is expected that the more UAV content is present, the worse the model would perform. In the Mel spectrogram this is visible, the worst performance is obtained by a ratio 1:1 and the best performance (especially in the low false positive rate region) is obtained when no UAV sound is present. For the SIF, however, the performance is much more spread and inexplicable. The best performance is achieved by a ratio 1:4, followed by 0:1, then 1:8 and the worst is still 1:1.

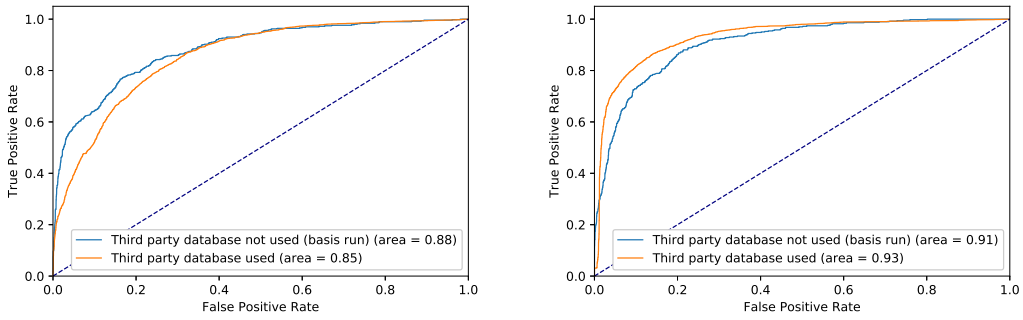


(a) SIF-CNN for different UAV/aircraft ratios. Best performance is obtained by the 1:4 ratio, followed by the ratio 0:1, then 1:8 and the worst is 1:1.

(b) Mel spectrogram-CNN for different UAV/aircraft ratios. Best performance is obtained by the 1:4 ratio, followed by the ratio 0:1, then 1:8 and the worst is 1:1.

Figure D.2: ROC curves showing the influence of the UAV/aircraft ratio.

With the addition of the third party database recordings, the SIF also performs differently then the others. Whereas for the Mel spectrogram (and the other features) the performance increased by using the third party database, for the SIF the performance decreases. This is shown in Figure D.3.

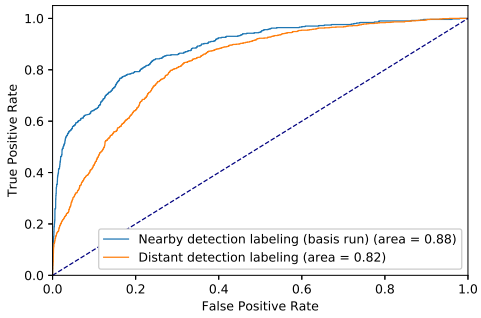


(a) SIF-CNN with and without third party database recordings. For this model the accuracy is higher if no third party database recordings are used.

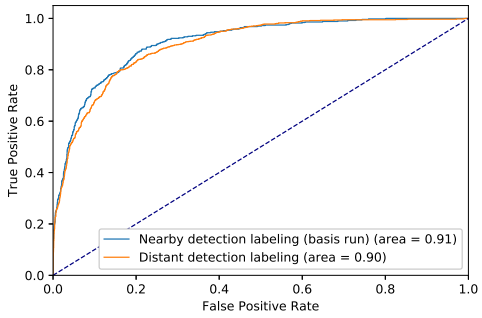
(b) Mel spectrogram-CNN with and without third party database recordings. For this model the accuracy is higher if third party database recordings are used.

Figure D.3: ROC curves showing the influence of the third party database recordings.

For the different types of labeling the SIF does show the same behavior as the Mel spectrogram, which is shown in Figure D.4. However, the decrease in performance is much bigger in case of the SIF (6%) compared to the other features (which is at most 3%).



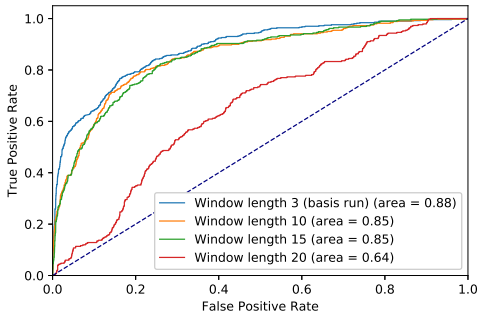
(a) SIF-CNN comparing the performance for different label types. Nearby detection labeling gives better performance than distant detection labeling.



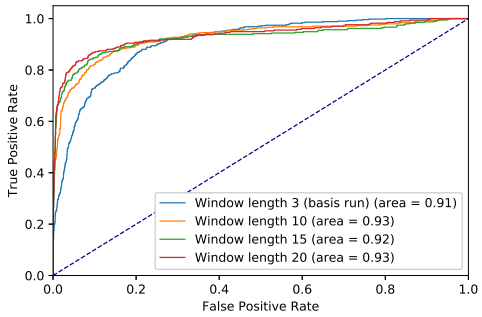
(b) Mel spectrogram-CNN comparing the performance for different label types. Nearby detection labeling gives better performance than distant detection labeling.

Figure D.4: ROC curves showing the influence of the labeling type.

Finally, the results presented in Figure D.5 show that the SIF actually benefits with shorter window lengths, whereas for the other features the have better performance when the window length is increase. Especially after a window length of 15 seconds the performance of the SIF drops significantly.



(a) SIF-CNN for different window lengths. The performance drops with increasing window length.



(b) Melspectrogram-CNN for different window lengths. The performance increases with increasing window length.

Figure D.5: ROC curves showing the influence of the window lengths.

The reason that the SIF does not work has remained unclear during this research. Due to the great performance that it could have for sound event recognition tasks [9], it is recommended to perform further research on SIF for this application.

E

Convolutional Recurrent Neural Network

During this research, a second model is used: the convolutional recurrent neural network (CRNN). As the name implies, the model is a CNN with an extra recurrent layer. The idea of combining the two comes from the fact that CNNs miss out on longer temporal context information and recurrent layers are not able to obtain the invariance in the frequency domain [16]. Combining the two therefore would solve each others disadvantage. Similar to the SIF, the results were not sufficient enough to be used in the paper.

The CRNN also begins with two sets of layers, but are composed differently than the CNN's first layers. For the CRNN, one set is composed of only one convolutional layer, followed by a max pooling layer. Also a batch normalization layer is tried, which should decreases the time required for training by normalizing the output of the layer to a mean of zero and a variance of one, however, the results are worse when this layer is added. After two convolutional/pooling sets, the feature maps are stacked and inserted in the recurrent layer, for which a Gated Recurrent Unit (GRU) is chosen. Those type of units are performing better on small datasets compared to the most common type of Recurrent Neural Network (RNN) unit, the Long Short Term Memory (LSTM). The GRU uses the sigmoid activation layer and gives an output for each frame, only based on historical (and current) inputs. The CRNN has the same loss function and optimizer as the CNN, as well as the dropout between the sets and the fully connected layer. The complete CRNN architecture is shown in Figure E.1 and the corresponding parameter values in Table E.1.

The CRNN also starts with convolutional and pooling layers. In order to decrease the time required for training, also batch normalization layers are applied. After 2 of those combined layers, the outputs are stacked and fed into a Gated Recurrent Unit (GRU). Those type of units are chosen as they perform better on small data sets compared to the most common type of Recurrent Neural Network (RNN) unit, the Long Short Term Memory (LSTM). Finally the outputs of the GRU are connected to a fully connected layer, that has the same outputs as the CNN model. Also in this model dropout is used.

The ROC curves belonging to the output of each CRNN for all the runs are shown in Figures E.2, E.3 and E.4. The window length is not changed as the input for the CRNN is the complete sound sample.

First of all, Figure E.2 shows the influence of changing the UAV/aircraft ratio. The MFCC-CRNN shows the expected output, which is that the detection performance increases when more aircraft content is present. The spectrogram-CRNN and the Mel spectrogram-CRNN, however, show that the ratio of 1:4 works better

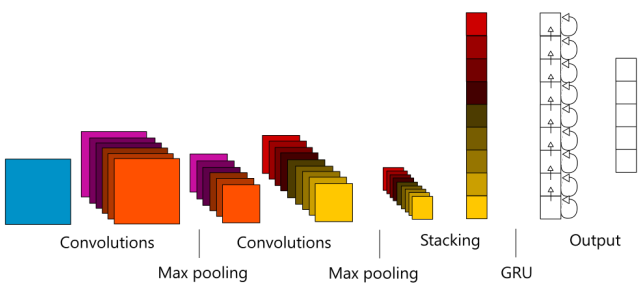
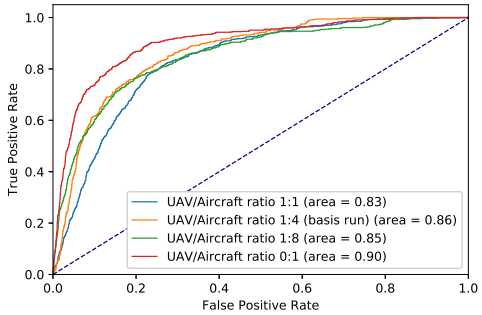


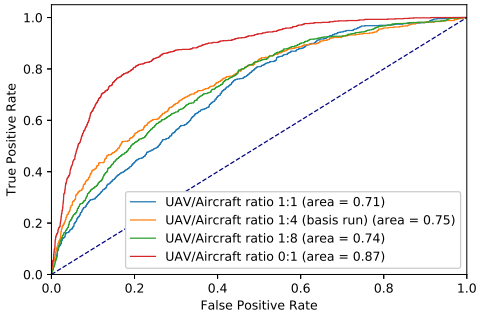
Figure E.1: Architecture of the CRNN.

Table E.1: Model parameters of the CRNN from Figure E.1.

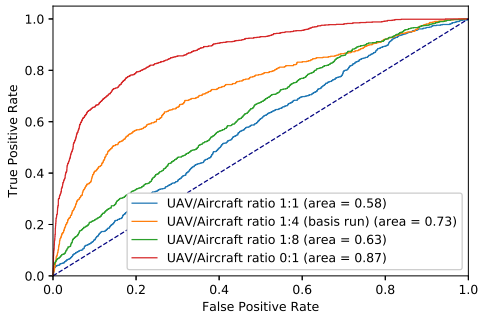
Parameter	**CRNN**
Convolution units	256
Kernel size	3x1
Pooling size	2x1
Dropout probability 1	0.25
Dropout probability 2	0.5



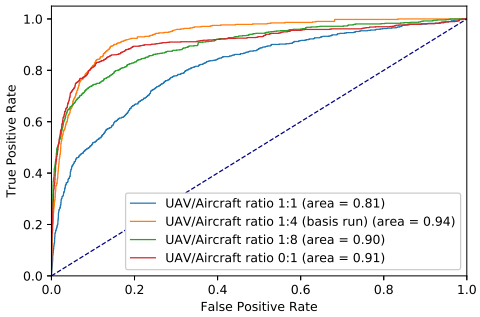
(a) MFCC-CRNN for different UAV/aircraft ratios.



(b) Mel spectrogram-CRNN for different UAV/aircraft ratios.



(c) Spectrogram-CNN for different UAV/aircraft ratios.



(d) SIF-CNN for different UAV/aircraft ratios.

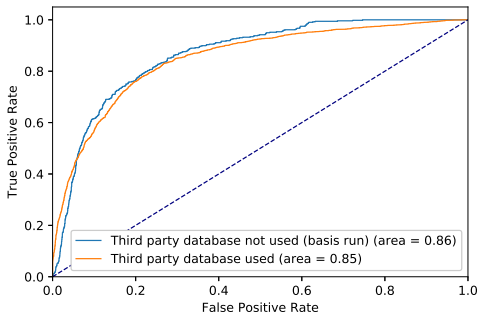
Figure E.2: ROC curves showing the influence of UAV/aircraft ratio.

than the ratio 1:8. For the SIF the expected output is completely different, having the ratio of 1:4 as most accurate, followed by a ratio of 0:1, then a ratio of 1:8 and finally the ratio of 1:1.

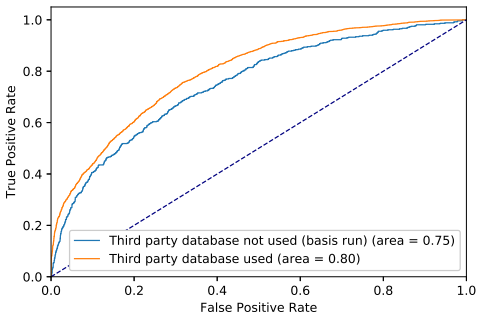
The influence of third party database recordings, which is shown in Figure E.3, neither is identical for each feature. The MFCC-CRNN is showing that third party database are only beneficial for low FPR rates ($<$ 0.1). For the spectrogram and SIF, the performance decreases and for the Mel spectrogram the performance increases when using the third party database.

Lastly, the influence of the labeling type is shown in Figure E.4. Only the influence of the labeling type shows the same trend as for the CNN models, which is that the nearby detection labeling works better than the distant detection labeling.

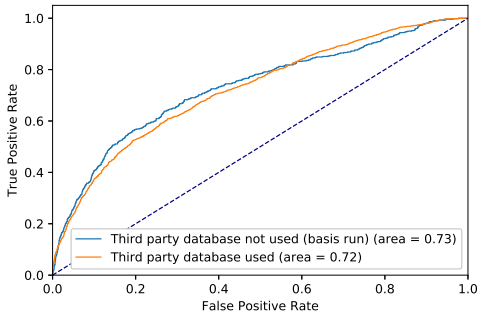
Even though CRNN could perform well in sound event recognition, for example in [16], the results presented above are not satisfactory to be really used in a real-world model. As the paper shows that increasing the amount of history used increases the performance (up until 20 seconds), other forms of (time) memory, such as a GRU (or an LSTM), should be improving the model as well. Therefore it is recommended to put further research on CRNN's for this application.



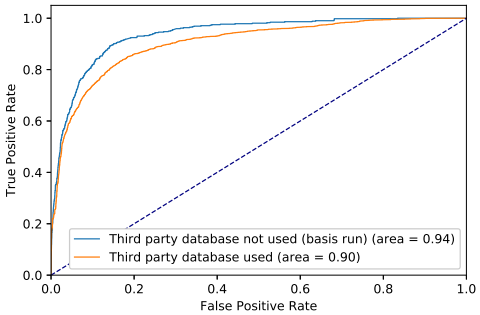
(a) MFCC-CRNN with and without third party database recordings.



(b) Mel spectrogram-CRNN with and without third party database recordings.

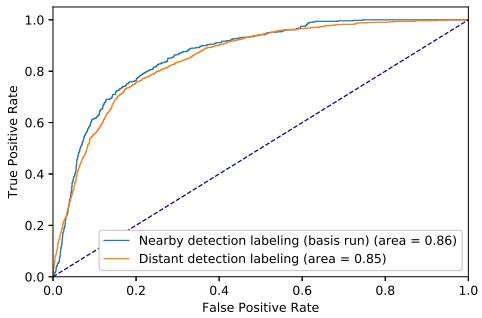


(c) Spectrogram-CNN with and without third party database recordings.

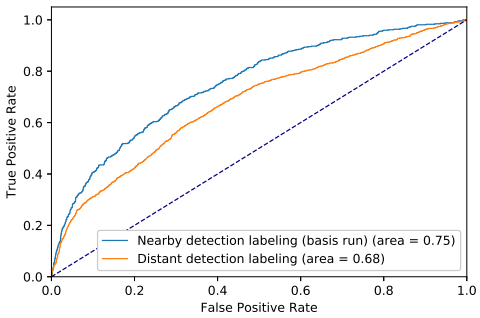


(d) SIF-CNN with and without third party database recordings.

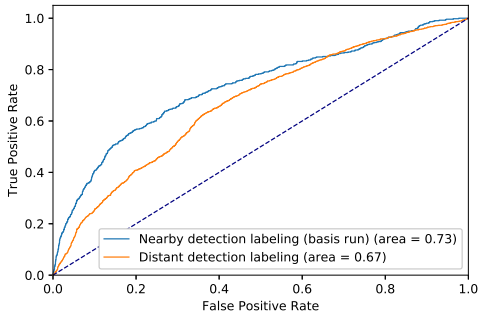
Figure E.3: ROC curves showing the influence of the third party database recordings.



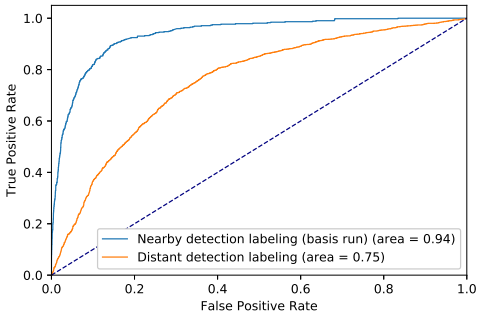
(a) MFCC-CRNN comparing the performance for different label types.



(b) Mel spectrogram-CRNN comparing the performance for different label types.



(c) Spectrogram-CNN comparing the performance for different label types.



(d) SIF-CNN comparing the performance for different label types.

Figure E.4: ROC curves showing the influence of the labeling type.

F

Model outputs

In this chapter, some extra examples are given for the outputs of the model combined with the spectrogram and the corresponding (nearby detection) label. They are sorted by how well the detection is performed. In Figures F.1 to F.4 correct outputs are shown, in Figures F.5 to F.8 partly correct outputs and in Figures F.9 to F.12 unreliable outputs.

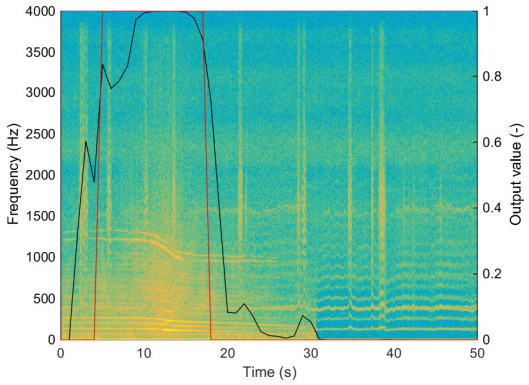


Figure F.1: Output of the Mel spectrogram-CNN (in black) and label (in red) shown on the spectrogram. It shows high output values for between 3 and 18 seconds, which match the label (except for the first second). It is also visible that when the aircraft recording stops (at 30 seconds) the output is always zero. A threshold of approximately 0.7 and higher would give a 100% accuracy for this sample.

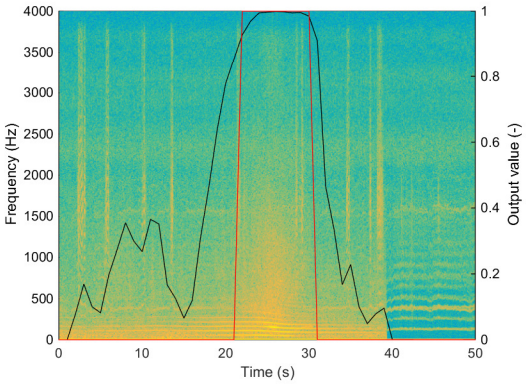


Figure F.2: Output of the Mel spectrogram-CNN (in black) and label (in red) shown on the spectrogram. Between 21 and 31 seconds the output is 1, as well as the label. Outside this time range the maximum output is 0.4, so a threshold above 0.4 would give a 100% accuracy for this sample. Also here the output is 0 when the aircraft recording has ended.

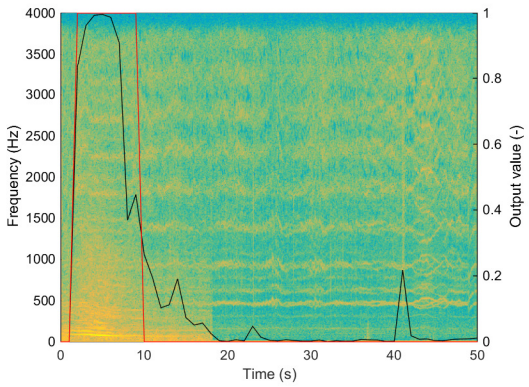


Figure F.3: Output of the Mel spectrogram-CNN (in black) and label (in red) shown on the spectrogram. Between 21 and 31 seconds the output is 1, as well as the label. Outside this time range the maximum output is 0.2, so a threshold above 0.2 would give a 100% accuracy for this sample.

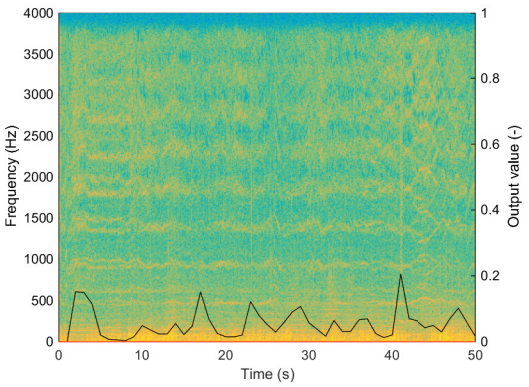


Figure F.4: Output of the Mel spectrogram-CNN (in black) and label (in red) shown on the spectrogram. For the whole sample the label is 0. The maximum output is 0.2, so a threshold above 0.2 would give 100% accuracy.

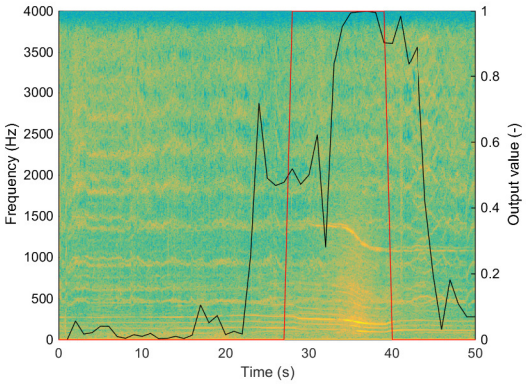


Figure F.5: Output of the Mel spectrogram-CNN (in black) and label (in red) shown on the spectrogram. For this sample, an accuracy of 100% can never be achieved for any threshold. There are outputs for which the label is 0 which are higher than the outputs for which the label is 1. However, the high outputs are located around the beginning and ending of the time range at which the label is 1 (between 28 and 40 seconds).

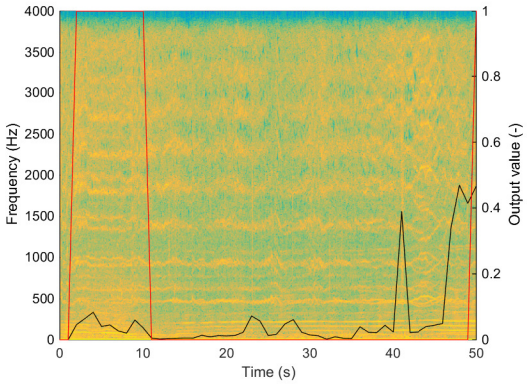


Figure F.6: Output of the Mel spectrogram-CNN (in black) and label (in red) shown on the spectrogram. The output is in general very low, with a maximum of 0.4. In the time range for which the label is 1, the output stays low. There is no threshold for which this sample achieves a 100% accuracy.

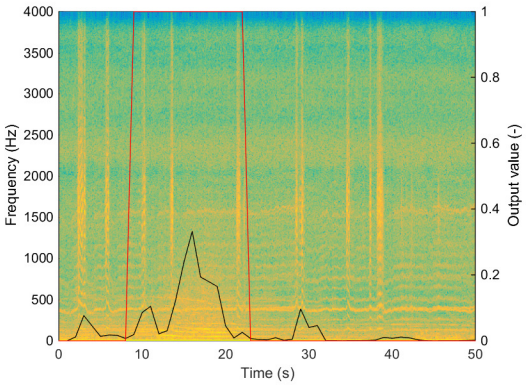


Figure F.7: Output of the Mel spectrogram-CNN (in black) and label (in red) shown on the spectrogram. Even though there is a small peak visible in the time range for which the label is 1, a 100% accuracy is never possible for any threshold. Furthermore, the peak is very low compared to those in Figures F.1, F.2 and F.3

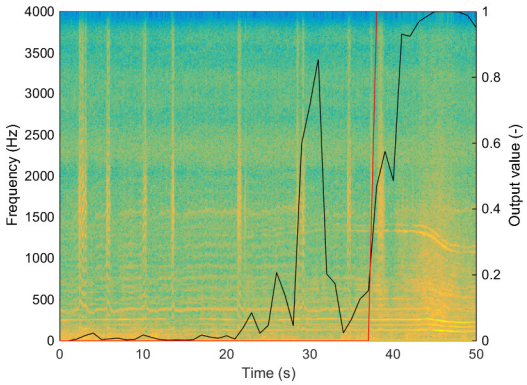


Figure F.8: Output of the Mel spectrogram-CNN (in black) and label (in red) shown on the spectrogram. Due to a large peak in the output at 30 seconds, for which the label should be 0, no threshold exist for which the accuracy is 100%. In the time range for which the label is 1, between 38 and 50 seconds, the output is also high.

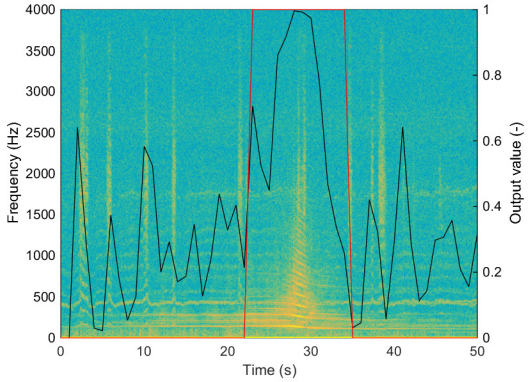


Figure F.9: Output of the Mel spectrogram-CNN (in black) and label (in red) shown on the spectrogram. In the time range for which the label is 1, the output is high. However, for the other cases, the output is irregular, as it jumps from middle high outputs to low outputs and back.

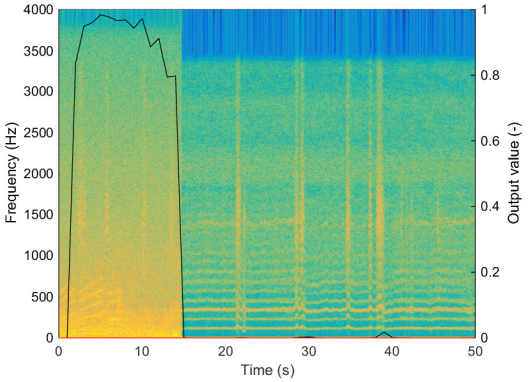


Figure F.10: Output of the Mel spectrogram-CNN (in black) and label (in red) shown on the spectrogram. For this whole sample the label is 0, but the output for the first 14 seconds are high. It falsely detects aircraft (with a high confidence level). Also here it is shown that as soon as the aircraft sound sample ends, the output goes to zero.

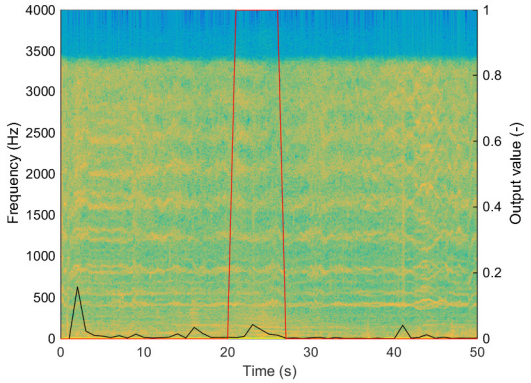


Figure F.11: Output of the Mel spectrogram-CNN (in black) and label (in red) shown on the spectrogram. The output is always low, even for the time range for which the label is 1.

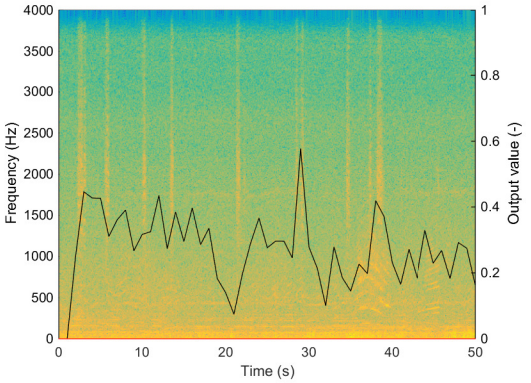


Figure F.12: Output of the Mel spectrogram-CNN (in black) and label (in red) shown on the spectrogram. The output is constantly between 0.2 and 0.6, even though the label is always 0.

III

Preliminary Report

1 Introduction

More and more UAVs are entering the air every day, both for professional as well as recreational purposes. Safety and regulations are the main topics considered nowadays in the drone industry, as UAVs form a hazard for people, other (air) traffic, buildings, etc. In the Netherlands, one of the already existing regulations is that UAVs should be located at least three kilometers from an airport and are not allowed to fly higher than 120 meters (for recreational use of UAVs)[1]. Furthermore, the drone should always be in line of sight of the operator, who can correct for possible dangerous situations. So in an ideal world, UAVs can not form any danger for other aircraft.

However, there are still situations in which UAVs and other air traffic might conflict. For example, emergency helicopters sometimes fly low in drone-permitted airspace. Moreover, due to the uprising automation of the UAVs perhaps an operator is not even needed while flying. Part of this problem can be solved by establishing (and following) good rules and laws, but also technology can help out. A project initiated by Single European Sky ATM Research (SESAR) that aims to increase air traffic safety regarding to UAVs is called Percevite[2]. Using multiple lightweight, energy-efficient sensors obstacles should be avoided to protect UAVs and their environment. One such a sensor is a microphone, which fulfills the task of 'hear-and-avoid', meaning that it should detect and avoid air traffic by sound. The goal of this research is to create a safer airspace by creating this hear-and-avoid algorithm.

In robotics, audio is already used as a source to base actions upon [17, 18]. Even though the so called 'machine hearing' is becoming more and more explored, a research that aims for 'hear-and-avoid' for UAVs has not been conducted yet. There are only three research groups that already have been trying to identify positions of other air traffic using sound, namely *Basiri et al.* [19–22], *Harvey and O'Young* [23] and *Tijs et al.* [24]. *Basiri et al.* try to determine the position of a UAV in a swarm of UAVs. The transmitting UAV sends a chirp sound in the air that has frequencies different than the UAV's ego-sound, which can be picked up quite well while flying. Also, they do tests with engines of the receiving UAV turned off and the transmitting UAVs not transmitting the chirp anymore. Also here, based on the engine sounds of the transmitting UAV its location can be determined. The hear-and-avoid algorithm can be seen as a follow up of these researches, as they have not managed to identify other air traffic by its original sound while also having the engines turned on. *Harvey and O'Young* show that with two microphones, the detection of other aircraft can be performed at such a distance that is double the distance to prevent head-on collision.

The first feasibility study for hear-and-avoid has been performed by *Tijs et al.* In this research an acoustic vector sensor is used in order to be able to detect other flying sound sources. Two co-authors, *De Bree and De Croon* [25], have used an acoustic vector sensor in order to detect sound recorded on a UAV for military purposes.

One of the reasons that there are not many researches performed on audio analysis for drones is that there are alternatives that provide traffic information, such as ADS-B, GPS, vision, etc. However, all alternatives have their down-sides and are never fully eliminating the chance of a collision. For example, ADS-B requires a system in an aircraft that is not always present or turned on. Vision based sense-and-avoid [6] often requires a lot of computational power and its images can be distorted due to speed, rain, fog, darkness, objects, etc.

[1] www.rijksoverheid.nl/onderwerpen/drone/vraag-en-antwoord/regels-drone-particulier
[2] www.percevite.org

Sound, on the other hand, is inevitable for motorized aircraft, so it is a promising method. Also, microphones are easy to use, omnidirectional, only weakly influenced by weather, lightweight and do not need a lot of (computational) power to process the data. The challenge that sound brings in this application is that many different (loud) sounds are present, such as the UAV's ego-noise, wind, air traffic and environmental sounds. The only sound of interest is the air traffic that forms danger to the drone, so the rest should be excluded.

The aim of this report is to set out the existing literature that is used for this study and to show the plan that will result in the creation of a hear-and-avoid system. This includes research in what has already been achieved in literature, as well as the proposed design of the new system and a preliminary planning. As the main phenomenon that will be worked with is sound, the fundamentals of sound are presented in chapter 2. Secondly, the structure of the algorithm is studied in chapter 3. The three modules of this system, detection, feature extraction and classification, are elaborated on in chapter 4, 5 and 6. Localization is not included in one of the modules, but is explained separately in chapter 7. Lastly, the project plan is presented in chapter 8. This plan includes the research questions, objectives and the way they are going to be performed. Finally, the conclusions of the plan are stated in chapter 9.

2

Sound

Sound is nothing more than a disturbance of air. It is energy that propagates through the air in the form of longitudinal waves [1]. Noise is exactly the same as sound, however, for noise the sound is classified as unwanted by the observer. Sound is a very useful phenomenon. It enables people to capture the environment and communicate with each other. Noise, on the other hand, is irritating and often disturbs the functionality of a person. When speaking of sound in aviation, it is usually referred to as noise as it usually is not of any use for anyone and only a burden for a lot of people. However, in this research aircraft sound will be the driving factor for a successful hear-and-avoid algorithm so it will not be referred to as noise.

In general, the main source of aircraft sounds is the propulsion system, but also the airframe, or to be more precise, it is the air around the airframe that creates a great amount of sound [26]. Due to the moving parts in the engines and the turbulent air that flows around the airframe pressure waves are created that are in the audible frequency spectrum for people. Most of the sound is created during approach due to full throttle, while for local residents most is received during departure due to the long time that an aircraft flies on a low altitude. For the most common aircraft, Eurocontrol has set up a database that stores information about the present noise during these maneuvers, the Aircraft Noise and Performance (ANP) Database [1]. In this research, the main focus will be on small aircraft and rotorcraft, as they will be the air traffic with the most probability to fly low in areas where drones are allowed, think for example about an emergency helicopter.

2.1. Sound Propagation

When a sound is created, a pressure wave p' is initiated. This oscillating wave has a (fundamental) frequency f expressed in Hertz (Hz), of which one Hz is equal to one cycle per second. A positive integer multiple of a fundamental frequency is called a harmonic. The frequency that humans hear is between 20 Hz and 20000 Hz [27], in which aircraft usually create sound in the frequency range of 50 Hz and 5000 Hz [28], so completely in the audible range of humans. From the frequency another important variable is obtained, the wavelength λ, which, as the name implies, is the travelled distance of a sound wave in a single period. It can be calculated using Equation 2.1, in which c is the speed of sound.

$$\lambda = \frac{c}{f} \tag{2.1}$$

The pressure wave oscillation can be described as a simple harmonic motion, which is shown in Figure 2.1 and expressed in Equation 2.2 [1] as a function of time t and distance r. A (in Pa) expresses the amplitude of the sound pressure at one meter distance from the source and $\omega = 2\pi f$ is the so called angular frequency.

$$p'(r,t) = \frac{A}{r}\cos\omega(t - r/c) = Re\left[\frac{A}{r}e^{i\omega(t-r/c)}\right] \tag{2.2}$$

The effective sound pressure p_e, which is the measure that is most used for amplitude, is shown in Equation 2.3. It is the root-mean-square of the sound pressure over a period T.

$$p_e = \left[\frac{1}{T}\int_0^T [p'(r,t)]^2 dt\right]^{\frac{1}{2}} = \left[\frac{1}{T}\int_0^T \left[\frac{A}{r}\cos\omega(t - r/c)\right]^2 dt\right]^{\frac{1}{2}} = \frac{A}{r\sqrt{2}} \tag{2.3}$$

[1] www.aircraftnoisemodel.org

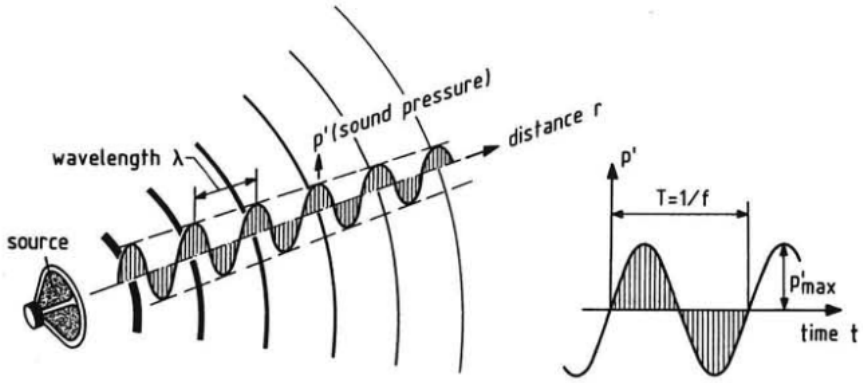


Figure 2.1: Progressive sound wave [1]

The effective sound pressure is used to capture aircraft sound. The most commonly used microphone for obtaining aircraft audio is the condenser microphone [1]. The condenser microphone has two charged parallel plates. When a sound pressure wave arrives, the outer plate, existing of a very thin diaphragm, will vibrate and causes a change in distance between the two plates.

$$C = \epsilon_o \frac{A}{d} \tag{2.4}$$

As shown by Equation 2.4, the capacitance C changes due to the change in distance d. A is the area of the plates, which stays the same and $\epsilon_0 = 8.85419E-12F/m$. With the changing capacitance an electrical current is generated which is equivalent to the sound pressure. The signal is amplified and the effective sound pressure p_e is obtained using a root-mean-square detector. When this signal is displayed, what is shown is the Sound Pressure Level (SPL) in decibel over time, such as in Equation 2.5. The reference pressure p_{e_0} is equal to 2E-5 N/m^2.

$$SPL = 10\log\frac{p_e^2}{p_{e_0}^2} \tag{2.5}$$

$$SPL(r_1) - SPL(r_2) = 20\log\frac{r_2}{r_1} \tag{2.6}$$

An important sound propagation law that is connected to Equation 2.5 is the inverse-distance law, which is expressed in equation Equation 2.6. This law describes that the sound pressure level reduces 6 dB when the distance to the sound source is doubled. When using this law, however, one must take into account that it only holds for distances that, compared with the size of the source, are large, so called far field.

2.2. Atmospheric Attenuation

During the movement of the sound pressure waves, the energy of the sound will be absorbed by the atmosphere. The energy is absorbed because in air, a small amount of internal friction occurs. The absorption of sound energy in air is called atmospheric attenuation. How much energy is absorbed per meter is indicated by means of the sound attenuation coefficient α. Multiple factors affect α, namely the frequency of the sound, the humidity and temperature of the air. This can be observed from Equation 2.7, which shows how to calculate the sound attenuation coefficient [29–31].

$$\frac{\alpha}{p_s} = \frac{F^2}{p_{s_0}}\left\{1.84\cdot 10^{-11}\left(\frac{T}{T_0}\right)^{\frac{1}{2}} + \left(\frac{T}{T_0}\right)^{\frac{-5}{2}}\left[0.01278\frac{e^{\frac{-3352}{T}}}{F_r, O + \frac{F^2}{F_r, N}}\right]\right\} \tag{2.7}$$

p_s is the atmospheric pressure, of which p_{s_0} is the reference value of $1atm$, T the atmospheric temperature, with reference value $T_0 = 293.15K$. The frequency is hidden in F, $F_{r,O}$ and $F_{r,N}$. $F = \frac{f}{p_s}$, where f is the acoustic frequency. For $F_{r,O}$ and $F_{r,N}$ the same holds, but this time $f_{r,O}$ and $f_{r,N}$ are the relaxation frequencies of molecular oxygen and nitrogen respectively. The influence of humidity h expressed in % is shown in Equation 2.8 and Equation 2.9 as it affects $F_{r,O}$ and $F_{r,N}$.

$$F_{r,O} = \frac{1}{p_{s_0}} \left(24 + 4.04 \cdot 10^4 h \frac{0.02 + h}{0.391 + h} \right) \tag{2.8}$$

$$F_{r,N} = \frac{1}{p_{s_0}} \left(\frac{T_0}{T} \right)^{\frac{1}{2}} \left(9 + 280 h \exp \left\{ -4.17 \left[\left(\frac{T_0}{T} \right)^{\frac{1}{3}} - 1 \right] \right\} \right) \tag{2.9}$$

2.3. Frequency Domain

All the previously described functions and characteristics are expressed in the time domain, which is the domain in which the sound is recorded. However, another domain that is very important for audio analysis is the frequency domain, because the sound signals of air traffic are in a huge range of frequencies. The distribution of the frequencies could give information about the type of the sound source. In the frequency domain the signal is decomposed in different frequency bands, including the phase. In order to transform the time domain signal to the frequency domain signal, a Fourier Transform (FT) is applied. The Fourier transform is based on the Fourier theorem, a theorem that explains that every periodic time function can be divided in a series of pure tones with different frequencies. Those series are shown in Equation 2.10 [32].

$$f(t) = \sum_{k=0}^{\infty} (A_k \cos 2\pi k f_1 t + B_k \sin 2\pi k f_1 t) \tag{2.10}$$

f_1 is the fundamental frequency of the sound. The Fourier coefficients are shown in Equation 2.11, Equation 2.12 and Equation 2.13, where k is an integer (k = 1, 2, 3,...).

$$A_0 = \frac{1}{T} \int_0^T f(t) dt \tag{2.11}$$

$$A_k = \frac{2}{T} \int_0^T f(t) \cos 2\pi k f_1 t dt \tag{2.12}$$

$$B_k = \frac{2}{T} \int_0^T f(t) \sin 2\pi k f_1 t dt \tag{2.13}$$

Another way to write Equation 2.10 is to use Euler identities, which is shown in Equation 2.14. C_k is called the complex Fourier spectrum coefficient, which is obtained using Equation 2.15.

$$f(t) = \sum_{k=-\infty}^{\infty} C_k e^{i2\pi k f_1 t} \tag{2.14}$$

$$C_k = \frac{1}{T} \int_0^T f(t) e^{-i2\pi k f_1 t} dt \tag{2.15}$$

These equations are applicable to harmonic noise free signals. However, most sound signals are noisy and include randomness, so called broadband time signal. Luckily also they can be evaluated by taking Equation 2.15 and set T to infinity. This causes the fundamental frequency to die out and C_k becomes a continuous frequency function, the Fourier transform, which is shown in Equation 2.16, where $p'(t)$ is the time signal. Going from the frequency domain to the time domain is also possible with the so called inverse Fourier transform of Equation 2.17.

$$C(f) = \int_{-\infty}^{\infty} p'(t) e^{-i2\pi f t} dt \tag{2.16}$$

$$p'(t) = \int_{-\infty}^{\infty} C(f) e^{i2\pi f t} df \tag{2.17}$$

Still, these functions are all continuous, which is not useful for this research as digitized sounds are discrete. Therefore, also a discrete version is created, the Discrete Fourier Transform (DFT), see Equation 2.18 [33]. k is in range from 0 to $N-1$. There is a popular algorithm to calculate the DFT, called Fast Fourier Transform (FFT) [34]. The FFT is useful as it decreases the computational complexity from $O(n^2)$ for DFT to $O(n \log n)$ for the FFT. It does so by decomposing the DFT in smaller DFTs until its small enough to be directly solved.

$$X_k = \sum_{n=0}^{N-1} x_n e^{\frac{-i2\pi kn}{N}} \tag{2.18}$$

2.4. Cepstral Domain

Some methods in audio processing also use the cepstral domain, the so called cepstrum, especially in human speech applications. A time domain signal is converted to the cepstrum by firstly taking a Fourier transform of the original domain, the take the magnitude of the resulting signal. Subsequently the logarithm of this signal is taken, followed by the inverse Fourier transform [35]. This process is showed in Equation 2.19. From this domain, four subdomains can be obtained, namely the power cepstrum, the phase cesptrum, the real cepstrum and the complex cepstrum.

$$cepstrum = F^{-1}\left\{\log\left\|F\left(f(t)\right)\right\|\right\} \tag{2.19}$$

The cepstrum shows the rate of change of the various frequency bands. It is therefore often used for pitch detection or voice identification. The independent variable that belongs to the cepstrum is called the quefrency. 'Cepstrum' and 'quefrency' both are anagrams of 'spectrum' and 'frequency' in order to highlight the relation to similar concepts [36]. There is also a third one, called 'liftering', which is 'filtering' in the cepstral domain. The quefrency is a measure of time, but not the same as time in the time domain. If a peak appears at a certain quefrency (which is an amount of samples in a defined time span), it indicates that there is a pitch present which is calculated by Equation 2.20.

$$Pitch = \frac{Sampling\ rate}{Quefrency} \tag{2.20}$$

2.5. Doppler Effect

One important sound effect that may be of huge importance in this research is the Doppler effect. It describes the change in observed frequency based on the relative movement of the emitting and receiving objects. When both objects would be in rest, Equation 2.1 describes the wavelength and source frequency relationship. However, when either of the objects move, the speed at which the sound waves arrive at the receiver change from c to $(c+dr/dt)$, resulting in a stretched or compressed wavelength λ' as in Equation 2.21. The ratio of the stretched wavelength and the original wavelength is shown in Equation 2.22. The observed frequency can be extracted from this ratio as shown in Equation 2.23.

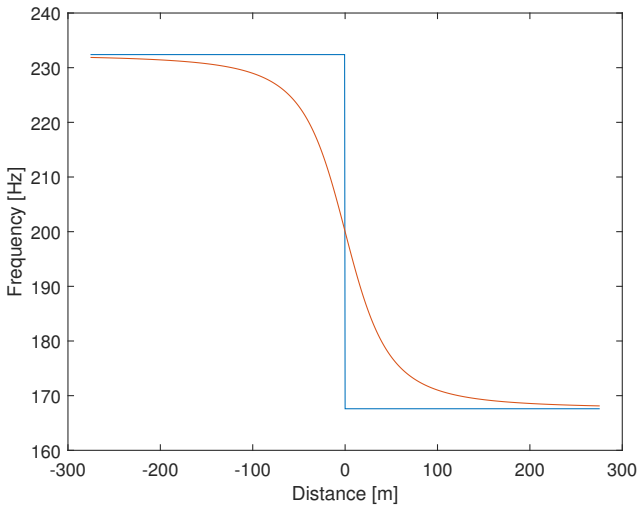


Figure 2.2: Doppler shifts for a passing source (orange) and a colliding source (blue)

$$\lambda' = \frac{c + dr/dt}{f} \tag{2.21}$$

$$\frac{\lambda'}{\lambda} = \frac{c + dr/dt}{c} \tag{2.22}$$

$$\frac{f'}{f} = \frac{c/\lambda'}{c/\lambda} = \frac{1}{1 + \frac{dr/dt}{c}} \tag{2.23}$$

Doppler shift is then defined as the difference between the observed frequency and original frequency $(f' - f)$. This shift is clearly visualized in Figure 2.2 for two different cases. The orange line represents an aircraft at an altitude 100m above the receiver and the blue line represents an aircraft at the same level as the receiver. Both pass exactly the location of the receiver. As the aircraft of the orange line is on a different altitude, it will never hit the receiver. Figure 2.2 shows that in these cases there is a smooth transition between the observed frequency. For the aircraft that goes straight through the receiver this is not the case. It has no transition and looks like a step function. This information might be useful when determining whether an object is on a collision course or not. As long as there is a sufficient transition before the shift, depending on the size of the objects, the objects will never hit each other.

2.6. Human Perception of Sound

In this research (and many others), human sound perception is the basis for (robotic) sound processing. Therefore, the human's hearing organ must be understood. As Figure 2.3 shows, the hearing organ exists out of multiple parts. Usually it is split up in three parts, namely the outer ear, the middle ear and the inner ear. The brain is the place where all the sound information is processed [37].

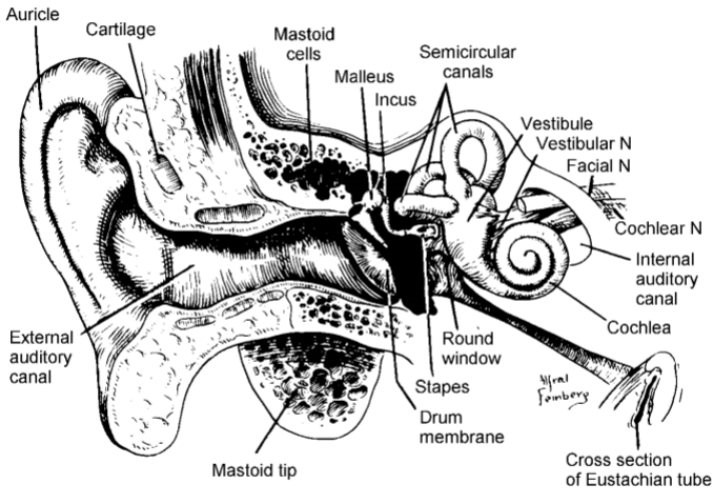


Figure 2.3: The section of the hearing organ [2]

2.6.1. The Outer Ear

The outer ear exists of the pinna and the auditory canal. The pinna has multiple functions. It is shaped such that the incoming sound is guided into the auditory canal. The reason that it sticks out is that in this way it blocks sounds from behind, which results in a better performance for localizing the sound source. The first part of the ear canal is a protective barrier consisting of wax and hairs, which is later followed by a part that has just a thin layer of skin, which has the purpose of directing the sound to the eardrum. The whole ear canal can be seen as a resonating tube that even amplifies the sound between three and four kHz. In the ear canal there is a slight bend, which prevents objects to go into the ear and damage the eardrum.

2.6.2. The Middle Ear

In the middle ear, which is also called the ear drum, the Eustachian tube, the hammer, anvil and stirrup are present. The Eustachian tube is the connection between the ear and nose, which has muscles to open up the tube and in this way equalize the air pressure between the nose and the middle ear. The hammer, anvil and stirrup (or in medical terms, malleus, incus and stapes respectively), are amplifying the sound. The complete amplification of the outer and middle ear together, is approximately 30 dB.

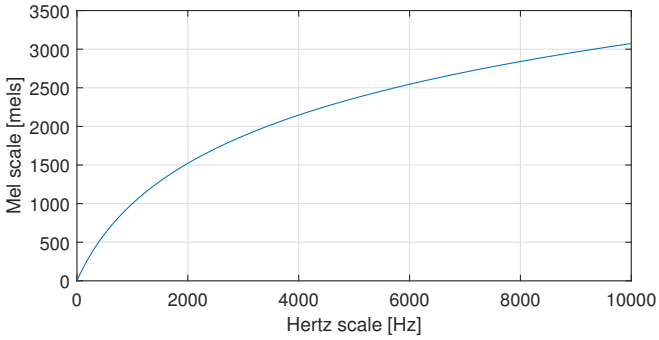


Figure 2.4: Relation between the Mel scale and the Hertz scale

2.6.3. The Inner Ear

In order to convert the sound into a signal that can be send to to the brains, the cochlea exists. Together with the vestibular labyrinth, which is the organ that takes care of our balance, the inner ear is formed. The cochlea is a twisted tube that includes many small hairs. The membrane inside the cochlea has one remarkable property: when it is carrying a wave, each part of the wave that carries a specific frequency travels until it is at the point where it resonates for that frequency. After that, the frequency component of the wave does not travel further. When a wave is traveling the membrane, cilia of hair cells are changing their position relative to the cells body, by which ion passages in the cell are open or closed, which in their turn stimulate the nerve ending. There are two types of nerve fibres in the cochlea, afferent fibres and efferent fibres. The afferent fibres take pulses from the cochlea and bring them to the brain and the efferent fibres do the opposite. This creates an active feedback loop for the ear. The frequency is perceived by humans in the Mel scale. This scale, which is presented in Equation 2.24 and Figure 2.4, shows that doubling the frequency is not perceived as a pitch that is two times as high. It is defined such that 1000 mels are equal to 1000 Hz. The name Mel is derived from the word melody as it is based on pitch comparison.

$$m = 2595 \log_{10}\left(1 + \frac{f}{700}\right) \tag{2.24}$$

2.6.4. Central Auditory Processing

With the information that is send to the brain, many different functions can be executed. The most used ones are explained in this section.

Cocktail Party Effect First of all, there is the well explored cocktail party effect [38, 39]. People with a healthy hearing system can determine, among a crowded place, which conversation they want to focus on, filtering out the other conversations. This is made possible by the brain focusing on signals with a specific time and intensity difference of arrival.

Sound Source Localization Humans are capable of estimating where the sound comes from. This is due to the intensity difference and time difference of arrival. The head is also of importance here as it is blocking part of the sound for one ear, just like the pinna. The shape of the pinna blocks the sounds from behind which is used for the spatial identification. Due to the small distance between the two ears, mostly the high frequency sounds are important for the localization. This will be further explained in chapter 7.

Ignore Sounds The brain can get used to a certain sound and filters it out so that it is not constantly heard. Still, it does notice when the sound is not there anymore, which is also experienced by the human.

Interaction with Other Parts of the Brain The presence of certain sounds is linked with a quick action of other parts of the brain, mostly in alarm signals. It affects the heart rate and tension in the muscles, which are preparations for a quick response. Also sounds can evoke certain emotions, especially in music.

3

Sound Event Recognition (SER)

There are various application areas of audio processing existing nowadays. The most popular ones are for example Automatic Speech Recognition (ASR) and Music Information Retrieval (MIR). Even though the various application areas have an overlap in their methods, they are usually all used for a specific type of audio processing. In Sound Event Recognition (SER), the main goal is to understand (parts of the) surrounding acoustic scene [3]. This includes a wide variety of subjects, from bird species identification [40] to detection of sounds in a meeting room [41].

3.1. Overview

A sound event can be described as a unique sound that comes from one physical object [42]. Examples are the ring of a phone, footsteps, a car passing by, etc. However, the definition of 'a single source' is very subjective, as the sound of a car could also be separated in the sound of the wheels and the engine. The most important properties the sound event are the duration and characteristic frequency content. The sound events hold cues and important information in them, which can be used for analysis of the environment. In this research, the sound events of interest will be the approach of other air traffic towards the aircraft. The difficulty here is that the sounds are similar to the ego-sound of the drone, so it will be hard to separate them.

Sound events, including aircraft sound, have very different characteristics compared to speech or music, which is why sound event recognition usually takes a different approach compared to ASR and MIR. The environment where all the sound events are occurring differ, as well as the type of interaction that is causing the sound event. A comparison between the four forms of audio is shown in Table 3.1 that compares the number of classes, window length, bandwidth, harmonics and repetitive structure. This shows that for sound events it is hard to define any of the five acoustical characteristics beforehand, which would be possible with speech or music. For aircraft sound especially it shows little correlation with voice and music and therefore another approach to this kind of sound processing is required.

Just as audio processing can be divided in various application areas, so can SER be divided in various categories. Broadly speaking are there three different categories, namely environmental sound event recognition, acoustic surveillance and environment classification.

Environmental Sound Event Recognition In this area, a certain environment is chosen in which a subset of sounds can occur. Depending on which environment is used, the recognition problem is defined. Note that speech is not included as sound event, even though it is present in a particular environment. For speech, ASR is applied. An example of environmental sound event recognition is the environment of healthcare. Papers [43, 44] describe how sound is used to count the amount of coughs per time unit, and to recognize people falling from stairs, collapse or screaming.

Acoustic Surveillance Instead of using security cameras to check the surroundings, also sound can be used for detection of prohibited occurrences. Another option would be to make it complementary to the already existing security cameras. Various types of surveillance have been performed with audio, such as aggressive sound event detection [45, 46] or office environment monitoring [47]. Also for biological monitoring acoustic

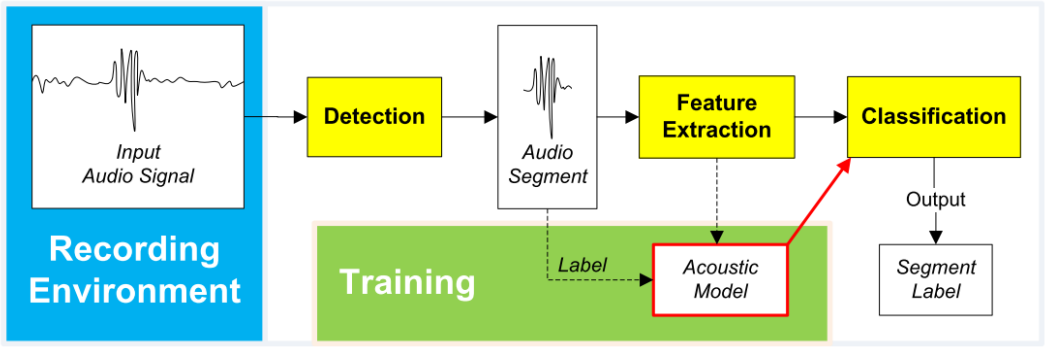


Figure 3.1: Common structure of a SER system [3]

surveillance is relevant, such as detecting birds [48] or recognition of other animal sounds [49]. Acoustic surveillance is also the category that is most applicable to hear-and-avoid.

Environment Classification In environment classification information is obtained about the current surroundings of the recording device. The use of this is that the settings of a device can be adjusted to whichever situation it is in. Think for example of a phone, that should not ring out loud in a meeting, or hearing aid systems, that can use the environmental information for the automatic tuning of the audio compression, so that the user always experiences the optimal sound information [18, 50].

Table 3.1: Sound category characteristics [7]

Acoustical Characteristic	Voice	Music	Sound Event	Aircraft sound
Number of classes	Number of phonemes	Number of tones	Undefined	Undefined
Window length	Short	Long	Undefined	Long
Bandwidth	Narrow	Relatively narrow	Broad Narrow	Broad
Harmonics	Clear	Clear	Clear Unclear	Unclear
Repetitive structure	Weak	Weak	Strong Weak	Strong

3.2. Structure of a SER System

When a SER system is used for audio processing, usually the following three modules are present: the detection module, the feature extraction module and the classification module. Such a typical system is presented in Figure 3.1. This setup is also used in this report, where the detection is described in chapter 4, feature extraction in chapter 5 and classification in chapter 6. A broad overview of each of them is given below.

3.2.1. Detection

In sound event recognition, detection is the process that catches the sound segments of interest, in this case the segments that include aircraft sound. The expected sounds present are the UAV's ego-sound, wind, microphone noise, environmental sounds, traffic, etc. There are various types of detection systems, detection-by-classification and detection-and-classification. An example of the latter is the novelty-detection system, which finds sound events by considering rapid changes in the recording over the long-term background noise [45]. As detection-by-classification the sliding window detector is popular, that actually classifies the present

sounds of a fixed-length sound segment [51]. In the latter procedure detection and classification are actually combined instead of being two separate modules, therefore called detection-by-classification. Detection is further elaborated on in chapter 4.

3.2.2. Feature Extraction

Features in audio signals are characteristics of that audio signal which hold information that is used for easy discrimination between classes. The features compress the audio signal so that it is cheaper to process. Preferably, the features are not influenced by sound occurrences that are not important. Many audio features exists, which are presented in [52] and [8]. In this research four main domains will be used in which the different features can be found: time domain, frequency domain, cepstral domain and image domain. The time domain is the domain in which the audio is recorded. Based on power, amplitude or zero-crossing multiple features exist. If a Fourier transform is taken of the the time domain signal the frequency-domain signal is obtained. The frequency domain can be split up in two, a perceptual frequency domain, consisting of features such as pitch, which is the perceived frequency, chroma, which describes the pitch based on the 12 pitch classes (A-G) that are used in music transcription, and brightness, which is a description of the amount of high-frequency components in a sound. Secondly there is the physical frequency domain, which is the domain where the mathematical frequency characteristics are described. Examples of features in this domain are spectral flux and energy ratio, which are capturing the changes in frequency energy distribution and describes the energy distribution of the spectrum respectively. The cepstral domain is obtained when the inverse Fourier transform of the log-magnitude of the frequency domain is taken, as Equation 2.19 shows. Doing so, the sound is transformed to a way how humans perceive it. Lately, there is a relatively new domain that has promising results and might be of good use in this application, which is the image domain. This domain utilizes characteristics of spectrograms. The interesting features for SER are described and discussed in chapter 5.

3.2.3. Classification

During classification, the audio segment is given a label that says to which predefined category it belongs to based on the training that it has done before. There are numerous techniques for classification, such as k-Nearest Neighbours (kNN), Dynamic Time Warping (DTW), Gaussian Mixture Models (GMM), Hidden Markov Model (HMM), Support Vector Machines (SVM) and Artificial Neural Network (ANN). The latter four techniques will be further elaborated on in chapter 6. kNN and DTW follow a basic approach, in which the training features are stored in a database. The classification will be performed based on the distance between the database features and the features observed during testing. The others use a more preferred approach which is in general computationally or memory-wise less expensive. For these methods a model of the feature vector space is created during training, which is again compared during testing with the observed features, based on distance from the original model.

4

Detection

The main goal of detection in a SER system is the extraction of a specific type of sound out of an audio recording. In the hear-and-avoid case, the drone will continuously record audio data, of which most of its data does not contain valuable information. That is, the drone's ego-sound and noise such as wind is not particularly of interest. However, it is a must to be able to capture audio events that are caused by other air traffic, which is a difficult task since the sound is of the same nature as the drone sound. Detection can be either a singular module in the SER system, which is called detection-and-classification, or one can detect while classifying, namely detection-by-classification. Both are elaborated on in section 4.1 and 4.2 respectively.

4.1. Detection-and-Classification

For the detection-and-classification approach the detection module is separated from the feature extraction module and the classification module and mainly serves to find a sound event in a continuous sound recording. It usually builds on low-level audio features, such as pitch, Zero-Crossing Rate (ZCR), spectral divergence, [3, 53] (see chapter 5). The audio segment is not further processed by the detection module, but is passed on to the feature extraction module. The process of finding blocks of audio that can be discriminated from the background sound is called novelty detection.

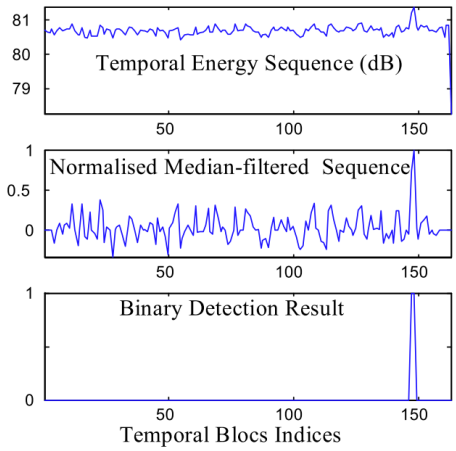


Figure 4.1: Impulse detection in a noisy environment [4]. In the upper plot the energy is expressed over time. The middle plot shows the same sequence which has been median-filtered and normalized. The binary detection is plotted in the lower plot.

[4] describes a novelty detection mechanism that works in noisy environments, which is why this method is taken as an example for the detection for this system. The detection module analyzes the energy variations of the signal using a non-linear median filter. First the signal's energy is obtained for blocks of 100ms, onto which a median filter is applied. As the median is taken per block of 100ms, only impulses are detected in this detector. The filtered signal is subtracted from the original energy, so that the relevant energy pulses are

emphasized. Lastly an adaptive threshold is used to come to a binary detection. The threshold depends on the standard deviation of the long term energy sequence. A visualization of this method is shown in Figure 4.1. In this research the definition of long-term should be chosen such that the aircraft's sound will not be filtered out eventually, even though it might be present in the sound recording for a longer time.

The biggest advantage of the detection-and-classification method is that there is no need of choosing a fixed segment length in advance, as a sliding window determines whether subsequent windows belong to the same sound event (if any) [3, 51]. This is useful because the time of presence of air traffic can vary. Furthermore it can be used real time with low computational cost. The difficulty for this method lies in in finding the right threshold.

4.2. Detection-by-Classification

In detection-by-classification a time-window segments the audio file, of which each segment is used for classification. The classifier determines whether the sound is background noise or a sound event of interest. A detection-by-classification framework is therefore actually equal to the SER system without the detection module [54]. The main advantage is that for the complete SER system the features only have to be obtained once. However, for this research only the sounds that are detected need to be classified. Therefore it might be more efficient to actually do the detection separately, such that it is not continuously classifying the recording. Secondly, a fixed window of sound might negatively influence the response time in a possible hazardous situation. Taking to a time window that is too short might not capture the Doppler shift (which might be useful for hazard detection), a time window that is too long might result in a situation where there is no time to react on the other traffic. Another disadvantage for the sliding window is that the sound events should be known beforehand [3, 55]. Due to the high in-class variance (different type of aircraft/helicopters, different speed, different altitude, different thrust, etc.) this is very hard to obtain. Additionally, the nature of the sound classes that are not important, such as ground traffic sounds, are equivalent to the sound classes that are important, such as the drone, aircraft and helicopter sound, which might be a disruptive factor.

4.3. Experiments

Table 4.1: Architecture of the preliminary detection module

Layer	**Parameters**
Input layer	
Convolutional layer	Filters: 32 Kernel size: 3 Activation function: Relu
Convolutional layer	Filters: 32 Kernel size: 3 Activation function: Relu
Max Pooling layer	Pool size: 2
Dropout layer	Dropout probability: 0.25
Convolutional layer	Filters: 64 Kernel size: 3 Activation function: Relu
Convolutional layer	Filters: 64 Kernel size: 3 Activation function: Relu
Max Pooling layer	Pool size: 2
Dropout layer	Dropout probability: 0.25
Flatten layer	
Fully connected layer	Units: 32
Dropout layer	Dropout probability: 0.5
Fully connected layer	Units: 4

A simple detection(-by-classification) module has been built. In order to understand the features and classifier, please refer to chapter 5 and chapter 6. Drone recordings and aircraft recordings that have been obtained

from free sound databases have been combined to form four types of recordings: solely drone sound or drone sound combined with either propeller aircraft, jet aircraft or helicopter sound. Each recording is seven seconds long and is sampled at a sample rate of 22.5 kHz. Before combining the drone and aircraft recordings, they have been normalized and added up, of which the drone sound was twice as loud as the aircraft sound. Four features have been chosen based on the availability in the Python library Librosa [56], which are the spectrogram, melspectrogram, MFCC and ZCR. These features are elaborated on in chapter 5, of which the melspectrogram is a spectrogram in the mel scale.

The convolutional neural network (see subsection 6.4.2) consists of two times two layers of convolutional layers followed by a max pooling layer and a dropout layer. After these eight layers the network is flattened, and followed by two fully connected layers with a dropout layer in between. The network tries to classify the sound, so there are four possibilities as an output. This architecture is obtained from a Keras tutorial[1]. An overview of the architecture and the corresponding parameters are shown in Table 4.1.

In order to train the network, 100 epochs are performed with the Adam optimizer [57] as optimization function. The data, consisting of 400 samples, has been split in a training set (70%) and a test set (30%). The test set data is unique compared to the training set data, even though the same drone and aircraft sounds are used. The results that are obtained for each feature are presented in Table 4.2. The results are split up in two. There is the classification accuracy, which is how well the network could distinguish between the four classes, and the detection accuracy, which is how well the network could distinguish between only drone sound and drone sound in combination with other air traffic. The detection accuracy is expressed as the positive predictive value (also called precision) of the drone label. This value tells how much percent of the sound data predicted as solely drone sound was actually solely drone sound. If it is lower than 100 percent, it means that a few aircraft/helicopter sound segments have not been noticed by the network, which is a dangerous situation in case of the hear-and-avoid situation. The confusion matrices for each feature are presented in Table 4.3, 4.4, 4.5 and 4.6 to confirm the detection accuracy and to show the division of true labels and predicted labels. The melspectrogram is for both types of accuracy the best features to use.

Table 4.2: Results of the preliminary detector module using a convolutional neural network

Input feature	MFCC	Spectrogram	Melspectrogram	ZCR
Detection accuracy (%)	92.0	74.0	87.0	72.0
Classification accuracy (%)	75.6	65.7	76.8	51.5

Table 4.3: Confusion matrix of the detection-by-classification test using the MFCC as the input

		Predicted label			
		Drone	Drone + helicopter	Drone + proppeler aircraft	Drone + jet aircraft
True label	Drone	24	3	1	0
	Drone + helicopter	0	27	3	5
	Drone + propeller aircraft	1	1	10	3
	Drone + jet aircraft	1	2	4	14

Table 4.4: Confusion matrix of the detection-by-classification test using the spectrogram as the input

		Predicted label			
		Drone	Drone + helicopter	Drone + proppeler aircraft	Drone + jet aircraft
True label	Drone	23	4	0	1
	Drone + helicopter	4	21	6	4
	Drone + propeller aircraft	0	2	7	6
	Drone + jet aircraft	4	0	3	14

[1] `cambridgespark.com/content/tutorials/convolutional-neural-networks-with-keras/index.html`

Table 4.5: Confusion matrix of the detection-by-classification test using the melspectrogram as the input

		Predicted label			
		Drone	Drone + helicopter	Drone + proppeler aircraft	Drone + jet aircraft
True label	Drone	27	1	0	0
	Drone + helicopter	3	24	3	5
	Drone + propeller aircraft	0	1	9	5
	Drone + jet aircraft	1	3	1	16

Table 4.6: Confusion matrix of the detection-by-classification test using the zero-crossing rate as the input

		Predicted label			
		Drone	Drone + helicopter	Drone + proppeler aircraft	Drone + jet aircraft
True label	Drone	21	4	1	2
	Drone + helicopter	5	13	9	8
	Drone + propeller aircraft	0	3	10	2
	Drone + jet aircraft	3	4	7	7

There are a few notes to be made on this preliminary detection network. First of all, the data set is very small (400 files). This means that there is very little generalization, so when a new data file is presented it may not be able to classify or detect it that well. Secondly, for the ZCR feature, which is a one dimensional vector, the same convolutional network has been used as for the other (two dimensional) features. A better performance could be achieved by designing for a one dimensional case as well. Lastly, the relative loudness between the drone and the aircraft is set to be 2:1 when combining the files, however, this ratio is just an estimation. Another test is performed in order to find the influence of the ratio on the classification and detection accuracy. This test used more aircraft and helicopter sounds and the drone sound was recorded on top of the drone. Only one feature has been used to check the influence, which was the melspectrogram. The results of this test, which are shown in Table 4.7, prove that network performs worse and worse when the drone sound gets louder relative to the aircraft/helicopter sound. The detection accuracy is increased relative to the first test up until a ratio of 13:1. The reason for an improvement in detection accuracy is the use of a better data set. Also three confusion matrices are presented in Table 4.8, 4.9 and 4.10. As until now it is unclear at which ratio real life recordings actually happen, the data set must come from recordings on a drone with other air traffic present or experiments have to be performed to obtain this ratio.

Table 4.7: Detection and classification accuracy for the detection-by-classification test using the melspectrogram with different drone to aircraft loudness ratios

Ratio drone sound : aircraft sound	Detection accuracy (%)	Classification accuracy (%)
1:1	97	97
3:1	100	87
5:1	100	71
7:1	100	58
10:1	100	55
12:1	100	49
13:1	100	47
14:1	18	18

Table 4.8: Confusion matrix of the detection-by-classification test using the melspectrogram with a drone to aircraft loudness ratio of 3:1

		Predicted label			
		Drone	Drone + helicopter	Drone + proppeler aircraft	Drone + jet aircraft
True label	Drone	23	0	0	0
	Drone + helicopter	0	26	0	4
	Drone + propeller aircraft	0	0	26	0
	Drone + jet aircraft	0	4	6	19

Table 4.9: Confusion matrix of the detection-by-classification test using the melspectrogram with a drone to aircraft loudness ratio of 12:1

		Predicted label			
		Drone	Drone + helicopter	Drone + proppeler aircraft	Drone + jet aircraft
True label	Drone	27	0	0	0
	Drone + helicopter	0	0	0	27
	Drone + propeller aircraft	0	0	0	29
	Drone + jet aircraft	0	0	0	26

Table 4.10: Confusion matrix of the detection-by-classification test using the melspectrogram with a drone to aircraft loudness ratio of 14:1

		Predicted label			
		Drone	Drone + helicopter	Drone + proppeler aircraft	Drone + jet aircraft
True label	Drone	16	0	0	0
	Drone + helicopter	28	0	0	0
	Drone + propeller aircraft	20	0	0	0
	Drone + jet aircraft	25	0	0	0

4.4. Detection Selection

Sections 4.1 and 4.2 present the advantages and disadvantages of both methods. Detection-and-classification has the advantage that the window length is not fixed, so it is applicable to sounds with various lengths. Moreover, this method can be used real-time due to the low computational cost. The detection-by-classification has to have a fixed window, but is still often given preference to because of its natural simplicity. Also, a preliminary test with this type of detector already shows promising results. As at this stage it is unclear which detector will work best for the hear-and-avoid system, in the research both will be developed and compared on their performance.

5

Feature Extraction

Audio feature extraction is the process that finds the correct parameters of the signal that can describe the recording's most important traits. The choice of which features are going to be extracted is of major importance in the SER system, as it can make or break the performance of the classification. The optimal feature describes the sound accurately while still being compact and while pointing out the best characteristics for the classification task. There exists plentiful different features already, which are summarized in [52] and [8]. For each domain, the existing features will be described, as well as their usefulness for this research. In Table 5.1 a detailed taxonomy of features is shown. For each domain the relevant features are elaborated on, for both the perceptual as the physical features. The difference between the two is that the perceptual features are properties known by human listeners, such as pitch and brightness. The physical features focus more on the mathematical aspects of the audio waveform. Note that the features described below are only the features that are useful in the category of environmental sounds, as speech and music features are not the feature of interest for this research. In Python, the library 'Librosa' is a useful tool to obtain certain features.

5.1. Time Domain Features

The benefit of the time domain is that the audio signal does not require any form of transformation before it is processed. It is therefore one of the most elementary domains of feature extraction. The time domain is split up in four categories: zero-crossing rate-, amplitude-, power- and rhythm-based features.

5.1.1. Zero-Crossing Rate-Based Features

The name of this feature set already comprises the way the feature extraction works: it counts the rate in which the signal crosses the zero line in one second. The Zero-Crossing Rate (ZCR) therefore estimates roughly the dominant frequency component in the signal [58]. It is used, among others, for auditory scene classification [59], environmental sound recognition [60] and audio surveillance [61].

5.1.2. Amplitude-Based Features

In this set of features, the envelope of the time-domain signal is the part of interest. The benefit of this feature set is that it is computationally inexpensive.

Amplitude descriptor (AD) The main application of the amplitude descriptor is for animal sound recognition [62]. The feature discriminates between high and low amplitude signal segments, using an adaptive threshold. The descriptor holds information about the duration, energy and variation of the duration. This feature is most helpful for the separation of sounds that have a very characteristic sound envelope.

MPEG-7 audio waveform (AW) In order to obtain this feature, the signal's waveform envelope is downsampled, after which the extreme values are taken inside non-overlapping frames, such as shown in Figure 5.1. It is mostly used to show the difference of waveforms, which finds its application in environmental sound recognition [63]. The MPEG-7 is an audio descriptor developed that is used for obtaining audio information such as intensity, pitch and timbre [64].

Table 5.1: Taxonomy of feature extraction techniques [8]

	Pysical features	**Perceptual features**
Time domain	Zero-crossing rate-based Amplitude-based Power-based Rhythm-based	Zero-crossing rate-based Perceptual autocorrelation-based Rhythm-based
Frequency domain	Autoregression-based Short-Time Fourier Transform-based Brightness-related Tonality-related Chroma-related Spectrum shape-related	Modulation-based Brightness-related Tonality-related Loudness-related Roughness-related
Wavelet domain	Wavelet-based direct approaches Hurst parameter features MP-based Gabor features Spectral decomposition Sparse coding tensor representation	Kerner Power Flow Orientation Coefficients Mel Frequency Discrete Wavelet Coefficients Gammatone Wavelets Perceptual Wavelet Packets Gabor functions
Image domain	Spectogram image features	Auditory image model Stabalized auditory image Time-chroma images
Cepstral domain	Linear Prediction Cepstrum Coefficients	Perceptual filter banks-based Autoregression-based
Multiscale spectro-temporal domain		Multiscale spectro-temporal modulations Computational models for Auditory receptive fields
Other domains	Eigenspace-based Phase space-based Acoustic environment-based	Eigenspace-based Electroencephalogram-based Auditory saliency map

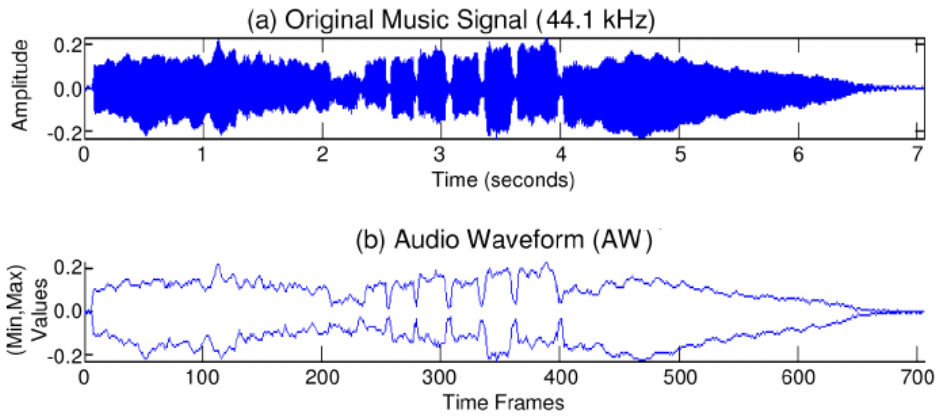


Figure 5.1: An MPEG-7 audio waveform (b) extracted from an audio signal (a) [5]

5.1.3. Power-Based Features

The power of an audio signal is defined as the energy of the signal transmitted per second. The energy is obtained by taking the square of the amplitude. In other words, another way to describe the power is by taking the mean-square of the waveform. Also sometimes the root-mean-square is taken as a feature.

Short-Time Energy (STE) STE is a feature that can be performed both in the time-domain as well as the frequency domain. It describes the average energy per time-window of a signal. It is used in environmental sound recognition in [59, 63, 65].

Volume The volume, which would be loudness in human perception, is the root-mean-square of the signal in one frame. It is linked with STE, by taking the square root.

MPEG-7 Temporal Centroid As the name suggests, this is a time average of the signal's envelope, which describes where most of the signal's energy is stored in time (on average). Also this feature is used in environmental sound recognition [63, 65].

MPEG-7 Log Attack Time (LAT) The MPEG-7 LAT feature shows the attack of a particular sound. The attack is computed by the logarithm of the time between the start of a sound signal and the initial local maximum. Again, the feature is used in [63, 65] for environmental sound recognition.

5.1.4. Rhythm-Based Features

Having information about the rhythm of the signal can be very useful. For example based on rhythm music identification can be performed. But also in this research there are rhythmic elements present, such as the rotation of the rotors. However, there are no researches performed yet that use rhythm features for sound events recognition regarding drones or aircraft.

5.2. Frequency Domain Features

The frequency-domain is the domain in which most of the features are present. In order to come to the frequency domain, the Fourier transform of the time-domain signal is taken.

5.2.1. Autoregression-Based Features

The technique used with autoregression is that there is a linear predictor, which estimates the next sample value based on a linear combination of the preceding sample values as shown in Equation 5.1. $\hat{x}(n)$ is the predicted value, a_i is the feature vector with the prediction coefficients and $x(n-i)$ the previous observed values. The prediction coefficients are obtained by minimizing the mean squared error between the actual sound segment and the predicted sound segment. One type of autoregression-based feature which is used in environmental sound recognition [66] is Code Exited Linear Prediction (CELP). It combines, among others, Linear Spectral Pair (LSP)[67] and pitch.

$$\hat{x}(n) = \sum_{i=1}^{p} a_i x(n-i) \tag{5.1}$$

5.2.2. STFT-Based Features

The Short-Time Fourier Transform (STFT) is a combination of Fourier transforms over multiple windows of time. In general these features either look at the envelope of the spectrogram or the phase of the STFT. The latter, however, has not been applied to sound event recognition yet.

Subband Energy Ratio The energy distribution of the signal's spectrum is approximated with this feature. The set of predefined frequency bands, called subbands, are variable in number and in their characteristics during design. With the latter is meant that also, for example, Mel scale is used for the subbands. For environmental sound recognition it is used in [59].

Spectral Flux Spectral Flux (SF) describes, as the name suggests, the quick changes in frequency energy distribution of the signal and therefore also in the power spectrum. This feature also includes timbral information, which is the information of sound that separates tones of equal pitch from sounding different. Again this is used in [59].

5.2.3. Brightness-Related Features

Brightness is described as the balance between the high and low frequencies in the energy of the signal. The more high frequency content is present, the brighter the sound. One measure for this is the Spectral Centroid (SC). It finds the first moment, which is equal to the frequency at the mean value of the spectral energy. This frequency is the predominant frequency. It is a popular feature in all kinds of audio classification. For SER applications it is used in [59, 63, 65].

5.2.4. Spectrum Shape-Related Features

Not only in the time domain the shape of the waveform can be used as a feature, also in the frequency domain this is the case.

Bandwidth Usually the bandwidth is obtained when taking the signal spectrum's second-order statistic, which separates low bandwidth sound from high bandwidth sounds. High bandwidth sounds are usually noise sounds, which is why this is usually a feature used in noise environments. Also this feature is used in [59, 63, 65].

Spectral Roll-Off Point [59, 61] use the spectral roll-off point as a feature. This feature shows the spectral shape's skewness, by taking the 95th percentile of the distribution of the power spectrum.

Spectral Flatness The spectral flatness shows how noisy-like a signal is, by estimating the uniformity of frequencies in the power spectrum. Mathematically it is defined as the ratio between two means, the geometric and the arithmetic mean. Noise has usually a higher flatness than tonal sounds. This feature is applied in [63, 65].

Subband Spectral Flux The Subband Spectral Flux (SSF) is inverse proportional to the SF. It has more relevance for signals where the frequency content is non-uniform. This feature has been especially introduced for environmental sound recognition [60]. It measures the part of prominent partials in each subband. Per subband the differences between neighbouring frequencies are accumulated, which result in the SSF.

5.3. Cepstral Domain

The cepstral domain is a smoothed log magnitude spectrum.The original cepstral domain was obtained as follows [68], which is also mathematically shown in Equation 2.19:

$$signal \rightarrow FT \rightarrow mag \rightarrow log \rightarrow IFT \rightarrow cepstrum \tag{5.2}$$

The reasons behind this order of steps is as follows. The function of the cepstral domain is that it holds pitch and timbral information. The pitch information is stored in the spectral envelope and the timbral information in the spectral details. As Figure 5.2 shows, the log magnitude of the original signal (a) is the sum of the log magnitude of the spectral envelope (b) and the log magnitude of the spectral details (c). So starting from the original signal, in order to obtain the spectrum of the signal such as shown in Figure 5.2, the Fourier transform is taken to come to the frequency domain, the magnitude to calculate the power per frequency and the log for scaling. Taking the inverse Fourier transform then finds the pitch and the timbre over time.

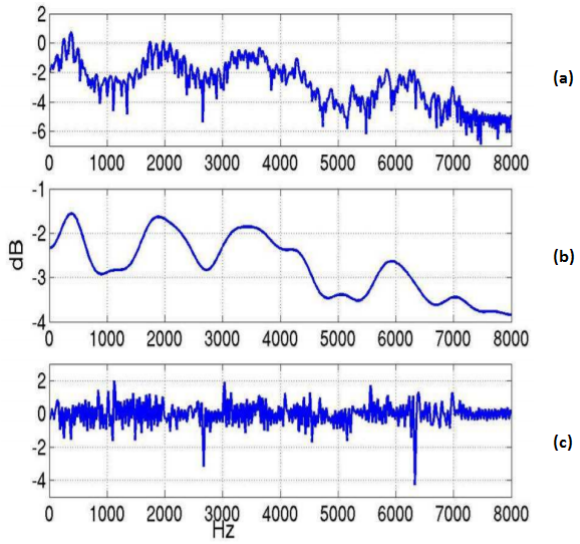


Figure 5.2: Separation of the the log magnitude of the original signal (a), the sum of the log magnitude of the spectral envelope (b) and the log magnitude of the spectral details (c)[1]

Linear Prediction Cepstrum Coefficient [59] and [18] use the Linear Prediction Cepstrum Coefficients (LPCC) for environmental sound recognition. The LPCCs are obtained by using Equation 2.19 over the output of an autoregressive filter of the signal. The benefits of this feature is that it is compact and noise robust.

Mel Frequency Cepstral Coefficients These Mel Frequency Cepstral Coefficients (MFCC) are based on the human auditory model. It is obtained by putting the output of a Fourier Transform through a Mel-scale filter bank, which are subsequently logarithmized and decorrelated using a DCT. With MFCCs timbral information is obtained using the first DCT coefficients. Even though MFCCs are mostly applied for speech recognition, also for audio surveillance and environmental sound classification it is of good use [59, 61, 69].

5.4. Image Domain Features

The image domain features typically take a spectrogram and use the image for determination of the feature. A spectrogram is a diagram where the energy per frequency (band) is plotted against the time, such as in Figure 5.3.

Spectrogram Image Features Spectrogram Image Features (SIF) features have been proven to be very useful for the area of sound event recognition [3]. The feature is obtained by first converting the spectrograms into grayscale images. These images are mapped into multiple (monochrome) images, which are in their turn divided into smaller blocks that are able to identify the intensity's spatial distribution. The central moments from all those blocks are added up which form the SIF vector.

Stabilized auditory image The Stabilized Auditory Image (SAI) shows the sound signal in two dimensions. However, whereas normally frequency and time are used on the axis, for SAI one axis is the frequency added with a filter bank, and the second is the strobed temporal integration process [70, 71] that is used for the generation of SAI. Paper [11] has shown that SAI can be used for robust sound event classification.

5.5. Raw Waveforms

Whereas usually always features are used in combination with a classifier in order to come to results, lately also satisfactory results have been booked with raw waveforms as input of the classifier [72–75]. The classifier, a neural network, finds its own features which it uses for classification. As this type of classifier is still in its in

[1] `www.speech.cs.cmu.edu/15-492/slides/03_mfcc.pdf`

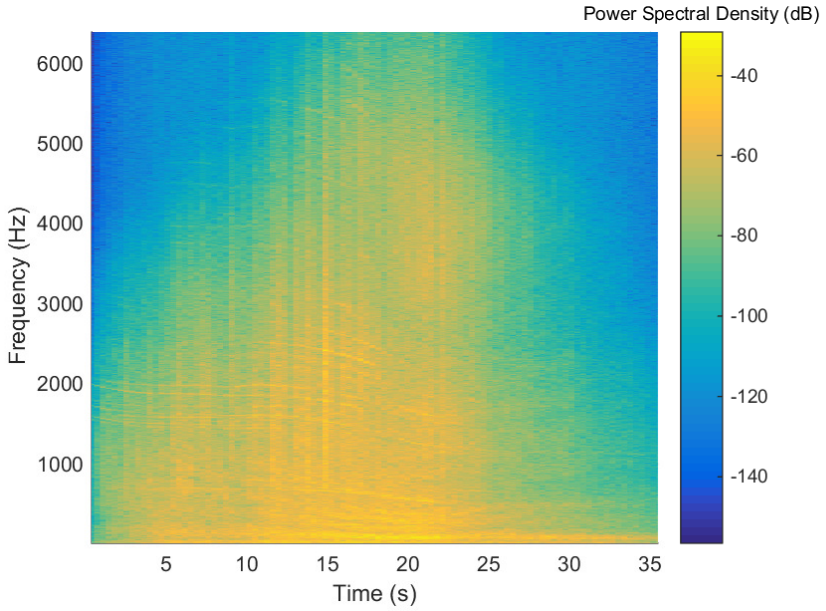


Figure 5.3: An example spectrogram of a fly-over

infancy, the results are not yet such worth mentioning compared to the accuracies that are shown in Table 6.1 (see section 6.5 for more information), and therefore will not be taken in consideration for this research.

5.6. Feature Selection

From the list of features that has been discussed above, only a few will be used in the research for a hear-and-avoid algorithm. One of the most promising features is the SIF. In [3], the noise robustness of the feature is shown in comparison to the MFCCs. [9] shows a slight modification which brings up the accuracy even more. They are described in more detail in section 6.5. MFCCs, which seems to have a worse performance compared to the SIF, will not be discarded. They have been shown to be one of the most useful features for drone detection [76], sound scene recognition [59, 63] and ASR [77].

6

Classification

In the early days of machine hearing, the popular classification techniques were HMMs, GMMs and SVMs. However, since the use of deep neural networks has shown superior classification possibilities, it has been used for sound as well. In order to program any of the classifiers, multiple libraries exists. For machine learning, 'Scikit-learn' [78], 'Keras' [79] and 'Tensorflow' [80] will be used as machine learning libraries and 'Kapre' [81] as audio pre-processing tool. If the detection is successful, the classification will be based on the binary decision: will the neighboring air traffic be colliding with the UAV or not. Of course, a safety factor will be implemented so that near misses are also counted as a collision.

6.1. Gaussian Mixture Models

GMMs are originally used to model data, which is trained in an unsupervised manner. The model comprises of a mixture of Gaussian density components in order to represent the feature space. The audio features usually are a set of vectors (x), which are real-valued. GMMs find the distance between a feature and the feature space ($f(x)$), which is shown in Equation 6.1 [42]. M is the amount of Gaussian mixtures, $P(m)$ is the mixture component's weight and $N(x;\mu_m,\Sigma_m)$ the Gaussian density, which is also expressed in Equation 6.2. In this equation, μ represents the mean and Σ the covariance.

$$f(x) = \sum_{m=1}^{M} P(m)N(x;\mu_m,\Sigma_m) \tag{6.1}$$

$$N(x;\mu,\Sigma) = \frac{1}{(2\pi)^{L/2}|\Sigma|^{\frac{1}{2}}} \exp\left(-\frac{1}{2}(x-\mu)'\Sigma^{-1}(x-\mu)\right) \tag{6.2}$$

The parameters are determined during training, in which the Expectation-Maximization (EM) algorithm is used. This algorithm initially guesses the model parameters, then performs the expectation step which evaluates the current model, which is followed with the maximization step, in which the log-likelihood of the model is maximized. The input for the EM algorithm is a training set of feature vectors.

Even though GMMs are originally unsupervised, which means there are no labels involved, it can be used for multi-class supervised model as well by modeling an individual GMM for each class (y). In that case the goal is to find parameters $\theta = (\theta_y, p_y)$, of which θ_y are the parameters found with the EM method that belong to class y and p_y the probability of class y. When an unlabeled input is given, the label is estimated using Equation 6.3, in which Bayes' rule of Equation 6.4 is applied. The class with the highest probability given by the model is taken as the label for input x.

$$h(x \mid \theta) := \underset{y}{\operatorname{argmax}} f(y \mid x) = \underset{y}{\operatorname{argmax}} f(x \mid y) \times f(y) \tag{6.3}$$

$$f(y \mid x) = \frac{f(x \mid y) \times f(y)}{f(x)} \propto f(x \mid y) \times f(y) \tag{6.4}$$

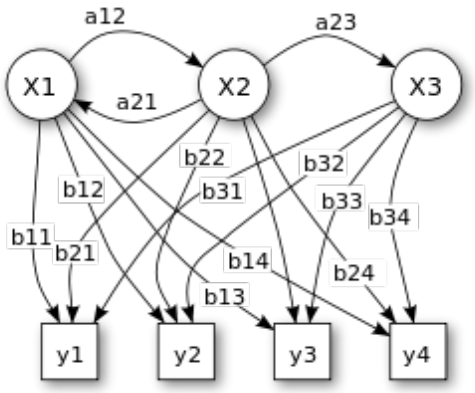


Figure 6.1: Parameters of an HMM [1]

6.2. Hidden Markov Models

HMMs are extending the capabilities of GMMs, as temporal information of the features is included. In an HMM there is a set of hidden states which are interconnected. The GMM is used to find the output probability distribution per state. A set of probabilities is used to guess the transition between the states [82]. The goal of the HMM is to compute the sequence of states that justify the observations that are obtained when a set of features is put in [83]. A visual representation of a HMM is shown in Figure 6.1. In this figure, X represents the states, y the possible observations, and a and b the state transition probability and output probability respectively. The reason that HMMs are popular in audio processing is that, as it includes temporal information, it can model the evolution of time for feature vectors that are frame-based [84].

In total three parameters are estimated with the HMM, which are the initial state distribution $\pi(i) = P(q_1 = i)$, the observation probability distribution $P(x_t \mid q_t)$ and the transition matrix $A(i,j) = P(q_t = j \mid q_{t-1} = i)$. At a time instance (t), the observation is described by (x_t) and the hidden state by $q_t \in 1,....,K$, in which K represents the possible states. The Baum-Welch algorithm is then applied on the training data to find the maximum likelihood, which is an implementation of an EM algorithm [82]. So mathematically, training (with training data X) is performed using Equation 6.5.

$$\theta^{k+1} = \underset{\theta}{\operatorname{argmax}} P(X \mid \theta^k) \tag{6.5}$$

Testing of the HMM is made possible with the so called Viterbi decoding [85]. This process finds the most probable sequence of states that could have created the observed sequence of vectors. This sequence is expressed in Equation 6.6.

$$q_{best} = \underset{q}{\operatorname{argmax}} P(X, q \mid \theta) = \underset{q}{\operatorname{argmax}} P(X \mid q, \theta) \cdot P(q \mid \theta) \tag{6.6}$$

6.3. Support Vector Machines

SVMs are completely different than GMMs and HMMs. The working principle of an SVM is as follows. In a high-dimensional space, a hyperplane is calculated between clusters of points that separate different classes of the classifier [86, 87]. The input to an SVM is a feature vector with dimension L. The feature vector is represented as a point in the L dimensional space. The whole set of vectors (X) is saved as a $T \times L$ matrix, where T is the amount of vectors in the set. Also, each vector has a label that refers to its class. The separating hyperplane is given by Equation 6.7, which does assume that the data is linearly separable. Still, the feature space to which it is applied can be a non-linear transformation of the inputs.

$$\omega \cdot X + b = 0 \tag{6.7}$$

[1] en.wikipedia.org/wiki/Hidden_Markov_model

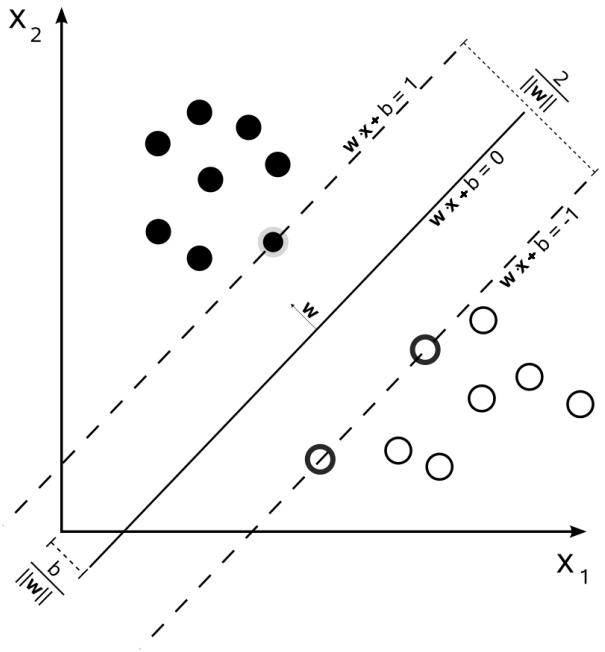


Figure 6.2: Linear separation of two classes using an SVM [2]

ω is the hyperplane's normal line and b is a scalar to determine the perpendicular distance between the origin and the hyperplane, which is given by $\frac{|b|}{\|\omega\|}$. The best values for ω and b are those who maximize the the margin between the two closest data points of each cluster. Mathematically this distance is described $\frac{2}{\|\omega\|}$. So the goal is to find $\min \frac{1}{2}\|\omega\|^2$, while the following constraint also holds: $d_t(y_t \cdot W + b) - 1 \geq 0$. These parameters are also shown in Figure 6.2, in which the black and white dots represent data of two different classes. Even though the previously described formulation is only applicable to two classes, it can be tweaked such that it also works for multi-class, overlapping data, etc [86].

6.4. Artificial Neural Networks

The methods described previously were the most popular ones for machine hearing not so long ago. However, nowadays deep learning using neural networks is getting more and more popular, also in the field of audio processing. There are various types of those networks, among others, the Deep Neural Network (DNN), Convolutional Neural Network (CNN), Recurrent Neural Network (RNN), Deep Belief Network (DBN) and autoencoder.

6.4.1. Deep Neural Network

In deep learning, a network of virtual neurons is created, a so called neural network. An example network is shown in Figure 6.3. The neurons are ordered per layer, at least one input layer, one output layer and one or more hidden layers. All the neurons of one layer are connected with all the neurons of the subsequent layer. A connection between two neurons has a weight (w), which determines how important this connection is for the following neuron. Also, each neuron has a bias which is added/subtracted from the current value of the neuron.

The data is inserted in the input layer, all the values are sent to the hidden layer neurons and are multiplied by the corresponding weight. Per neuron, all the values are added with also the bias, coming to a new value. This value is mapped into a fix-size output using a non-linear function. The three most common functions are expressed in equations 6.8, 6.9 and 6.10.

$$f(z) = \max(z, 0) \tag{6.8}$$

$$f(z) = \tanh z \tag{6.9}$$

[2] en.wikipedia.org/wiki/Support_vector_machine

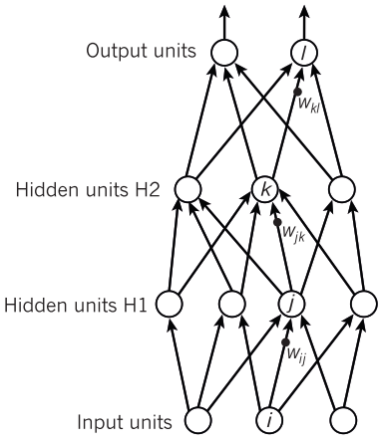


Figure 6.3: Example deep neural network [6]

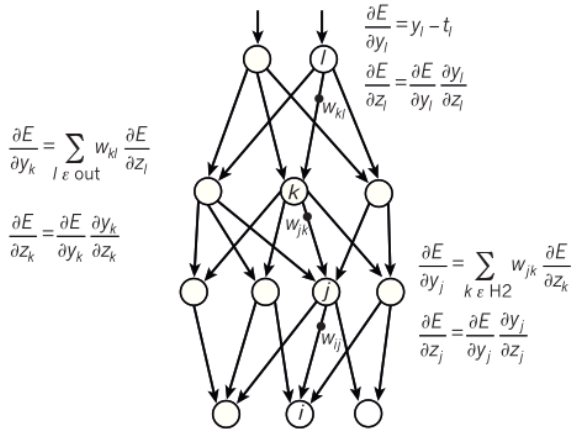


Figure 6.4: Backpropagation for a neural network [6]

$$f(z) = \frac{1}{1+\exp -z} \tag{6.10}$$

The mapped value is in its turn sent out to the next hidden layer, performing exactly the same steps. This continues until the output layer has been reached. The form of the output layer depends on the classification method. If it is a multi-class classification, each neuron in the output layer stands for a specific output, which has a value in the end. If the value is higher than a specific threshold, the computer is convinced that this output is right. When the classification is based on regression, there is usually one output of which the value is a continuous number depending on the input.

However, for the output to be right the weights and biases should be correct, as well as the number of neurons and hidden layers. Therefore, it has to 'learn'. Generally, there are two types of deep learning methods: supervised and unsupervised. Supervised learning means that the data that is put in the test series is labeled, therefore the network knows what the result should be. In unsupervised learning there are no labels, so the network can only cluster outputs. In supervised learning the outputs are known, so an objective function is created which computes the error between the output that was hoped for and the output that was obtained. By then tweaking the weights and biases the objective function is tried to be minimized. Stochastic Gradient Descent (SGD) is a common procedure, which consists of modifying the weights and biases by analyzing multiple small sets training data until a minimum in objective function has been achieved. This minimum does not necessarily have to be the global minimum, but could also be a local minimum. A common thought not so long ago was that the solution would always end up in a poor local minimum. However, in large networks this hardly seems the case. Both empirical as well as theoretical results show that the problem is not a big deal. After training, another set of data is put in the same neural network to check for the systems ability to give sane answers about data it has never seen before during training.

The modification of weights and biases (in for example SGD) is not random. Every time a data set has passed the neural network, backpropagation calculates the gradient of the objective function with respect to either the biases or weights. The name backpropagation comes from the fact that the derivatives are calculated starting by the output, and then working its way back to the input, such as Figure 6.4 shows. E is the cost function in these equations. The gradients then tell how much the weight/bias should increase/decrease. The initialization of the biases and weights is often random. However, there are various methods available for weight initialization for neural networks [88, 89]. A problem with deep neural networks (but also with other forms of neural networks), is the problem of overfitting. Overfitting means that the network has adapted to the training data so well, that its error on training data is very low, but, when new data is presented, the error is very high because the network did not learn to generalize. There are several options to prevent overfitting: the network should not be too complex for the amount of data available, the amount of data should be as high as possible and one can perform dropout in the neural network. When dropout is used in a neural network, every training round neurons in the network are 'dropped out', meaning that they are not forwarding or backwarding information. Which neurons are ignored is determined by randomness for each training round.

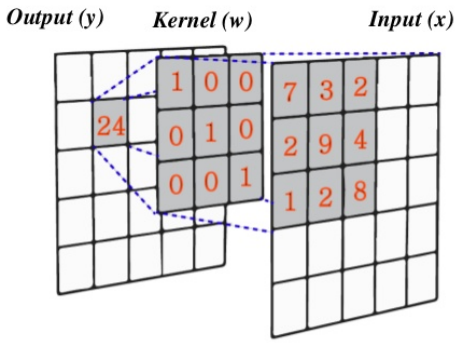


Figure 6.5: Mapping of an input to a convolutional layer [3]

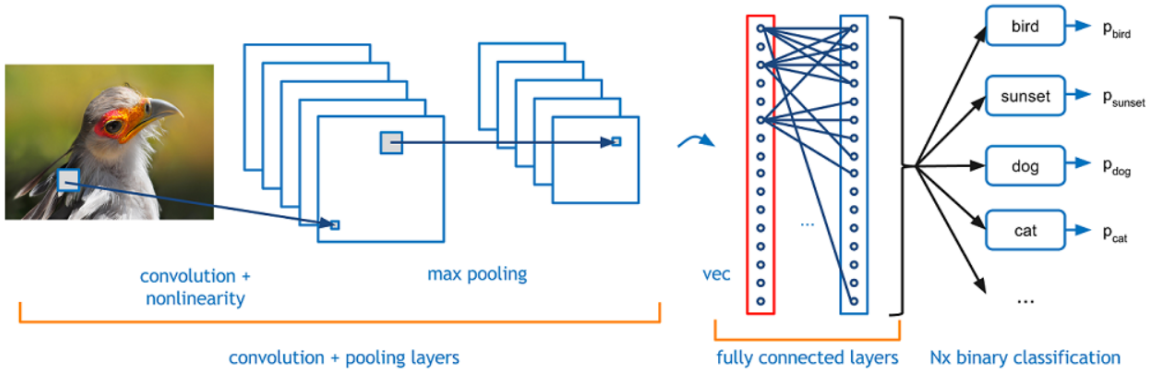


Figure 6.6: CNN architecture [4]

6.4.2. Convolutional Neural Networks

CNN are usually used when the data consists of higher-dimensional arrays, such as a color image. It exists of multiple stages. The convolutional layer and pooling layer comprise the first stages. The convolutional layers are obtained by moving a kernel over image, such as in Figure 6.5. The stepsize of the kernel is called stride. The output of the filter is called a unit, which, when moved over the image, forms a feature map. Per layer, multiple different filters are used. A convolutional layer therefore consist of a set of feature maps. Those layers function to catch associations between features from the previous layer. Like in a DNN, weights are used, which are called filter banks for CNNs. The filter bank of all the units of a feature map is always the same, the filter banks of feature maps of the same layer are different. There are two reasons for this type of architecture. First of all the motifs in an image are not bound to a location in the image, it can appear everywhere on the picture. Secondly, often data points which are spatially grouped are correlated, which actually form the motifs.

The pooling layers combine equivalent features into one. They reduce the dimension and make sure that distortions or small shifts do not affect the network. After multiple convolutional and pooling layers fully-connected layers are added which take care of the classification ability of the network. Also for CNNs a non-linear function is used to map the weighted sum of all units to a value fixed-size output. An overview of a CNN is shown in Figure 6.6 .

Even though convolutional neural networks are originally created for data with multi-dimensional arrays, it also works in one dimension, such as sound. An example of raw waveforms in combination with a CNN can be found in [72, 73]. Another option would be to use an image of the audio, such as the spectrogram.

6.4.3. Recurrent Neural Networks

An RNN is meant for data that is put in sequentially. Think of, for example, speech. It often exists of Long Short-Term Memory (LSTM) units that represents a state vector, which is updated every time a new element

[3] www.slideshare.net/SeongwonHwang/convolutional-neural-network-cnn-presentation-from-theory-to-code-in-theano
[4] adeshpande3.github.io/A-Beginner%27s-Guide-To-Understanding-Convolutional-Neural-Networks

of the sequence is put in. The state vector therefore does not only contain knowledge about the current situation, but also about the history. Its mathematical expression is shown in Equation 6.11 and 6.12.

$$\mathbf{h}_t = F(\mathbf{W}_{xh}\mathbf{x}_t + \mathbf{W}_{hh}h_{t-1} + \mathbf{b}_h) \tag{6.11}$$

$$\mathbf{y}_t = G(\mathbf{W}_{hy}\mathbf{h}_t + \mathbf{b}_y) \tag{6.12}$$

In these equations, the sequence of hidden activations $\mathbf{h}_t$ per timestep t is obtained by using an input vector $\mathbf{x}_t$, a weight $\mathbf{W}$ between each layer and a bias $\mathbf{b}$. This is used to find the output state vector $\mathbf{y}_t$. F and G represent the respective activation function. A visual representation of a RNN is given in Figure 6.7.

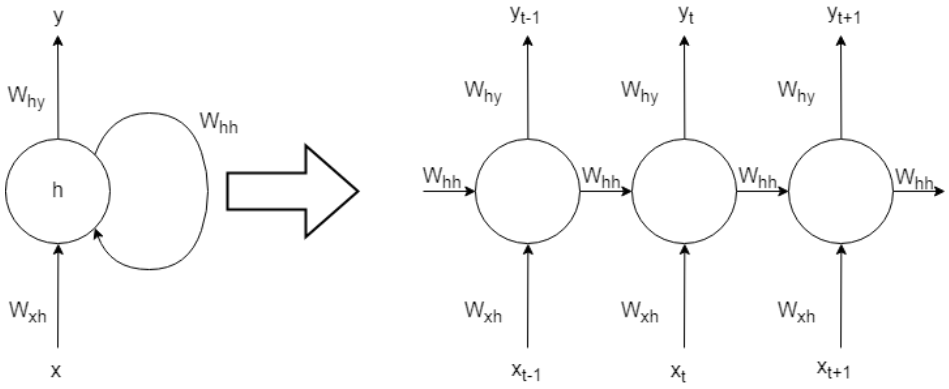


Figure 6.7: An RNN network, folded (left) and unfolded (right)

d

The big advantage of RNN is that it is a dynamic system. It used to have a huge disadvantage too. When an RNN is trained using backpropagation the gradients either vanish or explode after a lot of time steps when using backpropagation [90, 91]. This problem, however, is solved with the use of LSTMs.

6.4.4. Convolutional Recurrent Neural Networks

The CNN and RNN have also been combined for sound event detection [16], ASR [74, 92, 93] or music identification [94]. The idea of combining the two comes from the fact that CNNs miss out on longer temporal context information and RNNs are not able to obtain the invariance in the frequency domain [16]. Combining the two therefore would solve each others disadvantage. This classifier is named Convolutional Recurrent Neural Network (CRNN). The input for a CRNN is a time-frequency representation of the sound. The convolutional layers then function as the feature extractor and the recurrent layer provide the context information by integrating the features over time.

6.5. Classifier Selection

The previously described classification methods have all been applied for audio processing purposes. However, most of the existing works use clean sounds, which means that there is hardly any noise present. Using noise-included sounds makes classification more difficult. In Table 6.1 the classification accuracy for different state-of-the-art methods is presented. The results are coming from papers that all have been using the exact same approach. Namely, the sounds are obtained from the Real World Computing Partnership Sound Scene Database in Real Acoustic Environments [95], which is a database including 50 sound classes of different environmental sounds. These recordings are mixed with the NOISEX-92 database, which is a database with noise recordings. The Signal-to-Noise Ratio (SNR) is changed four times, from a clean recording up to a SNR of 0 dB. The table shows results from three different researches. First of all [3] proves the robustness of the SIF feature with an SVM (SIF-SVM) classifier compared with the old-fashioned feature and classifiers MFCC-HMM and MFCC-SVM. This is followed up by [10, 11], showing the improvement that the DNN makes when using the SIF feature (SIF-DNN). The same research group also discovers that CNN are even better [9], which is outperformed by a SIF-CNN structure in [12], called the 1MaxCNN.

Table 6.1: Comparison of state-of-the-art classifier systems for different noise levels for sound event recognition [3, 9–12]

System	Clean	20 dB	10 dB	0 dB	Mean
MFCC-HMM	**99.4%**	71.9%	42.3%	15.7%	57.4%
MFCC-SVM	98.5%	28.1%	7.0%	2.7%	34.1%
SIF-SVM [3]	91.1%	91.1%	90.7%	80.0%	88.5%
SIF-DNN [11]	96.0%	94.4%	93.5%	85.1%	92.3%
SIF-CNN [9]	97.3%	97.4%	95.7%	83.1%	93.4%
1MaxCNN [12]	99.1%	**99.0%**	**98.9%**	**97.5%**	**98.6%**

Based on the results of Table 6.1, the SIF feature in combination with CNN (e.g. the SIF-CNN and the 1MaxCNN) exceeds in accuracy, which is the reason that this combination will also be applied to the hear-and-avoid algorithm.

The previous systems have been applied to sound event recognition. Looking in the noise-robust classifiers for ASR, often RNNs are used [96–98]. Also in [13], which is unfortunately not under noisy conditions, an LSTM and CNN-LSTM are compared with DNN and CNN based on the classification performance, of which the result is presented in Table 6.2. The paper performs the Detection and Classification of Acoustic Scenes and Events (DCASE) challenge [99], where varous scenes (library, park, train, etc.) are classified. Both LSTM and CNN-LSTM are comparable to the performance of the CNN and therefore also a good candidate to use for classification purposes.

Table 6.2: Comparison of the accuracy of different artificial neural networks for the DCASE challange [13]

	DNN	CNN	LSTM	CNN-LSTM
Accuracy	75.3%	78.0%	77.3%	79.2%

The classifiers will be built such that it can recognize three important characteristics in the audio signal: the loudness of the aircraft sound, the frequency content (envelope) in the signal and the (absence) of the Doppler shift. The reasoning behind this is that the audio signal of an approaching aircraft gets louder and gets more high frequency power. The Doppler shift in the signal proves whether the aircraft and UAV are on a collision course or not.

7

Sound Localization

The localization of sound is of great help for the hear-and-avoid algorithm, as it can be used for determination of the escape path in case of a possible collision. There are multiple algorithms that are used for localization purposes, as well as the hardware needed for those algorithms.

7.1. Multiple Signal Classification

One of the most used methods of sound localization in robotics is Multiple Signal Classification (MUSIC) [100]. MUSIC is, among others, a method that estimates the Direction of Arrival (DOA), the number of signals present, the strength of noise and the strength of the waveforms. Starting from the audio signal, the form in which the (complex) output ($\mathbf{x}(t)$)is obtained from the array of microphones is shown in Equation 7.1 [101]. $\mathbf{s}_i(t)$ and $\mathbf{n}(t)$ are the signal complex envelop representation of the i^{th} source and the complex noise respectively.

$$\mathbf{x}(t) = \sum_{i=1}^{d} \mathbf{a}(\theta_i) s_i(t) + \mathbf{n}(t) \tag{7.1}$$

The vector $\mathbf{a}(\theta_i)$ is the so called array steering vector for direction θ_i, which holds the phase delays of the travelling wave between each element in the array. Furthermore, d is the amount of sources. Another way of writing Equation 7.1 is presented in Equation 7.2, where $A = [\mathbf{a}(\theta_1), ..., \mathbf{a}(\theta_d)]$ and $\mathbf{s}(t) = [s_1(t), ..., s_d(t)]^T$.

$$\mathbf{x}(t) = A\mathbf{s}(t) + \mathbf{n}(t) \tag{7.2}$$

The goal of MUSIC for the DOA problem is to estimate all θ_i for $i = 1, ..., d$. In order to solve this, a few assumptions are made. First of all the amount of sources is less than the amount of sensors. Secondly, the sources are uncorrelated, stationary and zero mean. The covariance matrix therefore looks like Equation 7.3, with σ_i^2 the power of the source per source (i) and H the Hermitian transpose. Thirdly, also the (white) noise is zero mean, stationary and uncorrelated, with the covariance matrix as in Equation 7.4, where M is the amount of sensors. Furthermore, the noise and signal are uncorrelated and lastly the vector $\mathbf{a}(\theta)$ is known for all θ.

Due to the assumptions made before, the covariance matrix of the output data can be written as in Equation 7.5. Because there are only a N number of data samples, R_x is usually estimated following Equation 7.6.

$$R_s = E[\mathbf{s}\mathbf{s}^H] = diag(\sigma_1^2, ..., \sigma_d^2) \tag{7.3}$$

$$R_n = E[\mathbf{n}\mathbf{n}^H] = \sigma_n^2 I_{M \times M} \tag{7.4}$$

$$R_x = E[\mathbf{x}\mathbf{x}^H] = AR_sA^H + \sigma_n^2 I \tag{7.5}$$

$$R_x = \frac{1}{N} \sum_{j=1}^{N} \mathbf{x}(t_j)\mathbf{x}(t_j)^H \tag{7.6}$$

From the fact that A is full column rank and R_s is non-singular, AR_SA of Equation 7.5 must be an $M \times M$ matrix with rank d [102]. This means that there are $M-d$ eigenvectors q_m that have zero as an eigenvalue. Q_n is the $M \times (M-d)$ matrix that holds these eigenvectors. One property of each of the eigenvectors in Q_n is that it is orthogonal to the signal steering vectors. Looking at Equation 7.7, which describes the pseudo-spectrum, P_{MUSIC} will go to infinity as the denominator gets to zero. The signal direction ϕ is therefore estimated as the M biggest peaks in the spectrum.

$$P_{MUSIC}(\phi) = \frac{1}{a^H(\phi) Q_n Q_n^H a(\phi)} \tag{7.7}$$

In a real situation, R_s is not available, only R_x can be obtained. However, when combining Equation 7.5 with $R_s q_m = \lambda q_m$, Equation 7.8 is obtained. It shows that R_s and the estimate of the signal covariance matrix R have the same eigenvectors that have eigenvalues $\lambda + \sigma^2$. Let $R_s = Q \Lambda Q^H$, then R is found by Equation 7.9. This equation shows that the matrix Q can be divided in two parts, namely the signal eigenvector matrix Q_s that defines the signal subspace and the noise eigenvector matrix Q_n that defines the noise subspace. The noise eigenvector matrix Q_n is identical to the eigenvector matrix that belongs to the zero-eigenvalues of R_s, so still it is orthogonal with respect to the signal steering vector. Therefore Equation 7.7 can be used for DOA by finding ϕ for all the peaks in the pseudo-spectrum.

$$Rq_m = R_s q_m + \sigma^2 I q_m = (\lambda_m + \sigma^2) q_m \tag{7.8}$$

$$R = Q[\Lambda + \sigma^2 I] Q^H = Q \begin{bmatrix} \lambda_1 + \sigma^2 & 0 & 0 & 0 & \cdots & 0 \\ 0 & \ddots & 0 & \vdots & \vdots & \vdots \\ 0 & 0 & \lambda_M + \sigma^2 & 0 & \cdots & 0 \\ 0 & \cdots & 0 & \sigma^2 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & \cdots & 0 & 0 & \cdots & \sigma^2 \end{bmatrix} Q^H \tag{7.9}$$

7.2. Time Difference of Arrival

Another popular method for sound source localization is the Time Difference of Arrival (TDOA). For this method, an array of microphones is used of which the cross correlation function between all the microphones is obtained. The largest peak in the cross correlation function of two time delayed signals is assumed to be the time difference between the incoming signals. This is shown in Figure 7.1, where time difference τ is equal to five. It is based on the human interaural localization approach, which also correlates sound between the two ears using so called coincidence detectors [103]. For two different microphones (i and j) of the array, the link of the two systems is shown in Equation 7.10 and Equation 7.11 [104]. $m_i(t)$ stands for the measured signal at microphone i, $s(t)$ is the received sound signal of the i^{th} microphone coming from the sound source, $h_r(t)$ represents the deterministic impulse response between the signals and $n(t)$ the noise in the signal.

$$m_i(t) = s(t) + n_i(t) \tag{7.10}$$

$$m_j(t) = (s * h_r)(t) + n_j(t) \tag{7.11}$$

Assuming that there is no scattering taking place when the signal is propagating from its source to the receiver and all the signals and noise are stationary zero mean, $h_r = \delta(t + \Delta T_{ij}) = \delta_{-\Delta T_{ij}}$, explaining that it only catches the TDOA ΔT_{ij}. The azimuth angle θ is related to ΔT_{ij} by Equation 7.12, where l_{ij} is the distance between microphone i and j. ΔT_{ij} is obtained from the audio signals m_i and m_j using the cross-correlation function $R_{m_i m_j}$, which is shown in Equation 7.13.

$$\Delta T_{ij} = \frac{l_{ij}}{c} \cos\theta \tag{7.12}$$

$$R_{m_i m_j}(\tau) = E[m_i(t) m_j(t-\tau)] = (R_{ss} * h_r)(-\tau) + R_{n_i n_j}(\tau) \tag{7.13}$$

[1] en.wikipedia.org/wiki/Multilateration

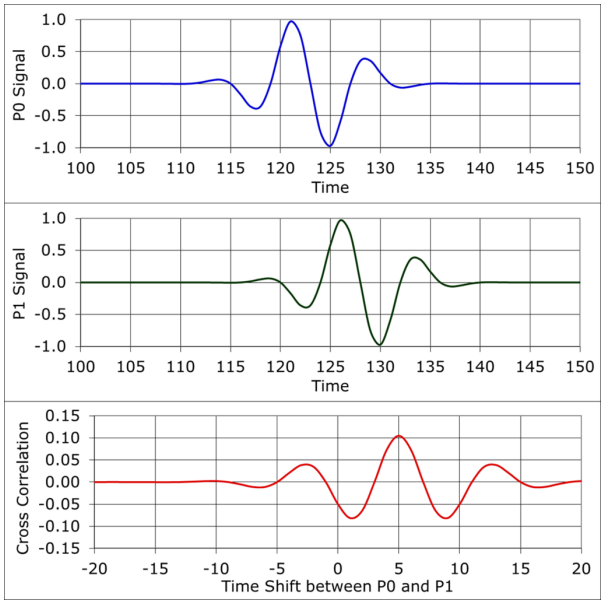


Figure 7.1: Cross correlation function between two signals[1]

R_{ss} and R_{nn} are the source and noise autocorrelation function respectively. Due to the assumption that the noise signals are independent, the $R_{n_i n_j}(\tau)$ part drops out. This leads to Equation 7.14.

$$R_{m_i m_j}(\tau) = (R_{ss} * \delta_{-\Delta T_{ij}})(-\tau) = R_{ss}(\tau - \Delta T_{ij}) \tag{7.14}$$

What this equation shows is that $R_{m_i m_j}$ is nothing else than a shifted version of R_{ss}. As R_{ss} has the property $R_{ss}(\tau) \leq R_{ss}(0)$, $R_{m_i m_j}$ has a maximum for $\tau = \Delta T_{ij}$. Unfortunately, $R_{m_i m_j}$ is not known in real practise, as there is just one realization of m_i and m_j present. Therefore, $R_{m_i m_j}$ is estimated and the TDOA ΔT_{ij} is found by Equation 7.15.

$$\Delta \hat{T}_{ij} = \underset{\tau}{\operatorname{argmax}}(\hat{R}_{m_i m_j}(\tau)) \tag{7.15}$$

The estimation of $R_{m_i m_j}$ is mostly done by means of the Generalized Cross Correlation (GCC), which is expressed in Equation 7.16. It is the inverse Fourier transform of a weight function $\Psi(f)$ and cross-power spectral density $\hat{S}_{m_i m_j}(f)$. The weight function can be computed in many different ways, of which most are present in [105]. The cross-power spectral density is found using [106]. Both are not further elaborated on as it is out of the scope of this research. Putting the estimation of the time difference $\Delta \hat{T}_{ij}$ in Equation 7.12, the azimuth angle θ can be obtained, which is equal to the direction of the sound source. When more than two microphones are used, the θ of different microphone pairs do will not be exactly equal (due to noise). An estimator, such as least squares estimator [107] or coherence pruning [22], can be used to find the most probable direction.

$$\hat{R}_{m_i m_j}(\tau) = \int_{-\infty}^{\infty} \Psi(f) \hat{S}_{m_i m_j}(f) e^{j2\pi f t} df \tag{7.16}$$

7.3. Beamforming

Beamforming is a method that uses also an array of microphones for simple, computationally cheap localization. However, the performance of the localization is very dependent on certain array characteristics, of which the number of microphones is the main factor. In beamforming, all separate microphones in the array obtain discrete time audio signals that are used to focalize it to the specific direction $\mathbf{r}_0$. Having $m_n(t)$ as the sound signal for microphone n and $w_n(\mathbf{r}_0, t)$ as a linear filter of impulse responses, the sum $y_{\mathbf{r}_0}(t)$ of the outputs is calculated by Equation 7.17. This equation can be transformed to the frequency domain, which result in Equation 7.18, where $S_s^0(k)$ represents the frequency component at a specified point (0) for source s, $W_n(\mathbf{r}_0, k)$ is the frequency response of the filter, B_n the frequency component of the noise and $D_{\mathbf{r}_0}(\mathbf{r}, k)$ is shown in Equation 7.19, where $V_n(r, k)$ is the steering vector. These equations hold for both farfield and

nearfield, and it is assumed that the wavefronts are planar. The information obtained is used to create an energy map of the surroundings $E(\mathbf{r}, t)$ following Equation 7.20. T is the length of the time window. The position of the sound sources is estimated as finding the maximum value for E(**r**,t). Usually, a set of possible sound source directions is used for evaluation of Equation 7.20

$$y_{\mathbf{r}_0}(t) = \sum_{n=1}^{N} w_n(\mathbf{r}_0, t) * m_n(t) \tag{7.17}$$

$$Y_{\mathbf{r}_0}(k) = \sum_{s=1}^{S} D_{\mathbf{r}_0}(\mathbf{r}_s^s, k) S_s^0(k) + \sum_{n=1}^{N} W_n(\mathbf{r}_0, k) B_n(k) \tag{7.18}$$

$$D_{\mathbf{r}_0}(\mathbf{r}, k) = \sum_{n=1}^{N} W_n(\mathbf{r}_0, k) V_n(\mathbf{r}, k) \tag{7.19}$$

$$E(\mathbf{r}, t) = \int_{t-T}^{t} \mid y_{\mathbf{r}}(\tau) \mid^2 d\tau \tag{7.20}$$

7.4. Binaural Localization

Humans and animals are capable of localizing a sound pretty accurate in 3D with only two ears. Also in robotics, this binaural approach has been tried [104]. In these approaches, two acoustical sensor are usually positioned in an artificial pinna. The human localization abilities were already explained in the 80s [108], which forms the basis of the binaural localization process. First of all, sound is modified by the body, head and pinna before it enters the ear. The same is done in robotics, by a relation that is called the Head Related Transfer Function (HRTF). Secondly, features are extracted from the sound, send to the brain and integrated there. For horizontal localization, humans make use of interaural cues [109] and for vertical localization spectral notches are used [110]. So combining the horizontal and vertical positions, the 3D localization is performed.

In the field of robotics, the binaural approach have not been implemented satisfactory enough yet. This is first of all because mimicking the human ear is quite difficult. The use of the interaural cues demands a high precision modeling of the disturbances caused by the head. Especially on UAVs, where there is not a body present that could modify the sound as the head does, it is hard to set up a HRTF. Another problem is that for existing methods, the used techniques are sensitive to changes in the environment. Whereas the binaural localization might work in an anechoic room, in a real-life environment the accuracy would be far from off if the exact description of the environment was not given. Consequently, researches looked for approaches with more than two microphones, which, as section 7.6 shows, does also not work for this research.

7.5. Single Microphone Solutions

The direction of a sound has also been found using only one microphone. A first example is [111, 112], which uses the Doppler effect to estimate some flight parameters, such as the speed and the altitude of a low-flying aircraft. Although this method seems to work satisfactory, it has some limits. First of all the frequency of the aircraft should be narrowband and constant, which means that it would only work on propeller aircraft. Another method has been established as well that works for jet sounds [113], but in order to work the microphone should be placed next to the ground as it takes the multipath interference of the sound into account [114]. Another problem with these methods is that the whole Doppler shift is needed for determining the flight parameters. This would be in case for this research too late, as the decision has to be made before the closest point of approach.

However, other research such as [115] proves that also with a single microphone range can be estimated using the relative magnitudes of two different frequencies. In order to be able to find the range, two frequencies from the source should be known as well as their relative magnitudes. Equation 7.21 shows how to calculate range d, where y_0 and y are the relative magnitudes of the frequencies at the source and the measured point respectively, the dynamic viscosity coefficient of air is described by η, ρ is the density, V the speed of sound and f_h and f_l the respective high and low frequencies. In order for this to work the frequencies must thus be known beforehand. Also, this has not been tested in a noisy environment, so it might be difficult to have a proper estimation for the hear-and-avoid algorithm using this method.

$$d = \frac{\ln(y_0/y)}{\frac{8\eta\pi^2}{3\rho V^3}(f_h^2 - f_l^2)} \tag{7.21}$$

Another method is to use special microphones, such as in [24, 25]. Using an Acoustic Vector Sensor (AVS), which consists of only one microphone in combination with three orthogonal particle velocity sensors, the sound of a horn is captured while moving the drone. A second test involves the presence of a secondary drone, which accelerates when the recording drone is passing. The disadvantage of this particular microphone was the low signal to noise ratio in the low frequencies and the disability to deal with high wind speeds. Lastly, in [116] a single-point stereo microphone is used. It takes noise of the robot into account, however, this is not comparable with the ego-sounds produced by the drone. Also, the size of the microphone is a limiting factor for its applicability on drones.

7.6. Difficulties in Localization

Even though the previously described methods are popular in localization for robotics, there are a few constraints that prevents this research from including localization of the aircraft sound. The most important constraint is the time constraint. The master takes in total nine months of full time work, which is expected to be not enough to make a detection module, feature extraction module, classification module and a localization module.

There is also the frequency constraint, which deals with the bandwidth of the signal. The sound that the UAV will experience will be broadband, meaning that there is a wide range in bandwidth with respect to the main frequency. However, for the MUSIC method this is a problem as it is based on narrowband sounds [104], which is thus not applicable in this area.

Furthermore, one should consider the environmental constraint. The environment of UAVs are dynamic, unpredictable and contaminated with noise. As many researches perform the experiments in acoustical controlled areas, it does not take noise into account, which deteriorates the performance of the aircraft localization.

For those reasons it is decided not to include localization in this research. However, as there are some promising researches performed on air traffic localization on drones, this could be a follow up topic/part of the hear-and-avoid algorithm. For now, as the localization is not performed, and thus the direction of escape cannot be established, it is assumed that landing is the safest option in case of a possible collision situation.

8

Project Plan

8.1. Research Question, Aims and Objectives

The main objective of this research is to decrease the chance of a possible collision in the air between air traffic and drones by creating a 'hear-and-avoid' algorithm for the drone. The hear-and-avoid algorithm is an audio classification mechanism that will be able to first of all recognize sounds in its environment over its ego-sound, classify this sound, determine whether it forms a danger and base actions upon that. This is not an easy task, because as the previous chapters show, until now there is no research done on an equal project and most of the detection and localization algorithms are performed under 'perfect' circumstances. The fact that no one has done it yet makes this research unique and creates a lot of possibilities for future research topics. The feasibility of the project is supported by the few researches that show promising results for robust (UAV) sound classification [9, 22, 23].

The steps to come to the hear-and-avoid algorithm is set out in the form of the research questions, which are shown in Table 8.1. The order of the questions is also the order in which they will be solved, as they follow up on each other. Also, each question has its sub questions that together help to answer the main questions.

Research question 1 and 2 are dealing with the detection and classification of the possible air traffic sounds. Question 1 deals with detection: it will use sounds from a recording that includes a lot of noise from wind, microphone inaccuracies and of course the UAV's ego-sound. The latter can possibly be captured or estimated and used to filter itself out, making it already easier to detect other objects from the same recording. Furthermore, also other forms of noise (wind/microphone noise) should be filtered out as much as possible to make detection even easier and to make classification more accurate.

Question 2 is about creating the classifier. As Figure 3.1 shows, a SER module consists of detector, audio features and a classifier is needed. The detector will be obtained by answering research question one. The features and classifiers will be obtained when question 2 will be answered, so when multiple audio features in combination with a classifier have been trained and tested. The preceding chapters describe that many classifiers and features exists, so combining them gives even more possibilities. However, the features and classifiers do not always work as good for one application as for the other, for example, there is a huge difference in building a classifier for music classification and speech recognition. It is therefore important that good research is performed on the feature and classification methods on the suggested features and classifiers of chapter 5 and 6. Furthermore it is important to determine on which basis the classifier will be built. For instance, is it sufficient to distinguish air traffic from normal traffic, or should also air traffic be distinguished in helicopters and airplanes, etc.

Questions 3 and 4 examine what should be done with the classified sound. First of all, when it should react to the sound, so questioning when something is a possible hazard or not, and how to find those details in the audio. When a sound is labeled as dangerous, the next thing to do is perform action. Localizing the source of the sound would help determine what way out is the most optimal for the drone to go to. However, the localization of the sounds on a drone with a single microphone will have very poor accuracy. Subsequently, when performing a manoeuvre to avoid the danger, it should do as quickly as possible with minimum chance of staying in the dangerous situation or creating a new dangerous situation. This also means it is bound to time. Air traffic is always fast moving, so it only needs little time to cover large distances. Therefore a drone has very limited time to react on that.

Table 8.1: Research questions

	Research Questions
1	In which way(s) can the presence of a non-drone sound be detected?
a	How can the ego-sound of the drone be filtered out?
b	How can the noise of the wind be filtered out?
c	How can the noise of the microphones be filtered out?
2	What procedure works best in order to classify the non-drone sound?
a	What are the classes for which the classification is going to be performed?
b	What classification algorithm performs best in terms of precision and recall?
c	Which audio features performs best as input for the classifier?
3	When should the drone react to the sound?
a	When is the sound a possible hazard for the drone?
b	How is possible hazard information extracted from the sound?
c	How much time is required to react to possible danger?
d	How much time is available to react to possible danger?
4	How is the algorithm able to perform real time while flying the drone?
a	How much computation space does the drone have for this application?
b	How much computation space is required for the algorithm?
c	How could real time drone state information help to improve the algorithm?

Table 8.2: Research sub goals

	Sub goals
1	Create feature extractors
2	Create classifiers
3	Train classifiers
4	Create hazard recognition
5	Combine 1 till 4 into one module
6	Implement algorithm in UAV
7	Perform the experiment
8	Verify & Validate

The last research questions considers the implementation of the algorithm on the drone. As can be concluded from the previous questions, the algorithm should process, classify and react on a sound as soon as possible. All this requires computation power, which is quite limited on a drone. Therefore the algorithm should be optimized as much as possible. When the algorithm is working on the drone, it can also use the drone's states, such as position, velocity, rpm, etc. for prediction purposes. For example, knowing the rpm the sound created by the rotors can be estimated and filtered out from the audio recording. The knowledge of position and velocity can help determining the optimal escape path.

In order to help answering the research questions and obtaining the research objective, sub goals have been created. These sub goals are presented in Table 8.2 and their planning in Appendix A. More details on the performance of those sub tasks is explained in section 8.2.

8.2. Methodology

The research questions of Table 8.1 will be answered in the following manner. For question 1 models will be created that approximate the UAV's ego-sound, the wind and the microphone noise. The assumption is made that only one aircraft or rotorcraft will be present at the same time, as the chance of low flying air traffic in UAV allowed areas is already small due to the restrictions and laws (see chapter 1). Also it is assumed that the ground traffic or other forms of noise is not present (yet). Therefore, the left over sound must be the object of interest.

The classification tasks will be based on certain features. The selected features are discussed in section 5.6,

which are the SIF features, MFCC, and due to the promising first results, also raw waveforms will be taken as input.

During the research, three of the classification techniques based on supervised learning that are described in section 6.5 will be tested: CNNs, LSTMs, and CNN-LSTMs, which all use the previously described features as an input. Two forms of classification will be performed that will answer research questions 2 and 3. One that determines the type of object (is it a air vehicle or a ground vehicle) and one that just determines whether the recording sounds like a colliding vehicle or not.

When all the classifiers have been built, the training can begin. Two types of data will be used: obtained audio fragments and augmentations of these fragments. The audio that is obtained comes from real recordings that are taken on site (on an airport) and that are available online. The recordings are performed with only a single microphone in order to keep the algorithm as simple and computationally inexpensive as possible. The data augmentation exist of shifting in pitch, adding noise, changing the loudness ratio drone/aircraft or stretching the sound. The audio consist not only of drone sounds and airplane sounds for both jet and propeller aircraft, but also sounds that are similar to the airplane sounds but are not the same. Examples are cars, trucks, tractors, etc. The reason to include them as well is that they could confuse the UAV that it hears an airplane and therefore may move, while ground vehicles would never be a hazard for flying drones (assuming that the drones do not intend to fly at very low altitudes). From all the audio data, various combinations will be created, namely batches of only UAV sound or batches that include UAV sound and one or more non-UAV sound(s). Part of these batches, approximately 70%, will be used for training. The other 30% is used for testing purposes.

As explained before, the localization of the source of the sound with a single microphone will not be performed in this research. A range estimation can be made using Equation 7.21, which could help in determining whether a sound comes from an aircraft that is in a hazardous position for the drone. It also helps answering research question 3 and 4. The frequencies used and their relative magnitude can be obtained from the Aircraft Noise and Performance (ANP) Database[1]. Research question 5 is investigated in the next section, section 8.3, where the best performing SER system will be implemented in the UAV.

8.3. Experimental Set-up

Due to the restrictions of UAVs around airports/aircraft, the most logical option of testing the algorithm is not possible. That would be, the drone is flown and aircraft would manoeuvre around it. If it is working correctly, the drone would avoid the aircraft successfully. Nevertheless, the sound of an aircraft can be generated accurately without the aircraft being physically present by means of simulations. Using [27], the sound propagation, attenuation and Doppler effect are simulated. The only factor that will be hard to replicate is the loudness of the signal. In the Cyberzoo at the TU Delft the UAV can be flown safely. Using an external speaker to generate aircraft noises can be a way to expose the UAV to air traffic noise. An important factor to keep in mind when performing tests in this way is that as much noise from other machines in the Cyberzoo as possible is diminished. Due to the same nature of aircraft and machine sounds the machine sounds could affect the tests.

There are two ways in which the test can be held, each with their own advantages and disadvantages. The first option is to have a stationary external speaker that simulates the sound of a moving aircraft. The second option comprises of a moving speaker that plays stationary air traffic sound. Option one has the benefit that only little space is needed to perform the tests, but has as downside that accurate simulations have to be made that perfectly represent dangerous and non-dangerous situations. For option two the sound is not much of a problem, but the movement is. First of all because it requires a lot of space, and secondly, in order to achieve the correct Doppler shifts, high velocities should be used while moving the box. However, it is for this option very easy to created dangerous situations by flying straight to the drone and safe situations by flying on a non collision course.

Weighing the two options up against each other, it seems more achievable to go for the first option. Sound propagation is an area in which already a lot of research has been done [117], so simulating various possible collision situations is possible. The tests consist of different test cases. First of all, multiple audio files are used for the test series, which are, just like the training data, all kinds of traffic sounds. The algorithm should first of all include as few false negatives as possible, but also as few false positives, in order to avoid constant avoidance maneuvers. Therefore not only aircraft and rotorcraft sounds are used, also other engine sounds, such as cars, lawn mowers, tractors, etc. should be used for testing. Another requirement is that the sound

[1] www.aircraftnoisemodel.org/data/aircraft

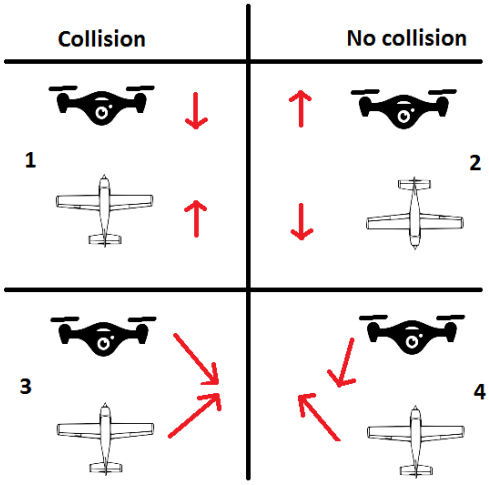


Figure 8.1: Collision test cases

audio files have never been used for training of the algorithm, as it then is already able to 'recognize' it. During the tests, the UAV is stationary (but in the air) while the sound is initiated at different distances from different positions. It is important to realize the path of the simulated sound in order to determine where to initiate the sound, as it determines whether it is on a collision course or not. For the tests the relative movement in various directions are measured, which are visualized in Figure 8.1: the simulated sound and the UAV moving straight to each other (1), the simulated sound and the UAV moving straight from each other (2), the simulated sound source moving in a direction that involves also a perpendicular vector with respect to the UAV that will result in a collision (3) and lastly one that does have the perpendicular vector but which is not a collision course (4). These different approaches are taken due the Doppler shift. When two objects move straight to each other, a Doppler shift will look like a step function, whereas the Doppler shift of two objects that are not moving exactly straight to each other looks like a sigmoid function, so there is a transition in frequency before the closest point of approach occurs. This is is visualized in figure Figure 2.2, where the blue line represents the sound object going through the receiving object and the orange line represents the sound object passing the receiving object. If the transition does not take place, it means either there is a collision course, the objects are moving away from each other or the distance of the objects is to far away. Testing the first two approaches prevents the algorithm from thinking that no transition always means collision. The latter two approaches cover all the other possible relative movements, which learns that when there is a Doppler shift present the objects will never collide (if they keep following the same path).

8.4. Results, Outcome and Relevance

From the test, various data will be obtained. First of all the position of the UAV should be known, which is synchronized with the position of the sound source and the sound information. This synchronization makes sure that the relative positions of the UAV with respect the sound source are always known. Furthermore the sound information is required in the synchronization so that validation can be done whether the UAV only moves for air traffic sounds. From the positions of the sound source the direction of the sound source is established to determine whether its moving to, from or perpendicular to the UAV. Having this information, as well as the position information of the UAV and the label of the sound source, validation can be performed.

The desirable outcome shows that the UAV does not change its speed, heading and altitude when there is no danger, so when there is no air traffic or when the air traffic is not on a collision course with the drone. Also it should move to safe location, which would be initially on the ground, if there is a the possibility of a collision. The choices that the UAV makes will be expressed in precision and recall. Recall is the most important of the two as every collision should be prevented at all times, even if that means some extra false positives. In a situation of collision it is better to be rather safe then sorry as the outcome of a collision could be lethal. However, precision is still important as otherwise the UAV would stop performing its task every time it hears something.

The conclusion to be drawn from the outcome is to show how well the audio classification algorithm

works and how applicable it is in the use of the UAV. The main deciding factor is the precision and recall. The relevance of this outcome is to show that this research will have a positive impact on the safety of air traffic. Furthermore it will open various new research possibilities, as a research like this has not been performed yet on UAVs.

8.5. Project Planning

The thesis has been divided in four phases, which starts with the literature study. This report has been created during this phase and includes the outcome of the literature study. Many papers about audio features and classification are analyzed during this stage in order to obtain a clear overview of which knowledge and methods have already been obtained so far. All this knowledge will be written out in the literature report and will be presented on the last day of the phase during the literature presentation.

After the literature study, the start of the plan as proposed in section 8.2 is carried out. This phase is a period of about thirteen weeks in which the basic model will be created. First of all, the detector and feature extractor will be built to obtain inputs for the classifiers. Secondly, the classifiers are built and trained that are the heart of the hear-and-avoid algorithm. The end of this phase is confirmed by the midterm meeting in which the complete model is elaborated on.

After the midterm, the testing, verification and validation of the created model is the next goal. Before the testing, also the adjustments that have to be done to make the algorithm ready for real time use on the drone. The testing procedure is elaborated on in section 8.3. Verifying and validating comes very naturally for supervised training methods. Due to the fact that all the data should be labeled, it is exactly known what comes in and what the model puts out, which would show whether the algorithm is verified or not. The validation procedure is explained in section 8.4. Testing shows whether the UAV changes position based on the obtained audio information. The hear-and-avoid algorithm is validated based on the precision and recall information obtained from the tests. If verification and validation have been performed the thesis should be finished, of which the green light meeting is the ultimate test.

If the green light meeting is successful, last period starts which consists of finalizing the report, the paper and preparing a thesis defence. If this is all finished correctly, the master of science is obtained.

The previously described planning is visualized in the Gantt Chart of Appendix A. Each phase is indicated with a different color and has multiple sub tasks. In black the milestones are indicated, which are the most significant meetings during the thesis. In the last period there are some open spaces, which are filled in by working towards the milestones. This could be report writing or preparations for the defence.

The whole thesis should take up nine months of full time work. The Gantt chart shows that there is also some time used as a buffer and holidays (in green), which means that including these extra weeks the thesis should be finished in the end of September.

9

Conclusions

This literature review explains the background information and the plan in order to come to the hear-and-avoid system, a system that can detect air traffic sound over the ego-sound of a drone, after which it performs avoidance maneuvers if necessary. Machine hearing, the field that tries to make robots hear like humans and animals do, should enable machines such as drones to use sound to base its actions upon. The system will consist of a detection module, a feature extraction module and a classification module. The detection module uses novelty detection in order to notice air traffic sound. The module captures energy variations in the signal. If this crosses an (adaptive) threshold, the sound is not considered background noise anymore but a sound of interest. This audio segment is passed on to the feature extractor. Two features will be obtained, the Spectrogram Image Feature feature and Mel-Frequency Cepstral Coefficients. Also raw audio signals have been proven to be a good input in a classifier and thus used as an input as well. The best performing feature (in combination with the right classifier) will be used for the hear-and-avoid system. The classifiers chosen are convolutional neural networks, long short-term memory and a combination of both. The classifier tries to identify whether the air traffic sound present is forming a danger for the drone (and therefore contrariwise also the drone for the air traffic). It can base its decision on a combination of the loudness, the frequency content and (the absence of) Doppler shift. The localization of the sound will be skipped in this research due to time restrictions and limitations in the different localization algorithms. Therefore, the avoidance manoeuvre of the drone is proposed to be a straight landing for now. The total length of the thesis is nine months. Including holidays and other reasons of delay, the graduation is expected in September.

With a successful development of the hear-and-avoid algorithm a next step is made in the safe use of drones, which is beneficial for the drone industry, but also the other airspace users as their safety is increased as well. For project Percevite it means one step closer to a sensor, communication and processing suite for small drones that does not need human intervention to avoid ground and air based vehicles.

Bibliography

[1] G.J.J. Ruijgrok. *Elements of Aviation Acoustics*. Delft University Press, 1993.

[2] Hallowell Davis and Sol Richard Silverman. *Hearing and deafness*. Holt, Rinehart & Winston of Canada Ltd, 1970.

[3] Jonathan William Dennis. Sound Event Recognition in Unstructured Environments using Spectrogram Image Processing. *Thesis*, (January), 2014.

[4] Alain Dufaux, Laurent Besacier, Michael Ansorge, and Fausto Pellandini. Automatic sound detection and recognition for noisy environment. In *Signal Processing Conference, 2000 10th European*, pages 1–4. IEEE, 2000.

[5] Fernando Pereira, Anthony Vetro, and Thomas Sikora. Multimedia retrieval and delivery: Essential metadata challenges and standards. *Proceedings of the IEEE*, 96(4):721–744, 2008.

[6] Yann Lecun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015.

[7] Nobuhide Yamakawa, Tetsuro Kitahara, Toru Takahashi, Kazunori Komatani, Tetsuya Ogata, and Hiroshi G Okuno. Effects of modelling within-and between-frame temporal variations in power spectra on non-verbal sound recognition. In *Eleventh Annual Conference of the International Speech Communication Association*, 2010.

[8] Francesc Alías, Joan Claudi Socoró, and Xavier Sevillano. A review of physical and perceptual feature extraction techniques for speech, music and environmental sounds. *Applied Sciences*, 6(5), 2016.

[9] Ian McLoughlin, Haomin Zhang, Zhipeng Xie, Yan Song, Wei Xiao, and Huy Phan. Continuous robust sound event classification using time-frequency features and deep learning. *PLoS ONE*, 12(9):e0182309, sep 2017.

[10] Zhipeng Xie, Ian Mcloughlin, Haomin Zhang, Yan Song, and Wei Xiao. A new variance-based approach for discriminative feature extraction in machine hearing classification using spectrogram features. *Digital Signal Processing: A Review Journal*, 54:119–128, 2016.

[11] Ian McLoughlin, Haomin Zhang, Zhipeng Xie, Yan Song, and Wei Xiao. Robust Sound Event Classification Using Deep Neural Networks. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 23(3):540–552, mar 2015.

[12] Huy Phan, Lars Hertel, Marco Maass, and Alfred Mertins. Robust audio event recognition with 1-max pooling convolutional neural networks. *Proceedings of the Annual Conference of the International Speech Communication Association, INTERSPEECH*, 08-12-Sept(1):3653–3657, 2016.

[13] Soo Hyun Bae, Inkyu Choi, and Nam Soo Kim. Acoustic scene classification using parallel combination of lstm and cnn. In *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2016 Workshop (DCASE2016)*, pages 11–15, 2016.

[14] S. Doljé. Quantifying microphone array directivity. Master's thesis, Delft University of Technology, dec 2017.

[15] Jonathan Dennis, Huy Dat Tran, and Haizhou Li. Spectrogram image feature for sound event classification in mismatched conditions. *IEEE Signal Processing Letters*, 18(2):130–133, 2011.

[16] Emre Cakir, Giambattista Parascandolo, Toni Heittola, Heikki Huttunen, and Tuomas Virtanen. Convolutional Recurrent Neural Networks for Polyphonic Sound Event Detection. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 25(6):1291–1303, jun 2017.

[17] Jie Huang, Tadawute Supaongprapa, Ikutaka Terakura, Fuming Wang, Noboru Ohnishi, and Noboru Sugie. A model-based sound localization system and its application to robot navigation. *Robotics and Autonomous Systems*, 27(4):199–209, 1999.

[18] Selina Chu, Shrikanth Narayanan, C-C Jay Kuo, and Maja J Mataric. Where am i? scene recognition for mobile robots using audio features. In *Multimedia and Expo, 2006 IEEE International Conference on*, pages 885–888. IEEE, 2006.

[19] Meysam Basiri, Felix Schill, Pedro U. Lima, and Dario Floreano. Robust acoustic source localization of emergency signals from Micro Air Vehicles. *IEEE International Conference on Intelligent Robots and Systems*, pages 4737–4742, 2012.

[20] Meysam Basiri and Felix Schill. Audio-based Relative Positioning System for Multiple Micro Air Vehicle Systems. *Robotics : Science and Systems*, (266470), 2013.

[21] Meysam Basiri, Felix Schill, Dario Floreano, and Pedro U Lima. Audio-based localization for swarms of micro air vehicles. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4729–4734. IEEE, may 2014.

[22] Meysam Basiri, Felix Schill, Pedro Lima, and Dario Floreano. On-Board Relative Bearing Estimation for Teams of Drones Using Sound. *IEEE Robotics and Automation Letters*, 1(2):820–827, 2016.

[23] Brendan Harvey and Siu O'Young. Acoustic Detection of a Fixed-Wing UAV. *Drones*, 2(1):4, jan 2018.

[24] E Tijs, GCHE de Croon, J Wind, B Remes, C De Wagter, HE de Bree, and R Ruijsink. Hear-and-avoid for micro air vehicles. In *Proceedings of the International Micro Air Vehicle Conference and Competitions (IMAV), Braunschweig, Germany*, volume 69, 2010.

[25] Hans-elias De Bree and Guido De Croon. Acoustic Vector Sensors on Small Unmanned Air Vehicles. *SMi Unmanned Aircraft Systems*, (November 2011), 2011.

[26] Michael JT Smith. *Aircraft noise*, volume 3. Cambridge University Press, 2004.

[27] Lawrence E Kinsler, Austin R Frey, Alan B Coppens, and James V Sanders. Fundamentals of acoustics. *Fundamentals of Acoustics, 4th Edition, by Lawrence E. Kinsler, Austin R. Frey, Alan B. Coppens, James V. Sanders, pp. 560. ISBN 0-471-84789-5. Wiley-VCH, December 1999.*, page 560, 1999.

[28] Aircraft noise. Technical report, Oakland International Airport, mar 2006.

[29] HE Bass, LC Sutherland, Joe Piercy, and Landon Evans. Absorption of sound by the atmosphere. In *IN: Physical acoustics: Principles and methods. Volume 17 (A85-28596 12-71). Orlando, FL, Academic Press, Inc., 1984, p. 145-232.*, volume 17, pages 145–232, 1984.

[30] H. E. Bass, L. C. Sutherland, and a. J. Zuckerwar. Atmospheric absorption of sound: Update. *The Journal of the Acoustical Society of America*, 88(4):2019–2021, oct 1990.

[31] H. E. Bass, L. C. Sutherland, A. J. Zuckerwar, D. T. Blackstock, and D. M. Hester. Atmospheric absorption of sound: Further developments. *The Journal of the Acoustical Society of America*, 97(1):680–683, jan 1995.

[32] R Spiegel Murray. Advanced mathematics for engineers and scientists. *McGrew Hill, USA*, pages 375–384, 1980.

[33] Steven W Smith et al. The scientist and engineer's guide to digital signal processing. 1997.

[34] William T Cochran, James W Cooley, David L Favin, Howard D Helms, Reginald A Kaenel, William W Lang, GC Maling, David E Nelson, Charles M Rader, and Peter D Welch. What is the fast fourier transform? *Proceedings of the IEEE*, 55(10):1664–1674, 1967.

[35] Curtis Roads and John Strawn. *The computer music tutorial*. MIT press, 1996.

[36] Alan V Oppenheim and Ronald W Schafer. From frequency to quefrency: A history of the cepstrum. *IEEE signal processing Magazine*, 21(5):95–106, 2004.

[37] Peter W Alberti. The anatomy and physiology of the ear and hearing. *Occupational exposure to noise: Evaluation, prevention, and control*, pages 53–62, 2001.

[38] Irwin Pollack and James M Pickett. Cocktail party effect. *The Journal of the Acoustical Society of America*, 29(11):1262–1262, 1957.

[39] Barry Arons. A review of the cocktail party effect. *Journal of the American Voice I/O Society*, 12(7):35–50, 1992.

[40] Jinhai Cai, Dominic Ee, Binh Pham, Paul Roe, and Jinglan Zhang. Sensor network for the monitoring of ecosystem: Bird species recognition. In *Intelligent Sensors, Sensor Networks and Information, 2007. ISSNIP 2007. 3rd International Conference on*, pages 293–298. IEEE, 2007.

[41] Andrey Temko, Robert Malkin, Christian Zieger, Dušan Macho, Climent Nadeu, and Maurizio Omologo. Clear evaluation of acoustic event detection and classification systems. In *International Evaluation Workshop on Classification of Events, Activities and Relationships*, pages 311–322. Springer, 2006.

[42] Tuomas Virtanen, Mark D. Plumbley, and Dan Ellis. *Computational Analysis of Sound Scenes and Events*. Springer International Publishing, Cham, 2018.

[43] Samantha J Barry, Adrie D Dane, Alyn H Morice, and Anthony D Walmsley. The automatic recognition and counting of cough. *Cough*, 2(1):8, 2006.

[44] Ya-Ti Peng, Ching-Yung Lin, Ming-Ting Sun, and Kun-Cheng Tsai. Healthcare audio event classification using hidden markov models and hierarchical hidden markov models. In *Multimedia and Expo, 2009. ICME 2009. IEEE International Conference on*, pages 1218–1221. IEEE, 2009.

[45] Chloé Clavel, Thibaut Ehrette, and Gaël Richard. Events detection for an audio-based surveillance system. In *Multimedia and Expo, 2005. ICME 2005. IEEE International conference on*, pages 1306–1309. IEEE, 2005.

[46] Luigi Gerosa, Giuseppe Valenzise, Marco Tagliasacchi, Fabio Antonacci, and Augusto Sarti. Scream and gunshot detection in noisy environments. In *Signal Processing Conference, 2007 15th European*, pages 1216–1220. IEEE, 2007.

[47] Aki Harma, Martin F McKinney, and Janto Skowronek. Automatic surveillance of the acoustic activity in our living environment. In *Multimedia and Expo, 2005. ICME 2005. IEEE International Conference on*, pages 4–pp. IEEE, 2005.

[48] Rolf Bardeli, Daniel Wolff, Frank Kurth, Martina Koch, K-H Tauchert, and K-H Frommolt. Detecting bird sounds in a complex acoustic environment and application to bioacoustic monitoring. *Pattern Recognition Letters*, 31(12):1524–1534, 2010.

[49] Felix Weninger and Björn Schuller. Audio recognition in the wild: Static and dynamic classification on a real-world database of animal vocalizations. In *acoustics, speech and signal processing (ICASSP), 2011 IEEE international conference on*, pages 337–340. IEEE, 2011.

[50] Sourabh Ravindran and David V Anderson. Audio classification and scene recognition and for hearing aids. In *Circuits and Systems, 2005. ISCAS 2005. IEEE International Symposium on*, pages 860–863. IEEE, 2005.

[51] Andriy Temko. *Acoustic event detection and classification*. PhD thesis, Universitat Politècnica de Catalunya, 2008.

[52] Dalibor Mitrović, Matthias Zeppelzauer, and Christian Breiteneder. Features for Content-Based Audio Retrieval. *Advances in Computers*, 78(10):71–150, 2010.

[53] Javier Ramirez, Juan Manuel Górriz, and José Carlos Segura. Voice activity detection. fundamentals and speech recognition system robustness. In *Robust speech recognition and understanding*. InTech, 2007.

[54] Jose Portelo, Miguel Bugalho, Isabel Trancoso, Joao Neto, Alberto Abad, and Antonio Serralheiro. Non-speech audio event detection. In *Acoustics, Speech and Signal Processing, 2009. ICASSP 2009. IEEE International Conference on*, pages 1973–1976. IEEE, 2009.

[55] Derek Hoiem, Yan Ke, and Rahul Sukthankar. Solar: Sound object localization and retrieval in complex audio environments. In *Acoustics, Speech, and Signal Processing, 2005. Proceedings.(ICASSP'05). IEEE International Conference on*, volume 5, pages v–429. IEEE, 2005.

[56] Brian McFee, Colin Raffel, Dawen Liang, Daniel PW Ellis, Matt McVicar, Eric Battenberg, and Oriol Nieto. librosa: Audio and music signal analysis in python. In *Proceedings of the 14th python in science conference*, pages 18–25, 2015.

[57] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[58] Benjamin Kedem. Spectral analysis and discrimination by zero-crossings. *Proceedings of the IEEE*, 74(11):1477–1493, 1986.

[59] Vesa Peltonen, Juha Tuomi, Anssi Klapuri, Jyri Huopaniemi, and Timo Sorsa. Computational auditory scene recognition. In *Acoustics, speech, and signal processing (icassp), 2002 IEEE international conference on*, volume 2, pages II–1941. IEEE, 2002.

[60] Rui Cai, Lie Lu, Alan Hanjalic, Hong-Jiang Zhang, and Lian-Hong Cai. A flexible framework for key audio effects detection and auditory context inference. *IEEE Transactions on audio, speech, and language processing*, 14(3):1026–1039, 2006.

[61] Asma Rabaoui, Manuel Davy, Stéphane Rossignol, and Noureddine Ellouze. Using one-class svms and wavelets for audio surveillance. *IEEE Transactions on information forensics and security*, 3(4):763–775, 2008.

[62] Dalibor Mitrovic, Matthias Zeppelzauer, and Christian Breiteneder. Discrimination and retrieval of animal sounds. In *Multi-Media Modelling Conference Proceedings, 2006 12th International*, pages 5–pp. IEEE, 2006.

[63] Ghulam Muhammad and Khaled Alghathbar. Environment recognition from audio using mpeg-7 features. In *Embedded and Multimedia Computing, 2009. EM-Com 2009. 4th International Conference on*, pages 1–6. IEEE, 2009.

[64] Yin-Fu Huang, Sheng-Min Lin, Huan-Yu Wu, and Yu-Siou Li. Music genre classification based on local feature selection using a self-adaptive harmony search algorithm. *Data & Knowledge Engineering*, 92:60–76, jul 2014.

[65] X Valero and F Alías. Applicability of mpeg-7 low level descriptors to environmental sound source recognition. In *Proceedings 1st Euroregio Conference, Ljubjana*, 2010.

[66] EnShuo Tsau, Seung-Hwan Kim, and C-C Jay Kuo. Environmental sound recognition with celp-based features. In *Signals, Circuits and Systems (ISSCS), 2011 10th International Symposium on*, pages 1–4. IEEE, 2011.

[67] Fumitada Itakura. Line spectrum representation of linear predictor coefficients of speech signals. *The Journal of the Acoustical Society of America*, 57(S1):S35–S35, 1975.

[68] Bruce P Bogert. The quefrency analysis of time series for echoes: cepstrum, pseudo-autocovariance, cross-cepstrum and saphe cracking. In *Proc. Symposium on Time Series Analysis, 1963*. John Wiley & Sons, 1963.

[69] F Beritelli and R Grasso. A pattern recognition system for environmental sound classification based on mfccs and neural networks. In *Signal Processing and Communication Systems, 2008. ICSPCS 2008. 2nd International Conference on*, pages 1–4. IEEE, 2008.

[70] Roy D Patterson, John Holdsworth, and Michael Allerhand. Auditory models as preprocessors for speech recognition. *The Auditory Processing of Speech: from Auditory Periphery to Words*, pages 67–89, 1992.

[71] Roy D Patterson, K Robinson, J Holdsworth, D McKeown, C Zhang, and M Allerhand. Complex sounds and auditory images. In *Auditory physiology and perception*, pages 429–446. Elsevier, 1992.

[72] George Trigeorgis, Fabien Ringeval, Raymond Brueckner, Erik Marchi, Mihalis A. Nicolaou, Bjorn Schuller, and Stefanos Zafeiriou. Adieu features? End-to-end speech emotion recognition using a deep convolutional recurrent network. *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5200–5204, 2016.

[73] Wei Dai, Chia Dai, Shuhui Qu, Juncheng Li, and Samarjit Das. Very Deep Convolutional Neural Networks for Raw Waveforms. pages 3–7, 2016.

[74] Tara N. Sainath, Ron J. Weiss, Andrew Senior, Kevin W. Wilson, and Oriol Vinyals. Learning the speech front-end with raw waveform CLDNNs. *Proceedings of the Annual Conference of the International Speech Communication Association, INTERSPEECH*, 2015-Janua:1–5, 2015.

[75] Yusuf Aytar, Carl Vondrick, and Antonio Torralba. SoundNet: Learning Sound Representations from Unlabeled Video. (Nips), 2016.

[76] Joaquin García-Gomez, Marta Bautista-Durán, Roberto Gil-Pita, and Manuel Rosa-Zurera. Feature Selection for Real-Time Acoustic Drone Detection Using Genetic Algorithms. In *Audio Engineering Society Convention 142*, 2017.

[77] Nidhi Desai, Kinnal Dhameliya, and Vijayendra Desai. Feature Extraction and Classification Techniques for Speech Recognition: A Review. *International Journal of Emerging Technology and Advanced Engineering Website: www.ijetae.com ISO Certified Journal*, 9001(12):1–5, 2013.

[78] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. Scikit-learn: Machine learning in python. *Journal of machine learning research*, 12(Oct):2825–2830, 2011.

[79] François Chollet et al. Keras, 2015.

[80] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. Tensorflow: A system for large-scale machine learning. In *OSDI*, volume 16, pages 265–283, 2016.

[81] Keunwoo Choi, Deokjin Joo, and Juho Kim. Kapre: On-gpu audio preprocessing layers for a quick implementation of deep neural network models with keras. *arXiv preprint arXiv:1706.05781*, 2017.

[82] Lawrence R Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989.

[83] Kevin Patrick Murphy and Stuart Russell. Dynamic bayesian networks: representation, inference and learning. 2002.

[84] Taras Butko. *Feature selection for multimodal: acoustic Event detection*. Universitat Politècnica de Catalunya, 2011.

[85] Andrew J Viterbi. Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. In *The Foundations Of The Digital Wireless World: Selected Works of AJ Viterbi*, pages 41–50. World Scientific, 2010.

[86] Christopher JC Burges. A tutorial on support vector machines for pattern recognition. *Data mining and knowledge discovery*, 2(2):121–167, 1998.

[87] Olvi L Mangasarian and Edward W Wild. Proximal support vector machine classifiers. In *Proceedings KDD-2001: Knowledge Discovery and Data Mining*. Citeseer, 2001.

[88] Mercedes Fernández-Redondo and Carlos Hernandez-Espinosa. Weight initialization methods for multilayer feedforward. In *ESANN*, pages 119–124, 2001.

[89] Celso AR de Sousa. An overview on weight initialization methods for feedforward neural networks. In *Neural Networks (IJCNN), 2016 International Joint Conference on*, pages 52–59. IEEE, 2016.

[90] Yoshua Bengio, Patrice Simard, and Paolo Frasconi. Learning long-term dependencies with gradient descent is difficult. *IEEE transactions on neural networks*, 5(2):157–166, 1994.

[91] Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. On the difficulty of training recurrent neural networks. In *International Conference on Machine Learning*, pages 1310–1318, 2013.

[92] Tara N Sainath, Oriol Vinyals, Andrew Senior, and Haşim Sak. Convolutional, long short-term memory, fully connected deep neural networks. In *Acoustics, Speech and Signal Processing (ICASSP), 2015 IEEE International Conference on*, pages 4580–4584. IEEE, 2015.

[93] Dario Amodei, Sundaram Ananthanarayanan, Rishita Anubhai, Jingliang Bai, Eric Battenberg, Carl Case, Jared Casper, Bryan Catanzaro, Qiang Cheng, Guoliang Chen, et al. Deep speech 2: End-to-end speech recognition in english and mandarin. In *International Conference on Machine Learning*, pages 173–182, 2016.

[94] Keunwoo Choi, György Fazekas, Mark Sandler, and Kyunghyun Cho. Convolutional recurrent neural networks for music classification. In *Acoustics, Speech and Signal Processing (ICASSP), 2017 IEEE International Conference on*, pages 2392–2396. IEEE, 2017.

[95] Satoshi Nakamura, Kazuo Hiyane, Futoshi Asano, Takanobu Nishiura, and Takeshi Yamada. Acoustical sound database in real environments for sound scene understanding and hands-free speech recognition. In *LREC*, 2000.

[96] Andrew L Maas, Quoc V Le, Tyler M O'Neil, Oriol Vinyals, Patrick Nguyen, and Andrew Y Ng. Recurrent neural networks for noise reduction in robust asr. In *Thirteenth Annual Conference of the International Speech Communication Association*, 2012.

[97] Oriol Vinyals, Suman V Ravuri, and Daniel Povey. Revisiting recurrent neural networks for robust asr. In *Acoustics, Speech and Signal Processing (ICASSP), 2012 IEEE International Conference on*, pages 4085–4088. IEEE, 2012.

[98] Felix Weninger, Hakan Erdogan, Shinji Watanabe, Emmanuel Vincent, Jonathan Le Roux, John R Hershey, and Björn Schuller. Speech enhancement with lstm recurrent neural networks and its application to noise-robust asr. In *International Conference on Latent Variable Analysis and Signal Separation*, pages 91–99. Springer, 2015.

[99] Dimitrios Giannoulis, Emmanouil Benetos, Dan Stowell, Mathias Rossignol, Mathieu Lagrange, and Mark D Plumbley. Detection and classification of acoustic scenes and events: An ieee aasp challenge. In *Applications of Signal Processing to Audio and Acoustics (WASPAA), 2013 IEEE Workshop on*, pages 1–4. IEEE, 2013.

[100] Ralph Schmidt. Multiple emitter location and signal parameter estimation. *IEEE transactions on antennas and propagation*, 34(3):276–280, 1986.

[101] ZI Khan, M MD Kamal, N Hamzah, K Othman, and NI Khan. Analysis of performance for multiple signal classification (music) in estimating direction of arrival. In *RF and Microwave Conference, 2008. RFM 2008. IEEE International*, pages 524–529. IEEE, 2008.

[102] Fawwaz Alsubaie. Multiple signal classification for determining direction of arrival of frequency hopping spread spectrum signals. Technical report, Air Force Institute of Technology, 2014.

[103] Lloyd A Jeffress. A place theory of sound localization. *Journal of comparative and physiological psychology*, 41(1):35, 1948.

[104] Sylvain Argentieri, P. Danès, and P. Souères. A survey on sound source localization in robotics: From binaural to array processing methods. *Computer Speech & Language*, 34(1):87–112, nov 2015.

[105] Charles Knapp and Glifford Carter. The generalized correlation method for estimation of time delay. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 24(4):320–327, 1976.

[106] Peter Welch. The use of fast fourier transform for the estimation of power spectra: a method based on time averaging over short, modified periodograms. *IEEE Transactions on audio and electroacoustics*, 15(2):70–73, 1967.

[107] Geetanjali U. Mandewalkar, Subhash S. Kulkarni, S. Veena, and H. Lokesha. Real time acoustic source localization of emergency signals. *2014 International Conference on Advances in Electronics, Computers and Communications, ICAECC 2014*, 2015.

[108] John C Middlebrooks and David M Green. Sound localization by human listeners. *Annual review of psychology*, 42(1):135–159, 1991.

[109] Beverly A Wright and Matthew B Fitzgerald. Different patterns of human discrimination learning for two interaural cues to sound-source location. *Proceedings of the National Academy of Sciences*, 98(21):12307–12312, 2001.

[110] Vikas C Raykar, Ramani Duraiswami, and B Yegnanarayana. Extracting the frequencies of the pinna spectral notches in measured head related impulse responses. *The Journal of the Acoustical Society of America*, 118(1):364–374, 2005.

[111] Brian G. Ferguson. Application of the short-time Fourier transform and the Wigner–Ville distribution to the acoustic localization of aircraft. *J. Acoust. Soc. Am.*, 96(2):821–827, 1994.

[112] Brian G. Ferguson. A ground-based narrow-band passive acoustic technique for estimating the altitude and speed of a propeller-driven aircraft. *The Journal of the Acoustical Society of America*, 92(3):1403–1407, 1992.

[113] J Schiller. Motion parameter estimation of a sound source using the multipath interference of the radiated noise. In *Signal Processing II: Theories and Applications. In Proceedings of EUSIPCO-83 Second European Signal Processing Conference*, pages 661–664, 1983.

[114] K.W. Lo, S.W. Perry, and B.G. Ferguson. Aircraft flight parameter estimation using acoustical Lloyd's mirror effect. *IEEE Transactions on Aerospace and Electronic Systems*, 38(1):137–151, 2002.

[115] James M. King and Imraan Faruque. Small Unmanned Aerial Vehicle Passive Range Estimation from a Single Microphone. In *AIAA Atmospheric Flight Mechanics Conference*, pages 1–6, Reston, Virginia, jun 2016. American Institute of Aeronautics and Astronautics.

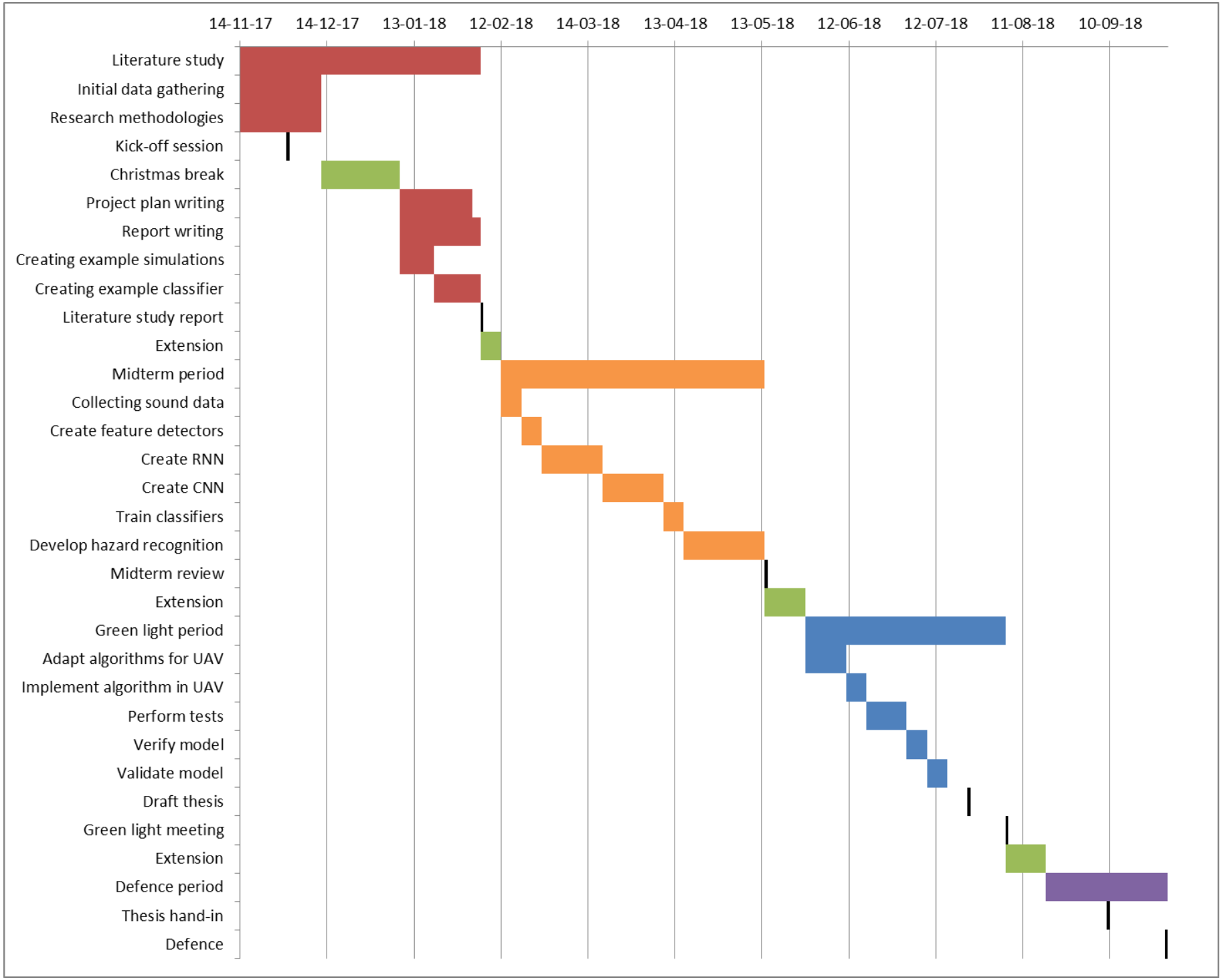
[116] Futoshi Asano, Mitsuharu Morisawa, Kenji Kaneko, and Kazuhito Yokoi. Sound source localization using a single-point stereo microphone for robots. *2015 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference, APSIPA ASC 2015*, (December):76–85, 2016.

[117] J. E. Piercy, T. F. W. Embleton, and L. C. Sutherland. Review of noise propagation in the atmosphere. *The Journal of the Acoustical Society of America*, 61(6):1403–1418, jun 1977.

A

Gantt Chart

The figure below shows the Gantt chart for this thesis. It exists of four periods: the literature study (red), midterm period (orange), green light period (blue) and the defence period (purple). In black the milestones are indicated and in green the holidays and buffers are shown.


14-11-17
14-12-17
13-01-18
12-02-18
14-03-18
13-04-18
13-05-18
12-06-18
12-07-18
11-08-18
10-09-18
Literature study
Initial data gathering
Research methodologies
Kick-off session
Christmas break
Project plan writing
Report writing
Creating example simulations
Creating example classifier
Literature study report
Extension
Midterm period
Collecting sound data
Create feature detectors
Create RNN
Create CNN
Train classifiers
Develop hazard recognition
Midterm review
Extension
Green light period
Adapt algorithms for UAV
Implement algorithm in UAV
Perform tests
Verify model
Validate model
Draft thesis
Green light meeting
Extension
Defence period
Thesis hand-in
Defence