UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH
Escola Tècnica Superior d'Enginyeria
de Telecomunicació de Barcelona

Supervised time-delay estimation for the passive acoustic localization of cetaceans

A Master's Thesis

Submitted to the Faculty of the

Escola Tècnica d'Enginyeria de Telecomunicació de Barcelona

Universitat Politècnica de Catalunya

by

Eduardo Cuesta Lázaro

In partial fulfilment

of the requirements for the degree of

MASTER IN TELECOMMUNICATIONS ENGINEERING

Advisors: Ludwig Houégnigan and Climent Nadeu

Barcelona, October 2017

Supervised time-delay estimation for the passive acoustic localization of cetaceans

Eduardo Cuesta

October 19, 2017

Abstract

Time-delay estimation is an essential part of a wide variety of signal processing applications. This paper follows up on earlier work for time-delay estimation using neural networks. Nonetheless, this work is specialized in the passive acoustic localization of cetaceans. We built a time-delay database from real cetacean vocalizations. Afterwards, we implemented a supervised estimation, based on high-level features and convolutional neural networks. These features are especially designed to deal with the high dimensionality of the cetaceans vocalizations. Finally, we show that our method outperforms traditional approaches when dealing with a realistic dataset which contains large amounts of noise.

Contents

1 Introduction **13**

1.1 Motivations 13

1.2 Objectives 14

1.3 Cetaceans bio-acoustics 15

1.3.1 Passive acoustic monitoring 15

1.3.2 Cetaceans classification 15

1.3.3 Vocalizations 16

2 State-of-the-art **19**

2.1 Time-delay estimation 20

2.2 Data modeling 21

2.2.1 Signal models 22

2.2.2 The family of the GCC methods 24

2.2.3 Adaptive eigenvalue decomposition algorithm (AED) 27

2.3 Algorithmic modeling 30

2.3.1 Nominal continuous estimation 31

2.3.2 Multidimensional discrete estimation 33

3 Database **37**

3.1 Motivation 37

3.2 Objectives 38

3.3 Design 38

3.3.1 Simulation scenario 40

3.3.2 Segment assembling 42
3.3.3 Noise embedding 44
3.3.4 Labeling 45
3.3.5 Datasets generation 47
3.4 Implementation 48
3.4.1 Spectrogram reconstruction 48
3.4.2 Contour reconstruction 51
3.4.3 Time-delay data generation 53
3.4.4 Labeling 55

4 Time-delay estimation **57**
4.1 Earliest attempt 57
4.1.1 Curse of dimensionality 59
4.2 Final design 60
4.2.1 Feature extraction 60
4.2.2 Architecture 62
4.2.3 AlexNet-like network 62
4.2.4 Fully convolutional network 64
4.2.5 Training procedure 64
4.2.6 Improving generalization 65

5 Results **67**
5.1 Methodology 67
5.2 Computation time 68
5.3 First experiment 69
5.3.1 First overview 69
5.3.2 Statistical significance 70
5.4 Second experiment 71

6 Conclusions and future development **73**
6.1 Conclusions 73

6.2 Future development 73

A Figures **75**

List of Figures

2-1 Transmission loss estimation using Bellhop 19

2-2 Statistical modeling. Figure credit from [23] 21

2-3 Sea reflections. Figure credit from [11] 23

2-4 Regression architecture of [26]. Figure credit from [26] 32

2-5 Figure credit from error histogram for noise std = 0.5. Figure credit from [27] 33

2-6 Error distribution of various estimators for variable SNR $\in [5, 1]$. Figure credit from [15] 36

3-1 Array arrangement 40

3-2 Fraunhofer distance versus frequency 41

3-3 Segment lengths distribution along species 43

3-4 Segment structure 44

4-1 CNN siamese network graph 58

4-2 Local spectrogram cross-correlation 60

4-3 Local spectrogram cross-correlation (coherent-print features) of a narrow band whistle under favourable conditions (SNR = 0 dB) 61

5-1 Part of the graph of the fully convolutional network. More demanding nodes, in terms of computation time, appears more reddish 69

5-2 Output of GCC and CNN1 estimators with respect to the target time-delay. Input signals from dataset 2 at SNR = -6 dB, time-delay = -95 ms 70

5-3 Output of PHAT and CNN estimators with respect to the target time-delay. Input signals from dataset 2 at SNR = -6 dB, time-delay = -95 ms 70

A-1 Output of the neural network and cross-correlation estimators with respect to target. Figure credit from [15] 76
A-2 Time resolutions pdf 77
A-3 Annotations to spectrogram 78
A-4 Bottlenose dolphin's whistle labeled as a time-frequency contour . . . 79
A-5 Generated 2-channel time-delay segment 80
A-6 Generated multi-channel time-delay segment 81
A-7 Building segments 82
A-9 Boxplot of the error of various estimators 84
A-10 Boxplot of the error of various estimators 85
A-11 Distribution of error for varying SNR 86

List of Tables

2.1 Signal-to-noise ratio in linear scale of each dataset 34

2.2 Mean and standard deviation of the error of [15] as a function of the SNR 35

2.3 Q_{KL} pseudo-distance from the neural network output (MLP) and cross-correlation output (XCOR) to the target $\delta(n - \tau_{12})$ 36

3.1 Time-delay resolution 46

3.2 Distribution of the error in samples vs distribution label std 47

3.3 Datasets' features 47

5.1 Mean μ_e and standard deviation of the error in milliseconds as a function of the SNR (dB) 71

5.2 Mean μ_e and standard deviation of the error in milliseconds as a function of the SNR (dB) 71

Chapter 1

Introduction

Time-delay estimation is an essential part of a wide variety of signal processing applications, such as sonar and radar direction finding, speech processing, seismology, neuromedicine, satellite navigation or bio-acoustics.

The recent growth of machine learning approaches reformulates the classical signal processing techniques, working around its limitations given by its assumed model. This thesis follows up on earlier work for acoustic source localization and time-delay estimation using supervised techniques [15, 16]. It is specialized in the localization of cetaceans, who are species of conservation concern and show interesting acoustic characteristics.

1.1 Motivations

In [15], it was shown that artificial neural networks can outperform classical methods in different aspects:

1. The great majority of time-delay estimation methods have demonstrated their optimality with random signals at high SNR. On the other hand, in real scenarios, when SNR is usually low and signals are far from random, machine learning tools may exploit the statistical structure of the data.

2. Classical signal processing approaches stand on models which normally do not

properly represent the complexity of sound propagation in real environments, such as reverberation. In realistic cases, supervised approaches should obtain more precise results.

3. In localization applications many effort is put on the tracking algorithm, such as Extended and Unscented Kalman Filters, or particle filters [14]. Nonetheless, a more accurate and robust time-delay estimation will simplify the task of the tracking algorithm.

Another important motivation is the lack of supervised methods developed for time-delay estimation, except some recent work [15, 16, 26, 27]. In this thesis, we focus on improving these methods for real scenarios.

Furthermore, since many cetacean species of interest are easier to hear than to see, underwater time-delay estimation is an area of growing attraction. Moreover, some of these bio-acoustic signals have desirable transmission properties.

1.2 Objectives

The main goal is to apply signal processing and machine learning tools to acoustic time-delay estimation for cetacean localization, looking for a reliable estimator in pseudo-real conditions. Furthermore, rather than trying to model the whole propagation system, we exploit the statistical structure of the corrupted observation signals. Therefore, our goals are,

1. To design a controlled database. It should reflect the real data structure while keeping under control complexity parameters, such as noise, range of animal species, special cases...

2. To find a suitable estimator.

    2.1. Robust features are extracted via signal processing techniques.

    2.2. A multidimensional time-delay response is estimated from these features with neural networks.

1.3 Cetaceans bio-acoustics

It is essential to understand the peculiarities of the acoustic signals treated in this work, as they determine the design of the database, and by extension, the architecture of the time-delay estimator.

Moreover, there is a strong motivation for localizing cetaceans through the sounds they produce. Back in 1930's, for the first time small cetaceans were successfully kept in captivity, and an impressive vocal behaviour was noticed. It was however no surprising; through a long evolution cetaceans have adapted to their environment and developed adequate and efficient techniques for navigation, prey localization and communication using sound. Likewise, this active acoustic behaviour allows us the remote study of these animals, who cover vast ocean areas and are rather evasive.

1.3.1 Passive acoustic monitoring

Passive acoustic monitoring (PAM) is an essential tool for surveying and studying cetaceans, since they use sound in their ordinary activities. These sounds are, so far, the only feasible mechanism to sense underwater life in hostile sea environments. Furthermore, passive monitoring does not disturb the animals, unlike active acoustic technologies which require high radiation of energy.

We propose a passive acoustic time-delay estimation for the localization of cetaceans, sensing the acoustic output from these animals in a non-invasive manner. In fact, this procedure is specialized in some specific vocalizations (Section 1.3.3), approaching all cetaceans species capable to produce tonal sounds.

1.3.2 Cetaceans classification

Before attempting to enter in the cetacean acoustic framework, it is worth to state some terminology associated to its biological classification.

The taxonomic order Cetacea consist of whales, dolphins and porpoises, which is divided into two extant suborders, Odontoceti (toothed whales) and Mysticeti (baleen whales or mustache whales). These are further divided into families, e.g. Delphinidae

(ocean dolphins) or Ziphiidae (beaked whales). Within families, cetaceans are classified into genera and species. For our purposes, only suborder taxonomies, odontocetes and mysticetes, are considered since their vocalizations are remarkably different.

Within the odontocetes suborder, dolphins are its main exponents, but the orca "killer whale" -who is actually in the dolphin family- and the sperm whale are also in this category. On the other side, some examples of mysticetes are the minke whale, the blue whale and the right whale.

1.3.3 Vocalizations

Earliest studies of cetacean vocalizations surveyed bottlenose dolphins and registered a wide variety of sounds they produce: 'whistles, rasping and grating sounds, rasping sounds, barks, and yelps' are some of the words used to describe them [31]. So far, despite the terminology being extensive and quite subjective, the cetacean vocalizations have been properly classified either by its biological function or by its acoustic characteristics.

In general, from an acoustics standpoint, these sounds can be considered either periodic, or aperiodic (rather short pulse-like). While tonal sounds are used for communication, shorter pulses are related to echolocation. However, there is no precise matching between signal properties and biological function; there are pulses with very short interclick intervals used for communication and short tonal signals emitted for echolocation, among other examples [32].

For our concerns, this classification can be summarized into two major categories: echolocation clicks and communication whistles.

The former are emitted by toothed whales (Odontoceti) during foraging. Baleen whales (Mysticeti), however, do not produce them. These clicks are highly directional broadband pulses of short duration.

On the other side, communication whistles are uttered by dolphins and baleen whales[1]. They are low directional, continuous frequency modulated signals, that often have harmonic components.

[1]Some toothed whales, dolphins, and porpoises do not produce whistles though [24]

Moreover, this signal characterization reveals the advantages of using communication whistles over echolocation clicks, for passive mammal cetacean localization:

1. Directivity of clicks renders cetacean localization more difficult to us; as the angle between the click's emitter and the sensor increases, the signal becomes attenuated and distorted. Therefore, it cannot simultaneously reach very sparse hydrophones. On the other hand, whistles propagate in a uniform way, allowing its detection in a wider coverage area.

2. Communication whistles are larger and have a more rich and dynamic spectral content to convey the information to the receiver in a noisy or complex environment [32].

In fact, in most areas, whistle sounds propagate much further than echolocation clicks, so whistles are likely to be more useful for long-distance applications [25]. Time-delay estimation is a clear example of this, where it is vital to synchronously sense the signal at various hydrophones, ideally placed far away from each other.

Chapter 2

State-of-the-art

Time-delay estimation (TDE) is the first stage of many signal processing systems, from direction of arrival, range estimation and tracking to motion compensation in moving images and stereo vision. The estimation covered in this work is specialized in the localization of cetaceans, and may be followed by a series of processing blocks, such as multi-hydrophone ranging, triangulation and transmission loss modelling. While the two former procedures have well-established analytical solutions, transmission loss modelling is estimated as a complicated function of the environmental parameters (Figure 2-1).

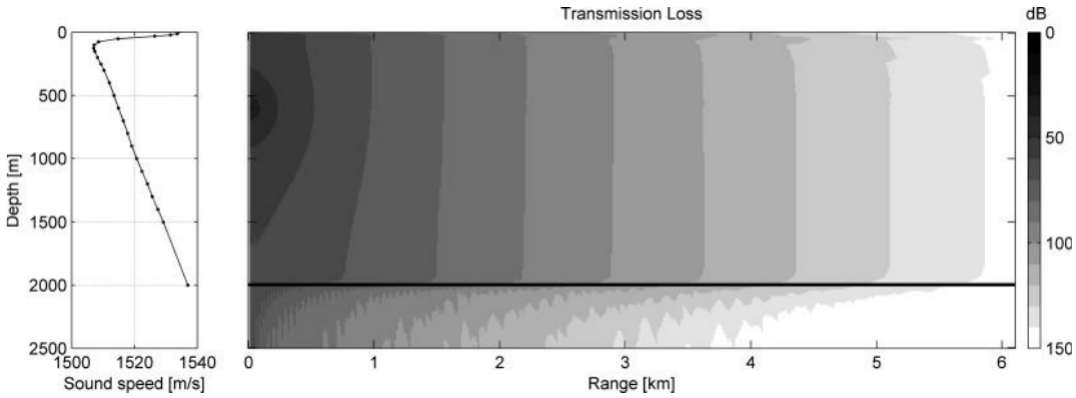


Figure 2-1: Transmission loss estimation using Bellhop

The efficiency of the whole system is limited by its first processing stage. Therefore, an accurate time-delay estimation is required in order to produce precise results.

2.1 Time-delay estimation

Depending on the application, TDE is known either as time of arrival (TOA) estimation or as time difference of arrival (TDOA) estimation. Active sonar and radar applications measure the time of arrival of the resulted echo of a transmitted pulse signal, whereas time difference of arrival is passively estimated as the travel-time-delay of a wavefront between two spatially separated receivers. The latter approach is the aim of this study, since it can be considered a PAM system (Section 1.3.1).

Both methods, despite being closely related, are intrinsically based on opposing principles. Time of arrival assumes that the concerned signal $s[n]$ is known, so it can be written as in Equation 2.1 under ideal free-field conditions. Therefore, the time-delay can be estimated on a single sensor, from $x[n]$ and $s[n]$, commonly by the matched filter approach.

$$x[n] = \alpha s[n - \tau] + w[n] \ , \qquad n = 0, 1, \ldots, N - 1 \tag{2.1}$$

Conversely, through passive estimation the reference signal $s[n]$ is no longer available beforehand. In this case, the time difference of arrival (Equation 2.3) is computed by comparing the signals, x_1 and x_2, received at two spatially spaced sensors.

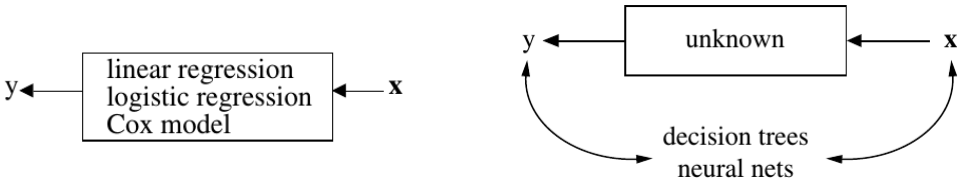
$$\begin{aligned} x_1[n] &= \alpha_1 s[n - \tau_1] + w[n] \\ x_2[n] &= \alpha_2 s[n - \tau_2] + w[n] \end{aligned} \tag{2.2}$$

$$\tau_{12} = \tau_1 - \tau_2 \tag{2.3}$$

A wide variety of methods have been deployed since sensor arrays were introduced to measure propagating wave-fields. It is beyond the scope of this section to review all of them, we rather describe several approaches from different backgrounds to contextualize the supervised estimation developed in this work.

The methods herein presented can be characterized by how they approach time-delay estimation[1]. Data modeling emphasizes inference; to get insights into the pro-

[1]See [23] for a detailed discussion between data and algorithmic modeling



(a) Data modeling (b) Algorithm modeling

Figure 2-2: Statistical modeling. Figure credit from [23]

cess by which time-delay is generated (Figure 2-2a). On the other hand, algorithmic modeling, machine learning approaches, focus directly on learning from data to make predictions, without attempting to understand the underlying process[2] (Figure 2-2b).

2.2 Data modeling

Time-delay estimators that are derived from a mathematical model, lie on this category. Most approaches are built upon the ideal free-field model (Equation 2.1), which allows to develop a close-form solution. Nonetheless, more complex propagation models are also considered.

Historically, this problem was approached from the perspective of estimation theory, so several unbiased estimators were evaluated in terms of variance [19][7][8][21]. However, there are sub-optimum methods in terms of approaching the CRLB, which perform better in real environments, since they suffer less from non-modeled factors[3]. This is a direct consequence of analyzing the potential of a method through its radical model; the conclusions are about the model's mechanism, and not about nature's mechanism [23].

Alternatively, recent studies improve substantially traditional approaches, such as [10], where the popular GCC method is revised and outperformed. Especially when facing reverberation and noise, by exploiting redundancy among multiple channels.

[2]It does not imply that no understanding can be extracted from algorithmic modeling. This only means that its main goal are the predictions, rather than the inference of the generation process

[3]For instance, the phase transform (PHAT), which is a sub-optimum method in relation to others of its family (GCC) from a statistical point of view, copes better with reverberation

2.2.1 Signal models

Mainly, three signals models are covered in the literature, which describe quite different acoustic scenarios: the ideal free-field model, the multipath model and the reverberant model.

2.2.1.1 Free-field model

It is the most used model, which represents the ideal single-path propagation of a wave-front to an array of sensors in an anechoic open space. It considers two signals acquired at two spatially separated sensors, as attenuated and delayed versions of a source signal plagued with additive noise (Equation 2.2). Zero time-delay means that the source bearing is broadside to the sensor pair; and maximum time-delay occurs when the source bearing is endfire.

$$x_k[n] = \alpha_k s[n - t - f_k(\tau)] + w_k[n] \ , \qquad k = 0, 1, 2 \ldots K - 1 \tag{2.4}$$

It can be extended to K hydrophones, as stated in Equation 2.4, where α_k ($0 \leqslant \alpha_k \leqslant 1$) are the attenuation factors due to propagation losses, $s[n]$ is the unknown source signal, t is the propagation time from the source to the reference sensor 0, $w_k[n]$ is an additive noise signal at the k-th hydrophone, τ is the relative delay between hydrophones 0 and 1, and $f_k(\tau)$ is the relative delay between hydrophones 0 and k, with $f_0(\tau) = 0$ and $f_1(\tau) = \tau$. For $k \geqslant 2$, the function $f_k(\tau)$ depends on both; the relative time-delay and the hydrophones array geometry.

It is further assumed that s[n] is stationary and rather broadband and $w_k[n]$ is a zero-mean, Gaussian stationary random process, which is uncorrelated with the source signal as well as the noise signals at other channels.

For this model, TDE aims to determine an estimate $\hat{\tau}$ of the true time-delay τ using a set of finite observation samples, where the signal and noise remain stationary. The techniques originated from this model are therefore usually employed in slowly varying environments.

2.2.1.2 Multipath model

The free-field model, which lays the basis of the most extended time-delay estimators, does not accurately describe the complexity of real channels though. For instance, in many cases, the received waveform at a sensor consists of delayed and weighted replicas of the original signal, in addition to the direct-path signal. This is the result of multiple reflections and attenuation of the signal along the channel.

In this scenario, the measured signals at each k hydrophone can be modelled as in Equation 2.5, where α_{km} is the attenuation factor from the source to the k-th sensor via the m-th path, t is the propath, t is the propagation time from the source to sensor 0 via direct path, $s[n]$ is the unknown source signal, $w_k[n]$ is an additive noise signal at the k-th hydrophone, τ_{km} is the relative delay between sensor k and sensor 0 for path m with $\tau_{01} = 0$ and M is the number of different paths, which is usually unknown.

$$x_k[n] = \sum_{m=1}^{M} \alpha_{km} s[n - t - \tau_{km}] + w_k[n] \ , \qquad k = 0, 1, 2 \dots K-1 \tag{2.5}$$

The assumptions about $s[n]$ and $w_k[n]$ are exactly the same that those applicable to the ideal free-field model; mainly, both are mutually uncorrelated and stationary.

This model is widely used in oceanographic environments, where the multipath propagation can be reduced to the strongest first-order reflections, that come from the surface and the bottom of the sea (Figure 2-3). As well as the ideal model, this model seek to estimate the relative time-delay of each sensor via the direct-path.

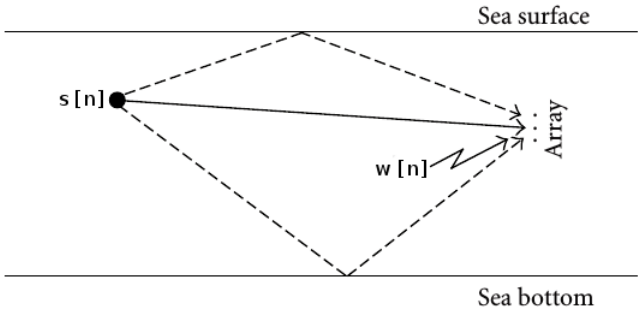


Figure 2-3: Sea reflections. Figure credit from [11]

2.2.1.3 Reverberant model

This model arises to address the shortcomings of the multipath model when the number of echoes (M) is large. This is the case of room acoustics, where the received direct-path wavefront is embedded in a substantial number of reflections of different orders.

The reverberant model represent the acoustic impulse response of the channel with an FIR filter. Thus, the received signals are expressed as in Equation 2.6, where $x_k[n]$ is related with the source signal $s[n]$ through the convolution with the channel impulse response h_k. The same signal assumptions, previously made for the ideal and multipath model, affect to this model.

$$x_k[n] = h_k * s[n] + w_k[n] \ , \qquad k = 0, 1, 2 \ldots K-1 \tag{2.6}$$

In this situation, there is no explicit reference to the time-delay τ within the mathematical model (Equation 2.6). It is, however, hidden within the channel impulse response. Therefore, the time-delay can be measured only after blindly[4] estimating the channel impulse responses.

2.2.2 The family of the GCC methods

The family of generalized cross-correlation (GCC) algorithms are so far the most used approaches to time-delay estimation. The work [19] provided a new understanding of the problem, which allowed to integrate several techniques within the GCC framework. Even nowadays, GCC methods are extended to face TDE in complex environments [10]. In particular, this extensive family encompasses all the set of algorithms composed by a cross-correlator preceded by pre-filters.

2.2.2.1 Cross-correlation (CC)

This is the earliest and most direct time-delay estimation method. It is based on the ideal free-field model stated in Equation 2.4 and employees two hydrophones ($K = 2$).

[4] since the source signal is unknown (PAM)

Herein, the true time-delay τ is estimated as the lag time that maximizes the cross-correlation function between two observation signals $x_1[n]$ and $x_2[n]$, as stated in Equation 2.7, where $R_{x_1x_2}$ represents the cross-correlation function.

$$\hat{\tau}_{12} = \underset{m}{\operatorname{argmax}}\, R_{x_1x_2}[m] \tag{2.7}$$

The cross-correlation function (CCF) is likewise defined as the expectation of the sliding inner-product of both signals along the range of possible delays (Equation 2.8). This range depends on the array geometry, which bounds the maximum time-delay τ_{max}, thence the CCF is computed for $m \in [-\tau_{max}, \tau_{max}]$.

$$R_{x_1x_2}[m] = E\{x_1[n]x_2[n+m]\} \tag{2.8}$$

Nonetheless, because of the finite observation time, the CCF is unknown and can only be estimated. In general, the CCF general, the CCF is therefore replaced by its time-average estimate (Equation 2.9).

$$\hat{R}^{CC}_{x_1x_2}[m] \triangleq \begin{cases} \frac{1}{N}\sum_{n=0}^{N-m-1} x_1[n]x_2[n+m], & m \geq 0 \\ \frac{1}{N}\sum_{n=-m}^{N-1} x_1[n]x_2[n+m], & m < 0 \end{cases} \tag{2.9}$$

2.2.2.2 Generalized cross-correlation (GCC)

The GCC algorithm [19] emerged as a revision of the cross-correlation (CC) function, so it also relies on the ideal free-field model ($K = 2$). As before, the time-delay is obtained as the lag time that maximizes the CCF, but unlike the CC method, this correlation is computed in the frequency domain with filtered observation signals. Therefore, the problem can be formulated as in Equation 2.7 and the CCF (Equation 2.8) to be estimated, is substituted by the GCC (Equation 2.10).

$$R^{GCC}_{x_1x_2}[m] \triangleq \mathcal{F}^{-1}\{\phi(f)S_{x_1x_2}(f)\} = \int_{-\infty}^{\infty} \phi(f)S_{x_1x_2}(f)e^{i2\pi fm}df \tag{2.10}$$

Equation 2.10 states the generalized cross-correlation function, where $\mathcal{F}^{-1}$ stands for the inverse discrete Fourier transform, $S_{x_1x_2}(f)$ is the cross-spectrum of $x_1[n]$

and $x_2[n]$, and $\phi(f)$ is a frequency domain weighting function. The cross-spectrum (Equation 2.11) has to be estimated, it is usually obtained by approximating the expected value by its instantaneous value (Equation 2.12).

$$S_{x_1x_2}(f) \triangleq E\{\mathbf{X}_1(f)\mathbf{X}_2^*(f)\} \tag{2.11}$$

$$\hat{S}_{x_1x_2}(f) \triangleq \mathbf{X}_1(f)\mathbf{X}_2^*(f) \tag{2.12}$$

Note that when the weighting function is constant, the GCC becomes a frequency-domain implementation of the CC. Besides, back in 1976 [19] showed the derivation of others TDE algorithms from the GCC by changing the weighting function $\phi(f)$. So far, this method has become extensively popular due to the flexibility it provides; since several weighting functions have been proposed to adopt the GCC to diverse environments. While some of them behave better at ideal conditions, as for instance the ML processor[5], others are more robust in demanding surroundings, such as the PHAT transform which deals with reverberation.

2.2.2.3 Phase Transform (PHAT)

The phase transform is a particular case of the GCC, since it discards the magnitude and only keeps the phase. It is motivated by the fact that the time-delay information is conveyed in the phase, rather than the amplitude, of the cross-spectrum. Thus, it is achieved by setting the weighting function as in Equation 2.13.

$$\phi(f) = \frac{1}{|S_{x_1x_2}(f)|} \tag{2.13}$$

The GCC for the PHAT weighting is therefore given by Equation 2.14.

$$R_{x_1x_2}^{PHAT}[\tau] \triangleq \int_{-\infty}^{\infty} e^{i2\pi f(\tau-\tau_{12})}df = \begin{cases} \infty, & \tau = \tau_{12} \\ 0, & otherwise \end{cases} \tag{2.14}$$

[5]This algorithm (ML) can achieve the CRLB under various assumptions: no reverberation, no multipath, constant time-delay, uncorrelated stationary Gaussian signal and noises and spectra of noises is known a priori

In this case, the PHAT correlation only depends on the time-delay to be estimated τ_{12} (Equation 2.14). The estimation no longer relies upon the observation signals (under free-field conditions), unlike the classical cross-correlation[6]. This makes the phase transform a more suitable TDE method for non-stationary signals, whose cross-spectrum is continuously changing.

Furthermore, by placing equal emphasis on each frequency, the PHAT weighting is sub-optimal under ideal conditions, but tends to be less susceptible to adverse conditions, especially reverberation [19].

2.2.3 Adaptive eigenvalue decomposition algorithm (AED)

The techniques heretofore presented address TDE by measuring the cross-correlation between two signals. Nevertheless, despite PHAT coping better with reverberation, the whole family of GCC methods fails when acoustic reflections become significant, since they are based on a direct-path signal model that does not represent real environments. The adaptive eigenvalue decomposition algorithm proposed in [3], approaches TDE from a different model. It assumes the reverberant model ($K = 2$), so it blindly estimates both channel impulse responses and afterwards measures the time-delay between the identified direct-paths. Actually, an accurate estimation of the whole propagation model is not straightforward, however, this algorithm only seeks to detect the direct-paths of both impulse responses for further TDE.

For the reverberant signal model introduced in Equation 2.6, the particular case of two receivers can be expressed as in Equation 2.15, by neglecting the noise terms.

$$x_1[n] * h_2 = s[n] * h_2 * h_1 = x_2[n] * h_1 \tag{2.15}$$

[6]As discussed in [23], these conclusions are about the model's mechanism. Herein, for instance, PHAT only depends on the time-delay because of the ideal free-field model. Nonetheless, when the derivation of the PHAT GCC comes from the reverberant model, it also depends on both channel impulse responses. Hence, the conclusions should be drawn with caution

This cross relation can be rewritten in matrix form as in Equation 2.16.

$$\mathbf{x}^T[n]\mathbf{u} = \mathbf{x}_1^T[n]\mathbf{h}_2 - \mathbf{x}_2^T[n]\mathbf{h}_1 = 0 \tag{2.16}$$

where,

$$\mathbf{x}[n] = \begin{bmatrix}\mathbf{x}_1^T[n] & \mathbf{x}_2^T[n]\end{bmatrix}^T, \qquad \mathbf{u} = \begin{bmatrix}\mathbf{h}_2^T & -\mathbf{h}_1^T\end{bmatrix}^T$$

and the channel impulse response vectors h_2 and h_1 of length M,

$$\mathbf{h}_i = [h_{i,0}, h_{i,1} \ldots h_{i,M-1}]^T, \qquad i = 1, 2$$

Multiplying Equation 2.16 by $\mathbf{x}[n]$ from the left-hand side and taking expectation yields Equation 2.17.

$$\mathbf{R}_{xx}\mathbf{u} = \begin{bmatrix} R_{x_1x_1} & R_{x_1x_2} \\ R_{x_2x_1} & R_{x_2x_2} \end{bmatrix} \mathbf{u} = \mathbf{0} \tag{2.17}$$

where $\mathbf{R}_{xx}$ stands for the covariance matrix of the received signals,

$$\mathbf{R}_{x_ix_j} = E\{\mathbf{x}_i[n]\mathbf{x}_j^T[n]\}, \qquad i, j = 1, 2$$

Equation 2.17 indicates that $\mathbf{u}$ is in the null space of $\mathbf{R}_{xx}$, which means that the impulse responses vector $\mathbf{u}$ is the eigenvector of the covariance matrix $\mathbf{R}_{xx}$ corresponding to the eigenvalue 0. Moreover, the covariance matrix $\mathbf{R}_{xx}$ has one and only one eigenvalue equal to 0 assuming the following conditions:

1. The two impulse responses $\mathbf{h}_1$ and $\mathbf{h}_2$ have no common zeros

2. The autocorrelation matrix $\mathbf{R}_{ss}$ of the source signal is of full rank

Nonetheless, the so far ignored independent white noise term, regularizes the matrix $\mathbf{R}_{xx}$, which tends to be positively definite and does not have a zero eigenvalue anymore. Therefore, the channel responses $\mathbf{u}$ can be identified as the normalized eigenvector of $\mathbf{R}_{xx}$ which corresponds to its smallest eigenvalue (Equation 2.18).

$$\hat{\mathbf{u}} = \underset{\mathbf{u}}{\operatorname{argmin}}\, \mathbf{u}^T \mathbf{R}_{xx} \mathbf{u}, \qquad ||\mathbf{u}||^2 = 1 \tag{2.18}$$

From Equation 2.18, the optimum filter weights $\mathbf{u}_{opt}$ of the impulse responses can be estimated adaptively by using a constrained LMS algorithm, as in Equatse responses can be estimated adaptively by using a constrained LMS algorithm, as in Equation 2.19.

$$\hat{\mathbf{u}}[n+1] = \frac{\hat{\mathbf{u}}[n] - \mu e[n]\mathbf{x}[n]}{\|\hat{\mathbf{u}}[n] - \mu e[n]\mathbf{x}[n]\|} \tag{2.19}$$

where μ is the updating step-size and the error signal e[n] is given by,

$$e[n] = \hat{\mathbf{u}}^T[n]\mathbf{x}[n]$$

It can be seen that minimizing the square value of $e[n]$ is approximately equivalent to solving the eigenvalue problem of Equation 2.18 [3].

Besides, it is crucial to provide a convenient initialization to ensure convergence, especially into the two direct-paths of the impulse responses. Considering that the first half of matrix $\mathbf{u}$ is an estimation of $\mathbf{h}_2$, [3] suggests to initialize it by 1 at a sample somewhere in the middle ($M/2$, where is able to represent positive and negative delays during the adaptation). Conversely, as the second half of $\mathbf{u}$ represents an estimate of $-\mathbf{h}_1$, a negative peak should dominate it (-1), modeling the direct-path of $-\mathbf{h}_1$.

Once the algorithm has converged, the time-delay is estimated as the time difference between the direct-paths of the impulse responses $\mathbf{h}_1$ and $\mathbf{h}_2$. It means that the time-delay is measured as the difference between the indices of the highest peaks of each estimated impulse response (Equation 2.20). This procedure is therefore assuming that the direct-path is the principal contribution to the received signal.

$$\hat{\tau}_{12}^{AED} = \underset{m}{\operatorname{argmax}} \left|\hat{\mathbf{h}}_{1,m}\right| - \underset{m}{\operatorname{argmax}} \left|\hat{\mathbf{h}}_{2,m}\right| \tag{2.20}$$

2.3 Algorithmic modeling

Data modeling approaches have provided major insights into time-delay estimation and they are as well the only deployed methods in real applications. Nevertheless, their assumptions are hard to catch in some areas of interest, including underwater Figure credit from for cetacean localization, where these general methods fail to meet the demands of such a complex framework. In particular, the observation signals do not match the assumptions of the methods described in the above section. While these time-delay estimators assume that the received signals are broadband and stationary, whistles are narrowband and non-stationary, due to their chirp-like nature. Most of these algorithms have also proved their effectiveness at rather high SNR, however, an efficient cetacean localization application should be able to confront adverse conditions, such as extremely low SNRs.

Apart from the observation signals, traditional methods barely handle the complexity of underwater propagation. On the other side, machine learning techniques can be tuned to match specific acoustic surroundings.

Given all these facts, algorithmic modeling may approach better Figure credit from for whale localization, avoiding unsuitable data models. Unfortunately, as far as an artificial database is concerned, data still relies upon a generation model. Nonetheless, the dependency between the estimator and the model is certainly weaker, since they are indirectly connected through the data. In other words, the estimator does not optimize an error derived from an assumed generation model anymore, but rather an error that comes from the predictions of data synthetized by a generation model.

So far, little has yet been published using machine learning techniques besides the sub-sample estimation of [26][27], and the sample estimation of [15][16]. These works achieve satisfactory results training artificial neural networks, specifically multilayer perceptrons. Both approaches can be characterized by the output they provide: while the former [26][27] brings a nominal continuous time-delay, the later [15][16] yields a multidimensional discrete distribution.

2.3.1 Nominal continuous estimation

The method presented in [26] was the very first supervised Figure credit from, it estimates a constant time-delay, providing sub-sample precision, through the use of artificial neural networks. The model is trained with a handmade database composed of one thousand data for training and one thousand data for test.

2.3.1.1 Data generation

Data is generated from an ideal free-model (Equation 2.4), where the time-delay varies randomly from 1 to 10 sampling intervals, the noise levels have standard deviation values that goes from 0.1 to 0.5 and the input length is as large as 256 samples.

It is further assumed that the reference signal is known, as well as its position within the segment. In particular, it is a sinusoidal of frequency w_0 rad/s starting at the first sample of the segment. This method performs therefore TOA, rather than TDOA estimation, so it can not be used as a PAM system.

The reference signal and a delayed replica of it are both filtered by a fourth-order bandpass infinite impulse response (IIR) filter. The filtered signals are locally normalized with respect to its highest values.

Finally, the training is supervised by labels that represent nominal time-delays in seconds, real values from 0.0 to 0.5 seconds (1 to 10 samples).

2.3.1.2 Architecture

The network is arranged in a regression architecture; a feed-forward network with few hidden layers and a single output neuron with linear activation, in addition, typically a MSE is optimized through a variant of gradient descendent (Figure 2-4). In this case, two hidden layers with hyperbolic tangent activations are used. The first hidden layer consists of 10 neurons and the second one is composed of 5 neurons. The model is trained with resilient backpropagation, which means that only the sign of the gradient is considered in order to update the weights. Besides, the input signals are not extra pre-processed, the filtered waveform is directly applied to the network.

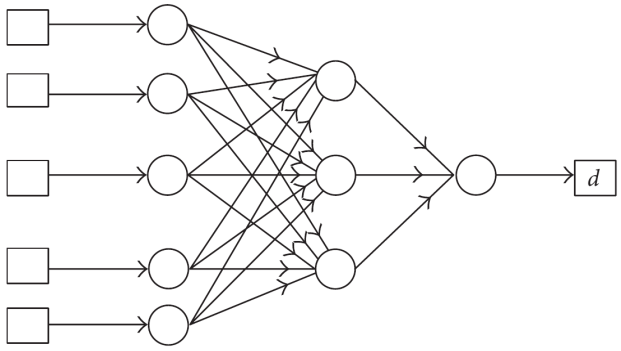


Figure 2-4: Regression architecture of [26]. Figure credit from [26]

Data is applied to the network in three different ways:

Parallel input form the reference signal $s_f[n]$ and its delayed replica $r_f[n]$, both of length $N = 256$ samples, are concatenated in $\mathbf{x} = \left[s_f[n], r_f[n]\right]$ of length $2N$.

Difference input form the difference between both signals, $\mathbf{x} = s_f[n] - r_f[n]$ of length N, is applied to the network.

Single input form only the filtered and normalized delayed signal $r_f[n]$ is used. This reveals that the reference signal does not provide further information in TOA estimation, since the network is actually measuring the absolute time-delay of a known signal.

2.3.1.3 Revision

This model was revised in [27], but minor changes were reported. The only relevant update, is the introduction of a pre-processing stage. The filtered waveform is now transformed by the Discret Cosine Transform into DCT coefficients. Afterwards, the optimal combination of hidden neurons and number of DCT coefficients is determined with a validation dataset. Consequently, only the most sensible coefficients to time-delay variations, for a specific sinusoidal, are kept. Finally, the feature compression reduces the size of the previous model [26] to one hidden layer with 15 neurons.

2.3.1.4 Results

To sum up, this supervised Figure credit from shows comparable results to those obtained by the cross-correlation technique. Moreover, the obtained error is normally distributed, where the 99.9% laid within ±0.153 sampling intervals (Figure 2-5).

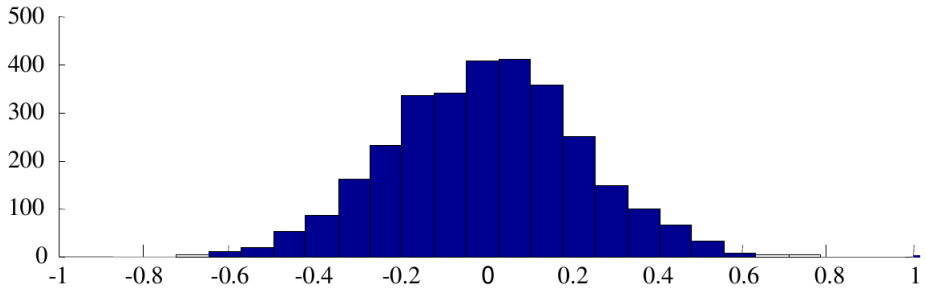


Figure 2-5: Figure credit from error histogram for noise std = 0.5. Figure credit from [27]

On the other hand, this model is extremely limited to a specific scenario, where the source signal is known and rather deterministic, both reference and replica signals can be easily represented in a low-dimensional space and the time-delay range is quite narrow (from 0 to 10 samples).

2.3.2 Multidimensional discrete estimation

The work reported in [15] estimates a multidimensional discrete time-delay, which implies sample precision, training multi-layer perceptrons. This model is trained with 8 handmade datasets, containing each one 400000 samples.

2.3.2.1 Data generation

Herein data is generated from the ideal free-model as in [26], time-delays and noise standard deviations are larger though. While the maximum time-delay of [26][27] can be addressed by beamforming techniques, in [15] its range is wider, in such a way that it falls below the spatial Nyquist range.

In this case, both reference and replica signals, as well as their positions within the processing segment, are unknown. Therefore, this system performs indeed TDOA

estimation. In addition, this database is composed of chirp signals which featured random characteristics; there are linear and quadratic chirps, its duration vary from 10 to hundreds of samples at a sampling rate of 16 kHz, and its frequency range also changes among examples.

In overall terms, seven artificial datasets with increasing white Gaussian noise and one dataset with varying noise are constructed (Table 2.1).

Dataset id	1	2	3	4	5	6	7	8
SNR	∞	5	2.5	2	1.67	1.25	1	varying

Table 2.1: Signal-to-noise ratio in linear scale of each dataset

2.3.2.2 Architecture

Multilayer perceptrons architectures including a single hidden layer and 30 hidden units are used. Sigmoid and linear activation functions are respectively used for the hidden and output units. The training procedure is conducted through a standard backpropagation algorithm, with a fixed mini-batch size of 100, during 100 epochs. Weight decay and a sparsity penalty along with $L2$ regularization are set to improve generalization over the validation dataset.

Moreover, the concatenation of both signal waveforms is directly processed by the network, as in the parallel input form of [26]. On the other side of the network, the labels represent the ideal time-delay response, a Kronecker delta function at the discrete nominal delay (Equation 2.21). It means that the labels are one-hot encoded; a maximum value of 1 at the true time-delay and zeros elsewhere. The author also proposed to use some other label functions in further researching, for instance, Gaussian windows.

$$Label(n, \tau_{12}) = \delta(n - \tau_{12}) \tag{2.21}$$

2.3.2.3 Results

This work not only focus on the nominal time-delay, but it also analyzes the predicted delay distribution, since most post-processing stages, such as tracking, can take advantage of a smooth multidimensional time-delay estimation.

Therefore, on one hand, the error between the estimated nominal time-delay and the true time-delay is evaluated in terms of bias and variance, and on the other, the similarity of the output of the neural net with the ideal target is measured with the symmetric Kullback-Leibler divergence.

Table 2.2 summarizes the evolution of the error as the SNR decrease, where the best trained model in terms of mean error is compared with the non-supervised methods described in the previous section. As noise increases, the neural network proves to perform consistently better than any other method at stake; its error remains confined, while non-supervised approaches face large variance error and strongly biased estimates (more than one order of magnitude higher).

	MLP		PHAT		XCOR		AED	
SNR	μ_e	σ_e	μ_e	σ_e	μ_e	σ_e	μ_e	σ_e
∞	0.98	1.32	0.16	3.86	0	0	43.29	32.86
5	2.13	8.89	196.38	115.70	222.51	119.86	207.95	79.63
2.5	4.86	16.55	181.09	113.27	271.72	114.09	200.99	78.34
2	6.60	19.43	171.02	110.21	292.26	106.84	197.93	78.47
1.67	8.55	21.82	161.57	106.70	304.85	102.86	195.04	78.82
1.25	13.09	26.29	147.38	99.47	313.32	102.37	191.80	79.08
1	18.74	30.23	138.22	94.24	312.02	106.29	191.41	78.87
varying	8.84	23.35	171.35	110.48	274.01	117.78	199.54	79.61

Table 2.2: Mean and standard deviation of the error of [15] as a function of the SNR

In the general case, when the SNR is variable, all the trained models ($MLPx$) outperform non-supervised methods (Figure 2-6). Besides, half of the data is roughly unbiased, and the mean error does not exceed 10 samples.

Likewise, a similarity measure, named Q_{KL}, is derived from the Kullback-Leibler divergence (Equation 2.22). It quantifies how the output of the estimator NN_{12}, diverges from the ideal target $\delta(n - \tau_{12})$.

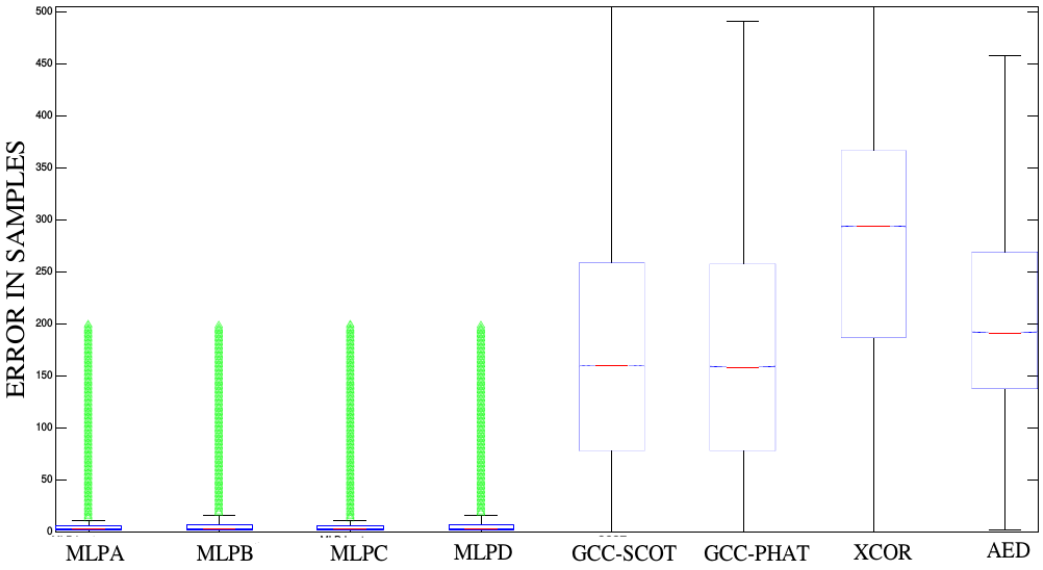


Figure 2-6: Error distribution of various estimators for variable SNR $\in [5, 1]$. Figure credit from [15]

$$Q_{KL} = KL(NN_{12}, \delta(n - \tau_{12})) - KL(\delta(n - \tau_{12}), \delta(n - \tau_{12})) \tag{2.22}$$

where KL(P, R) is the symmetric Kullback-Leibler divergence between distributions P and R,

$$KL(P, R) = -\sum_x p(x)log(r(x)) + \sum_x p(x)log(p(x))$$

This metric behaves as a distance, since it approaches 0^+ when the output of the neural network resembles its ideal target. For the noiseless and noisy case, the measure Q_{KL} is evaluated for the neural network and the cross-correlation estimator (Figure A-1). In both cases, the shape of the target distribution is much closer to the shape of the neural network, and thus Q_{KL} produces higher values for the cross-correlation estimator (Table 2.3).

	$Q_{LK}(Target)$	$Q_{LK}(MLP)$	$Q_{LK}(XCOR)$
$SNR = \infty$	0	0.0022	13.82
$SNR \in [5, 1]$	0	0.1895	14.30

Table 2.3: Q_{KL} pseudo-distance from the neural network output (MLP) and cross-correlation output (XCOR) to the target $\delta(n - \tau_{12})$

Chapter 3

Database

This chapter covers the foundations of this work. From the motivation of building a custom database to its design and implementation. Some examples of the time-delay segments generated for this database are showed in Figure A-6 and Figure A-5.

3.1 Motivation

Nowadays there is no standardized database of underwater acoustics, neither for localization nor classification of marine mammals species. Given the large number of species and their very different vocalizations, a general database for classification or localization is a challenge.

In the literature, authors who develop analytical approaches work with small real datasets, while those who require a training stage synthesize their own artificial sets [27][26][22].

On the other hand, despite the database synthesized in [15] being quite useful to validate a supervised approach for time-delay estimation, it is not intended to work with real data. There is still a gap between these data and real recordings, because of its artificial nature. For our concerns, it is worth to build the database from real signals to get closer to the real implementation.

3.2 Objectives

The main goal is to build a database to approach time-delay estimation for cetaceans whistles. Furthermore, we consider several secondary objectives:

- To find out the factors that have the most influence on time-delay estimation for these particular acoustic signals. We should build the database accordingly, devoting major efforts to the crucial information.

- Motivated by [15], where the introduction of a time-delay distribution as a target was a really big step forward, we attempt to develop a suitable label probability distribution to train a supervised model.

- To build a scalable database, by using the most established input formatted data. In such a way, that the present database could be extended with more labeled whistles through the current generation software. Furthermore, despite there being no standard, the tendency moves towards the format implemented in [30] and [12].

- To study the introduction of more channels for time-delay estimation, inspired by previous work [4][10] which shows relevant advantages.

3.3 Design

The design of this database is the most important building block of this work, considering that the time-delay estimator relies on the statistical structure of the data. It should reflect the main features of real cetaceans vocalizations.

We started this database with the DCL 2011 workshop dataset [12], which is also available in *Tethys*. It is likewise a collection of recordings described in detail in [28] and [2]. Briefly, it consist of whistles and echolocation clicks from five species gathered from ships and platforms in the Southern California Bight and around Palmyra Atoll: bottlenose dolphins (Tursiops truncatus), melon-headed whales (Peponcephela

electra), short- and long-beaked common dolphins (Delphinus delphis and D. capensis), and Grays spinner dolphins (Stenella longirostris longirostris). In all cases, single species schools were visually verified and no other species were sighted during each encounter. Nonetheless there are multiple, and sometimes overlapped sources, corresponding to multiple individuals of the same school. Measurements were taken using either towed or dipped hydrophones as described in the cited papers. Hydrophone response was flat across most of the whistle bandwidth, although the custom preamplifiers which were designed to whiten ambient ocean noise varied. Data were sampled at 192 kHz with 16 or 24 bits of quantization.

A subset of the DCL 2011 was examined to provide a ground-truth for the assessment of the workshop [12]. These labels define how frequency evolves with time[1]. In general, these signals are whistles, so they look like complex pseudo-chirps, which may have a very rich harmonic like structure [2]. They are a great challenge, due to their quick frequency sweeping, where the well-known uncertainty principle of time-frequency representations (STFT, Wavelet...) can be observed.

These clean labels are indeed a good starting point for developing our time-delay database, mainly due to two reasons:

- To avoid having to perform preprocessing tasks which are out of the scope of this work, for instance, automated whistle extraction, blind source separation and classification.

- To take control over the different factors that affect real recordings. Once we have the clean annotations, we can integrate each component in the signal independently (SNR, number of channels...). As a result, we get a further insight into the problem.

For this purpose multiple sub-datasets were created with a complexity gradually increasing towards resembling real environments.

[1] This time-frequency function is usually referred by the word contour

[2] Actually, when it is about cetacean vocalization, harmonics are related with overtones [32]

3.3.1 Simulation scenario

An array of omnidirectional hydrophones simulates the time-delay response for an acoustic source placed in the coverage area. Although the specific array disposition does not make a difference for time-delay estimation, it asserts time-delay approaches over other methods under particular conditions.

In the present study, we simulate a widely used array arrangement. It was proposed in [1] for dealing with dolphin echolocation clicks[3]. This setup is motivated by the fact that, with four hydrophones arranged in a configuration other than a line, it is possible to analytically determine the exact position of the sound source to one of two points [29]. Likewise, more channels provide a better elimination of disturbances of different backgrounds. Especially if they are located far from each other, since this background noise would usually become more and more incoherent, which is in general easier to disregard.

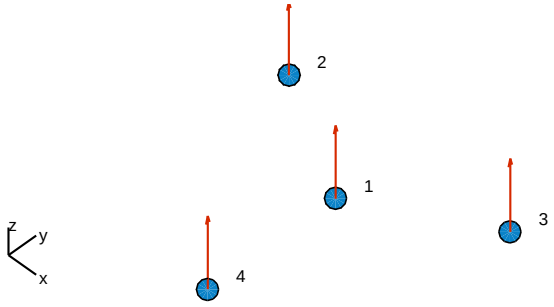


Figure 3-1: Array arrangement

The four sensors array is arranged in a symmetrical star configuration, with one center hydrophone and three extending arms spaced *120°* apart (Figure 3-1). Where *hydrophone 1* is the master; the time-delay reference point.

Echolocation clicks are very directional and can not be recorded at very sparse hydrophones simultaneously. Therefore, this array arrangement usually has a shorter array size. However, for our purposes, it is worth to place the sensors far away from the master in order to wider the coverage area.

[3]Despite [1] being about clicks, rather than tonal sounds, the array arrangement has nothing to to with that

It has been defined a space between master and slave hydrophones of *1 km*. In other words, hydrophones 2, 3 and 4 lie on a circumference of radius *1 km* from the master. This configuration constraints the maximum time-delay, from any slave hydrophone to the master, to *625ms*.

3.3.1.1 Array processing outlook

Besides, from the array processing theory, a wave source is considered to come from far-field if it satisfies Equation 3.1, where D refers to the sensor spacing.

$$|r| > \frac{2D^2}{\lambda} = \frac{2D^2 f_{max}}{\nu} \tag{3.1}$$

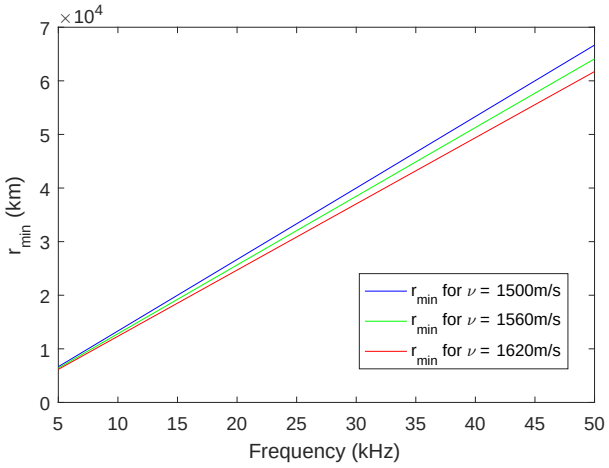


Figure 3-2: Fraunhofer distance versus frequency

Figure 3-2 shows the Fraunhofer distance for the frequency range of interest, which defines the limit between the near and far field. It reveals the infeasibility of a far-field assumption, seeing that a source can not be sense at such a large distance. Therefore, a plane wave model does not apply to this scenario.

Furthermore, the spatial aliasing requirement, stated in Equation 3.2, must be adhered to exploit the variety of array processing methods. In our scenario the maximum non-aliased frequency is *0.8* Hz, which overtakes our minimum frequency.

$$D < \frac{\lambda_{min}}{2} \tag{3.2}$$

3.3.2 Segment assembling

Unlike generating a whole large waveform full of whistles, we built small signals segments[4]. This method provides us with the best balance of realism, control and flexibility. The segments simulate the output frames of a sliding window segmentation.

In this way, we can better handle the signal generation. As segments are generated contour by contour, we can ensure at each generation step a proper distribution of all the control factors; such as time-delay, relative position of the contour inside the segment, noise properties...

3.3.2.1 Segment length

The segment length should be fixed to a certain value, taking into consideration the trade-off between contour length and computing resources. The larger portion of contour the model processes, the better the estimation; however, larger segments are also computationally expensive.

This length has been fixed at 2 seconds, based on a statistical analysis of the contours length of [12]. Furthermore, the whistle duration proposed in [32] is considered as a guidance.

While in [32] the bottlenose whistle length is claimed to be around 1 second, our analysis of the whistle length along the different species of [12], shows that *75%* of the segments are below 1 second (Figure 3-3). In fact, the bottlenose whistle length appears to be a little bit larger than the common and melonheaded dolphin one, yet it is not quite significant. Moreover, neither of the outliers correspond to a very large whistle, but to a time overlap of shorter signals that compose a rather wide segment[5].

Therefore, taking into account a maximum time-delay of *625ms*, computed in Section 3.3.1, and a whistle length as large as *1s*, a segment length of *2s* offers a good balance between time-delay estimation reliability and resources consumption.

On the other hand, parts of larger contours -and especially of its delayed versions- could leave the segment scope, but the time-delay estimation should still be feasible

[4]Henceforth the reserved word segment, refers to the input processing unit of our model
[5]It is actually a result of the segmentation algorithm of 3.4.2.1

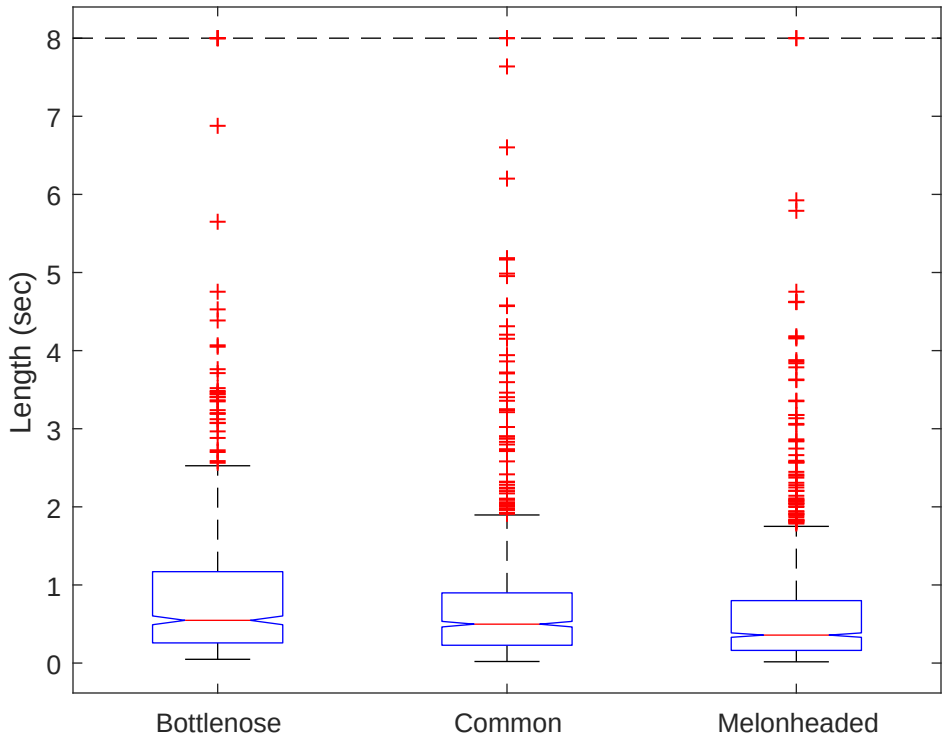


Figure 3-3: Segment lengths distribution along species

using the remaining signals.

3.3.2.2 Segment structure

Once the segment length has been worked out, the format of the segment can be defined. In general, a segment unit is composed by 4 building blocks:

Offset a left zero padding referenced to the master channel, which ideally goes from zero to segment length (in samples). Its objective is to avoid overfitting when the model detects -either implicitly or explicitly- a contour within a segment. Conversely, it simulates the fact that a contour may appear at any position in the segment.

Delay another zero padding which depends on the computed time-delay for a certain pair of sensors. It goes from minus maximum time-delay to maximum time-delay in samples. For the master reference hydrophone the delay is always zero.

Contour the whistle itself. Despite its length being variable, most of the time it is between zero and 1 second (Figure 3-3).

Padding the right zero padding required to fulfill the segment.

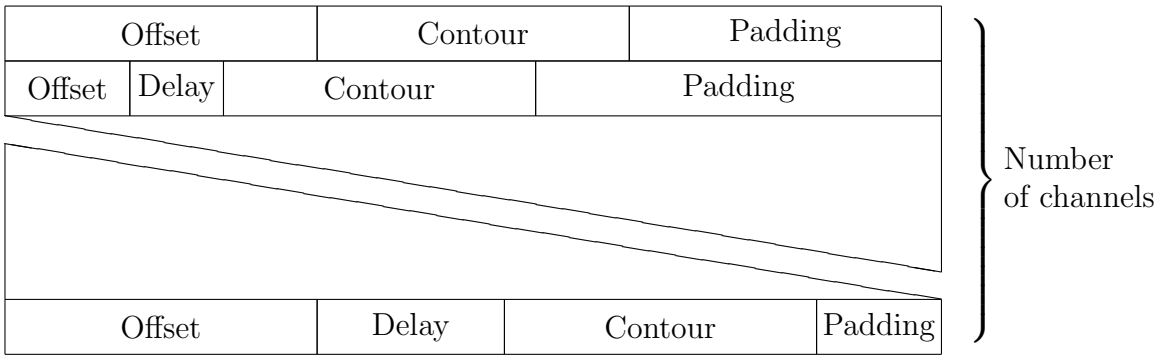


Figure 3-4: Segment structure

Figure 3-4 shows the segment format, where negative delays are also allowed. It means that the offset is reduced when there is a negative delay. In that case, the contour may even exit the segment from the left part. On the other hand, a too positive delay may cause the equivalent effect. These circumstances are handled as limit cases and might be avoided on demand. Since unequivocally, an uncompleted contour cannot be as highly correlated with its original version as when it appears unchanged.

3.3.3 Noise embedding

We have extracted real noise from non-labeled signal segments of the original recordings, since we aim to keep close to the actual implementation. In addition, this noise has been manually cleaned from hidden whistles, which are not labeled.

Moreover, not only the quantity of noise may perturb the time-delay estimation, but also its quality. Therefore, we focus on the coherence of the noise when building the segments, but always from a realistic point of view.

Simulating the conditions of sparse hydrophones, different segments of noise are randomly sampled for each channel, looking for high incoherence between them. Actually, we found that coherence of the noise changes between segments, since we

sample from a large signal which presents different characteristics (sometimes it is broadband, its spectrum has diverse patterns, random time impulsive components...). Furthermore, as noise is rather non-stationary and shows a wide variety of patterns, the coherence between channels is usually quite low.

3.3.4 Labeling

So far, authors who develop supervised approaches, have labeled its target time-delays in several ways. While [26][27] learn a continuous function with nominal delay labels, [9] discretize these delays in classes[6], and [15] build a target distribution.

In general the real time-delay is discretized into classes through a time-delay resolution. For instance, in [9] for a uniform linear array (ULA) the DOA space is divided in 91 classes. On the other hand, one of the major novelties of [15][7], was to encode the target delays into distributions, instead of one-hot vectors classes [9]. Soft labels which assist the learning procedure by allowing a controlled uncertainty around the predicted time-delay.

Thence, building on the progress of [15], a time-delay distribution label is proposed. It outperformed previous experiments with the data of [15] where the ideal target $\delta(n - \tau_{12})$ was considered, and also provided an advantageous smooth multi-dimensional output for further post-processing.

In fact, in this work, both approaches are combined. Firstly, a discretization of the time-delay is mandatory, in order to limit the number of neurons in the output layer when coping with ANN and distribution labels. Secondly, a distribution around the discretized time-delay is built.

3.3.4.1 Time-delay discretization

Essentially, we should discretize either the time-delay range or the distribution label to achieve an efficient learning of the time-delay response. Thence, we propose several

[6]Despite [9] dealing with direction of arrival (DOA) rather than time-delay, the same procedure applies to TDE

[7]Label distributions are actually covered in further research, there is no explicit reference to soft distribution labels in [15]

resolutions (Table 3.1), for the scenario defined in Section 3.3.1, which relates the angle resolution with the time-delay.

Resolution (samples)	Resolution (ms)	Resolution (degrees)	Size of output layer
1	0.0052	0.0005	240001
1047	5	0.5	251
2094	10.9	1	126
5235	25	2.5	51

Table 3.1: Time-delay resolution

Firstly, we started with the lowest resolution (25 ms). Afterwards, when the model achieved good performance, the resolution was progressively increased to the maximum of 5 ms. The results and conclusions are therefore extracted at the maximum resolution of 5 ms.

3.3.4.2 Distribution label

Before building this database, we did several experiments with data from [15]. A two hidden-layer multilayer perceptron architecture with sigmoid activations was used. The training procedure was conducted using Adam backpropagation, optimizing a MSE. Likewise, we set a dropout of 0.4 and we split the data in sixty percent for training, twenty percent for validation and twenty percent for test.

Thus, we relate the estimation accuracy with the distribution label. After different experiments, the normal distribution with a MSE was the most suitable choice.

Moreover, Table 3.2 shows the effect of widening that distribution; by increasing the allowed uncertainty around the true time-delay in the labels, the global mean error and its variance decrease in the predictions. This characterization of the error comes from an overall statistical analysis of all the independent errors, which are computed as the absolute difference in samples between the maximum of the predicted function and the ground-truth. It has nothing to do with the minimized MSE through backpropagation, which refers to the error in the predicted label function.

It is consistent with the bias-variance decomposition of the error. As the estima-

Distribution labels σ	Predictions mean error	Predictions median error	Predictions std error
1	17	6	27.7
2	15.9	6	26.5
3	15.1	7	24.7
4	14.7	7	23.2

Table 3.2: Distribution of the error in samples vs distribution label std

tion, the distribution label, allows more variance, it becomes more and more unbiased. Besides, larger variance in the distribution label may damage our ability to resolve multiple sources, but it is out of the scope of this work.

Despite this study coming from quite different data, it is a pretty good reference for developing the labels of the first release of our database. Summing up, we use as a target a Gaussian of standard deviation one around the discretized time-delay[8].

3.3.5 Datasets generation

We have generated multiple sub-datasets with diverse features, fitting our estimator to various environments of interest. The goal is to study the generalization capabilities of our model when the conditions becomes more demanding, in search of the most robust architecture.

Dataset	**Limit cases**	**Channels**	**Species**	**SNR (dB)**
1	✗	2	1	0
2	✗	2	1	-6
3	✗	2	1	-9
4	✗	2	1	-12
5	✗	2	1,2,3	mix

Table 3.3: Datasets' features

Table 3.3 shows the characteristics of each dataset, where limit cases refers to the circumstances presented in section 3.3.2.2, channels means number of hydrophones, species may include from only the bottlenoise dolphin to all the available classes;

[8]These units are on a difference scale than Table 3.2, where the time-delay has sample resolution

melon-headed whales and common dolphins, and SNR specify the quantity of noise.

3.4 Implementation

The following section briefly details how a new time-delay database is constructed from the classification datasets of [12], based on the previous design.

Firstly, we introduce the general approach for reassembling pseudo-real signals from the annotations of [12], labeled time-frequency contours (Figure A-4). Afterwards, the data generation procedure simulates the time-delay response in a particular scenario, the signals are embedded in noise and the labels are built.

3.4.1 Spectrogram reconstruction

The main idea is rather straightforward, to paint the time-frequency annotations on a canvas; the spectrogram. However, the major part of them are marked at an unfeasible time-frequency resolution. In other words, the window length does not match such a high frequency resolution. Actually, the labeler had used his intuition, smoothing and interpolating the time-frequency contours in a convenient way. Process which is thoroughly described in the metadata of [12].

After gathering all the quantized contours at a reasonable time-frequency resolution, a clean version of the original spectrogram is resembled. Nonetheless, although these annotations completely describes the instantaneous frequency magnitude along time, some signal information is lost during the labeling. These information gaps are approached in accordance with its implications on time-delay estimation.

3.4.1.1 Assumptions

We have made two major assumptions, due to the lack of information in the annotations of the DCL 2011 workshop dataset:

1. All the overlapped contours within an annotation segment arise from the same source, considering a source as either one or several mammal cetaceans emitting

sound waves from roughly the same point. This has not got significant repercussions on time-delay estimation, since the source is seen by the sensors as a random combination of real acoustic sources which share the same time-delay, the same relative localization to the sensors.

2. The amplitude is assumed to be constant. Firstly, because there is no amplitude information in the annotations. Yet also it is not even worth to estimate it from the recordings; seeing that these recordings are quite noisy, a sophisticated algorithm would be required to get a reliable estimator. Actually, we tested this approach with unsatisfactory results. The time evolution of the estimated amplitude was not consistent, in fact, it did not measure anything but noise. Likewise, we do not expect, that neither the amplitude itself nor its time evolution, could provide too much information for our purposes. Commonly, it is disregarded in applications that deal with cetacean vocalizations, such as specie classification or whistle contour extraction.

3.4.1.2 General approach

Firstly, the canvas where the contours are printed is defined. Formally speaking, we reserve memory for a large magnitude spectrogram matrix, due to memory concerns, it is define as a boolean -since the amplitude is assumed constant- sparse matrix. In this way, we use the memory resources efficiently, storing only the nonzero elements and their row indices.

For each annotations contour, the time-frequency pairs are quantized in order to match the desired resolution of the spectrogram. Lastly, its respective frequency bins are marked as true along the required frames.

3.4.1.3 Standardization

Despite being a direct reconstruction, a standardization of the data is required, since the data of [12] is an heterogeneous compilation of different recordings -at different places, labeled by various teams...- its annotations are different too; while some of

them have a huge time resolution, others are sampled at a lower one.

Figure A-2a shows the great variety of time resolutions used for labeling a certain dataset. There is a strong difference among datasets, while some of them present very high resolution (Figure A-2a), others are low quality (Figure A-2b). Besides, the sampling frequency may change over different contours, but also along the contour.

Thence, we should resample all the contours at a uniform rate without introducing aliasing. Either decimation or interpolation, accordingly as the sampling rate decreases or increases, is required. For the later case, we use cubic spline interpolation, which properly fit the smooth contours of our complex pseudo-chirps.

However, before applying the resampling filter, we must take care of the endpoint effects. These artifacts arise because the filter assumes that the signal is zero outside the borders of the signal. This is avoided through padding the contour symmetrically, just by repeating its extreme values, and afterwards subtracting from the signal, linear or higher order trends. Finally, once the signal has been resampled, the trend is recovered and the padding is removed from it (Figure A-3).

Furthermore, we should detect the intra-silences within a contour before attempting to interpolate it. Otherwise, these silence holes are filled with signal. It is important highlight, that these signal drop-outs are very characteristically features of cetaceans whistle vocalizations, which are worth to remain unaffected.

Fortunately, this silence information is implicit in the annotations; whenever there is a too large space -in relative terms- between two consecutive samples in the time array of a contour, it is likely to be silence. Therefore, a silence detector can be implement as follows:

1. Compute the approximated derivative of the time vector; by calculating n order differences between adjacent elements. In our case, order 10 works.

2. Find the peaks of the derivative, constraining the searching to a minimum peak distance of 8 samples.

Once we have retrieved all the silence intervals of a contour, they can be restored from the interpolated whistle.

3.4.2 Contour reconstruction

We aim to reassemble each independent whistle waveform from the reconstructed magnitude spectrogram. So firstly, we segment each contour from the whole spectrogram, and afterwards each waveform is estimated from its spectrogram magnitude information and its noisy audio signal.

3.4.2.1 Spectrogram segmentation

The whole spectrogram is fragmented in smaller units, called contours, due to design reasons (section 3.3.2). This procedure also provides an efficient waveform reconstruction, since despite processing a large spectrogram, only segments which contains signal are reassembled. Likewise, no blind source separation is carried out, which seems to be fuzzy and unclear even manually[9].

As far as, this is a segmentation of clean signals, we just need to apart whistles one from the other, through the gap between them. Therefore, the spectrogram is processed frame by frame, in such a way that when there is no signal during a minimum number of frames, between two consecutive contours, they are considered as independent vocalizations.

3.4.2.2 Waveform reconstruction

At this point, we have already reconstructed and fragmented the magnitude spectrogram. Nonetheless, there is no phase information at all, which would allow an exact waveform reconstruction, yet an estimated reassemble of the original time domain signal is still feasible.

This is an extended audio processing practice, normally known as short-time synthesis. Nevertheless, some of the most common methods require also the phase information, such as filter bank summation (FBS) and overlap-add (OLA). For our concerns, a least-squares approximation does the trick.

This approach seeks to estimate a sequence $x_e[n]$ whose STFT magnitude $|X_e(n, w)|$

[9]For the discussion of the motivations and implications, see the first assumption of section 3.4.1.1

is closest (in a least-squared-error sense) to the known spectrogram magnitude $|X(n, w)|$. The iteration takes place as follows:

1. An arbitrary sequence (usually white noise) is selected as the first estimate of the synthesized waveform $x_e^1[n]$

2. The STFT of the initial estimate $x_e^1[n]$ is computed, and its magnitude replaced by the target magnitude $|X(n, w)|$ (Equation 3.3).

$$X^1(m, w) = |X(n, w)| \frac{X_e^i(m, w)}{|X_e^i(m, w)|} \tag{3.3}$$

3. Finally, the waveform is re-estimated taking into account the sliding window effect. Therefore, ensuring that the sum of all the analysis windows add up to a constant (Equation 3.4).

$$x_e^i[n] = \frac{\sum_{m=-\infty}^{\infty} w[m-n] g_m^{i-1}[n]}{\sum_{m=-\infty}^{\infty} w^2[m-n]} \tag{3.4}$$

   where, $g_m^{i-1}[n]$: is the inverse DFT of $X^{i-1}(m, w)$

4. The process continues iteratively until a stopping criterion is met. The simplest one is to limit the number of iterations.

It can be shown that this process reduces the distance between the estimated magnitude spectrogram and the target one at each iteration. Thus, this approach converges to a local minimum, though not necessarily a global minimum.

Besides, this algorithm takes advantage of all the available information, since instead of white noise, real noisy recordings play the role of a first reliable estimation. An initial guess which, despite being considerably noisy, contains some worthwhile phase knowledge.

At the end of the process, the reconstructed waveforms look absolutely like real noiseless cetaceans vocalizations. However, whereas a constant contour amplitude has been assumed (section 3.4.1.1), the target magnitude spectrogram behaviours rather

like a binary mask during the iterative process. It therefore causes abrupt changes in the waveform, which are acoustically perceived as audio clicks. Phenomenon of minor significance, seeing that these signals are afterwards embedding in noise.

3.4.3 Time-delay data generation

We generate several multichannel segments from each reconstructed contour. Particularly, the user defines how many delays per contour and offsets per delay are generated. Thence, for each contour, segments are built as follows (Figure A-7):

1. If limit cases are not allowed, firstly we check the length of the contour, when it is larger than the maximum expected, one second[10], the contour is skipped.

2. We reserve memory for output_buffer, a data structure that contains all the serializable information, both multichannel signals and labels.

3. Then, time-delays are generated according to an ideal free-model. We have implemented two approaches; it can be randomly sampled, either directly from a uniform distribution, or from a localization (azimuth and elevation). For our purposes, we prefer well distributed time-delay data, rather than sampling from random directions[11].

4. For each time-delay various offsets are sampled from a uniform distribution, generating several multichannel segments with the same time-delay.

5. We randomly sample one segment of noise for each channel, as in Section 3.3.3.

6. Once we have the position of the contour in the master channel (offset), the required delays for the rest of channels and all the noise signals, we can build the segments.

7. We label the segments with distributions around the generated time-delays.

[10]Since segments are designed for contours as large as one second (Section 3.3.2.1)
[11]This second approach is direction of arrival (DOA) oriented

8. After doing all the data generation computations with double precision, we cast output_buffer to single precision. Finally, we serialize output_buffer into disk, releasing physical memory.

3.4.3.1 Segment assembling

When building multichannel segments we should handle four different cases:

Positive delay When the delay is positive, the segment is formed by concatenating a zero padding of offset length, another zero padding of delay length, the contour, and a right padding, if it is required to fulfill the segment.

Very positive delay The segment is built as before, and if the length of the segment is greater than the maximum one, the segment is cropped. In this case, the contour appears incomplete, leaving the segment scope by the right side.

Negative delay When the delay is negative and lower than the offset, the segment is built by concatenating a zero padding of offset plus delay length, the contour, and a right padding.

Very negative delay When the delay is negative and greater than the offset, the segment is composed by the most right part of the contour and a right padding. Therefore, the contour leaves the segment scope by the left side.

Afterwards, the segment is embedded in noise and it is normalized between minus one and one, in order to match the *wav* audio format.

3.4.3.2 Noise embedding

Segments are embedded in noise through the SNR introduced by the user. Thence, we should first estimate the power of noise that we aim to add.

Therefore, we compute the power of contour, then we measure the power of the noise only in the contour scope, looking for a reliable SNR estimator within the whistle

area. Lastly, we figure out the gain of the noise (Equation 3.6), adding both signals in the right proportions (Equation 3.5).

$$x[n] = s[n] + h * w[n] \tag{3.5}$$

$$h = \frac{RMS\left(c[m]\right)}{10^{SNR/20} RMS\left(w[m]\right)} \tag{3.6}$$

where, $x[n]$: is the noisy segment waveform
$s[n]$: is the noiseless segment
$w[n]$: is the noise segment
$c[m]$: is the contour, the whistle waveform
$w[m]$: is the noise segment along the duration of the whistle
SNR : is the desired signal-to-noise-ratio in dB
h : is the gain of the noise

3.4.4 Labeling

For label the segments, the time-delay range is mapped into the output layer range[12], from one to size of output layer. Later on, a normal distribution is build around the true discretized time-delay (Equation 3.7).

$$Label \sim \mathcal{N}(\tau_{12}, 1.0) \tag{3.7}$$

[12]The output neurons can be understood as quantization levels of the time-delay (Section 3.3.4.1)

Chapter 4

Time-delay estimation

During the development of the time delay estimator, several models were trained for a wide variety of low-level features[1] and, despite that large effort, our attempts were unsuccessful. In fact, our best performing estimator, thoroughly described in Section 4.2, is fed from high-level features. In the interests of clarity and concision, only the most ambitious model which deals with low-level features is briefly stated.

4.1 Earliest attempt

Following the trend towards building end-to-end learning systems and after failing to learn directly from the waveform, we designed a model that works over the spectrogram of both hydrophone channels.

This model is fed with two channel spectrograms images, magnitude and phase, for both the reference and delayed signal. The neural network is arranged in a CNN siamese architecture[2]. Thus, two identical subnetworks, sharing the same parameters and weights, as well as backpropagation, which is also mirrored across both subnets.

Each subnetwork aims to produce a low dimensional representation of its input, feature vectors with the same semantics, making them easier to compare in upper layers, where the time-delay is to be estimated. Besides, sharing weights across the

[1] These include, among others, the waveform, the spectrum, the unwrapped phase, both magnitude and phase spectrograms, the cepstrum, MFCC coefficients...

[2] Architecture proposed in [6] to constrain the network to extract similar features

two subnetworks means also fewer parameters to train for, which in turn means less data required and less tendency to overfit.

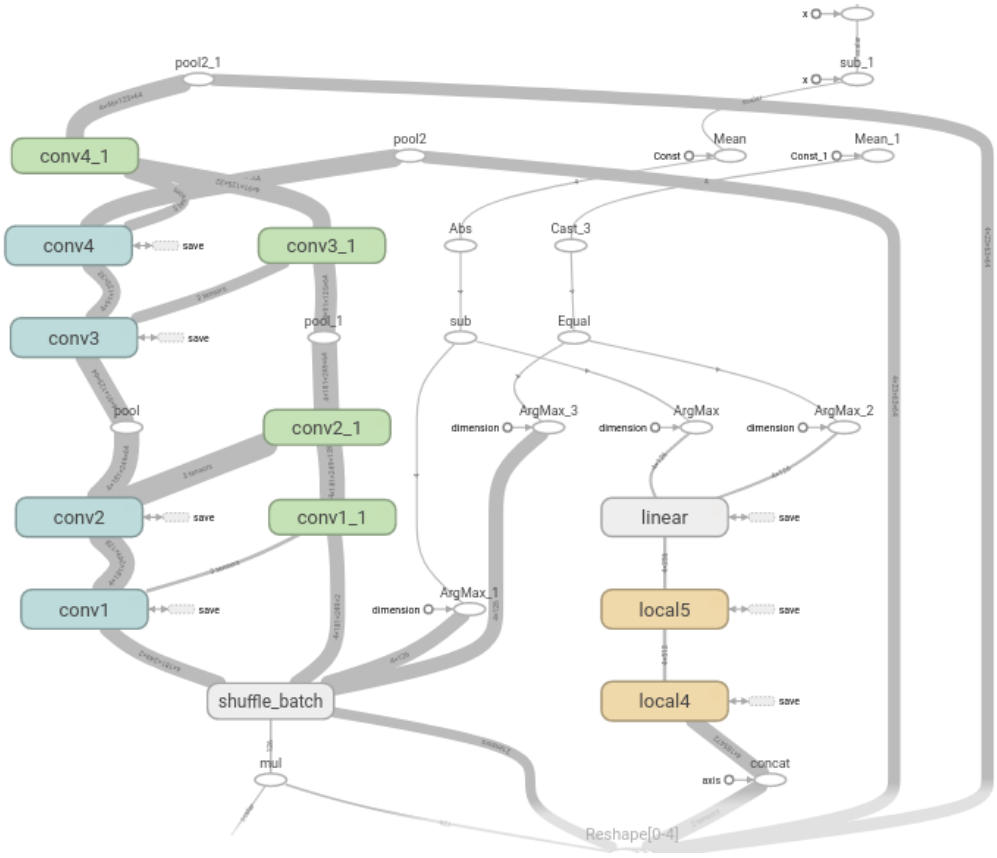


Figure 4-1: CNN siamese network graph

Figure 4-1 shows the baseline graph[3] of the trained network, where the cross-connections between siamese convolutional layers represent the shared tensors (filter weights). Several variations of this architecture were also considered, including stacked LSTM networks within upper layers of the siamese subnetwork.

This model aims to project the data into a low-dimensional space via four convolutional layers, where the information of each hydrophone flows in parallel through the siamese subnetworks, while the time-delay is estimated within the two upper fully connected layers.

Nevertheless, after several variations in the architecture, different initializations, regularizations, backpropagation methods and hours of GPU training, the model was

[3]The framework used for the implementation, TensorFlow, uses a dataflow graph to represent the computation in terms of the dependencies between individual operations

unable to learn from the data. Therefore, we found that we can not efficiently learn from such a high dimensional feature space.

4.1.1 Curse of dimensionality

Despite many efforts being made, to reduce the number of learned parameters, we failed to learn from the data we had. At this point, there is no other option, but to revise the data features.

Therefore, an in-depth examination of the feature space, from the signal characteristics to the time-delay information, can provide us valuable insights:

- The time-frequency features of the signals are very sparse. The duration of a whistle may vary from 2ms to 2s (Figure 3-3), and its indispensable frequency range goes from 5 kHz to 50 kHz. Besides, the database encompass variations from narrowband whistles ($BW \approx 500Hz$) to wideband ones ($BW \approx 20kHz$). It means that each hydrophone spectrogram image reach up to dimensions as large as [2000 time steps, 300 frequencies, 2 channels]. Likewise, a waveform based approach should use, at least, 2 seconds of segment sampled at 96 kHz to catch the whole frequency range of the signals.

- On one hand, the model seek to estimate the delay between two segments of length 2s. On the other hand, we know beforehand that the delay is constrained by the physical model to 0.625s (Section 3.3.1). Thence, the data is actually lying in a lower dimensionality manifold. Moreover, all time-delay approaches significantly improve its performance limiting the maximum lag.

This data dimensionality leads us to a double handicap, as the feature dimensionality increase, the amount of data required to train efficiently the system increase too. However, although more data is required, high dimensional features demand more resources, so less data can be used at once.

4.2 Final design

Motivated by the curse of dimensionality and under the premise that the time-delay information lies on a lower dimensional manifold, this model is based on high-level features, which are specialized in time-delay estimation.

4.2.1 Feature extraction

We have designed very specific features to approach time-delay estimation. These features have proved to be robust against real noise and well-suited to the sparse signal properties of whistles. Besides, they include the constrains of the physical model, limiting the dimensionality of the features by the maximum time-delay.

Firstly, the magnitude spectrogram of both hydrophone channels is extracted, with a time resolution at least equal to the labels' one (Section 3.3.4.1). Secondly, the energy of each frame is locally normalized. Afterwards, one hydrophone channel spectrogram is padded on both sizes with the corresponding number of frames to the maximum time-delay. Moreover, instead of zeros, the padding frames are filled with the mean of each frequency bin along time. Finally both hydrophone channel spectrograms are locally cross-correlated, sub-band by sub-band.

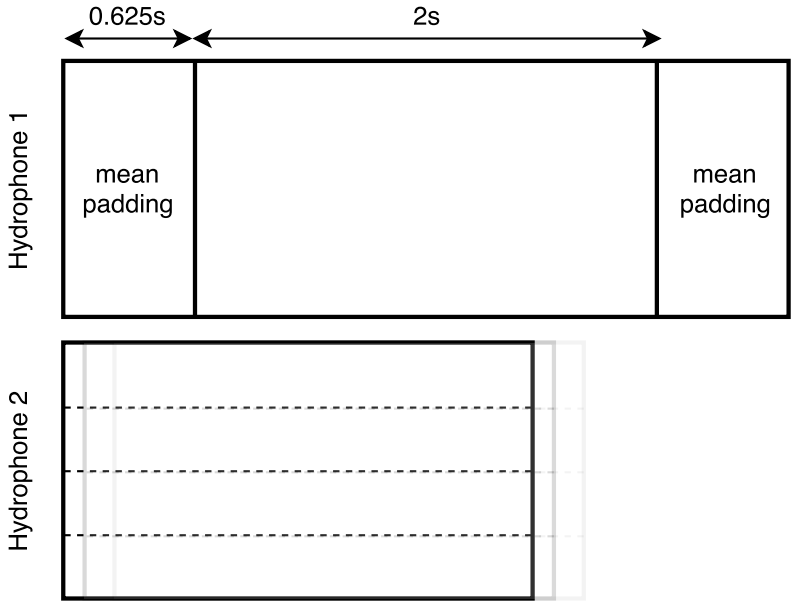


Figure 4-2: Local spectrogram cross-correlation

The last step aims to highlight the time-frequency coherence between both spectrograms. The cross-correlation is therefore computed as the sliding inner product of the spectrograms along time, but only over bins within a sub-band (Figure 4-2). Thus, we extract several coherence functions of overlapped frequency sub-bands, building an output image of partial cross-correlations functions. Each pixel r_{ij} of this image is computed as in Equation 4.1, where S_1 and S_2 are the spectrograms of the input signals, M is the number of frames, f_{min} and f_{max} define the frequency range of the sub-band j and $i \in$ [min_lag, max_lag] . Likewise, these cross-correlations are always non-negative as the spectrograms, S_1 and S_2, also are.

$$r_{ij} = \sum_{m=0}^{M-1} \sum_{n=f_{min}(j)}^{f_{max}(j)} S_1(m+i,n) S_2(m,n) \tag{4.1}$$

These features reveal a pattern, named coherence-print, where the correlation is locally maximized (Figure 4-3). Therein, the time-delay can be determined by the position of the coherence-print. Likewise, the shape of this print rather depends on the cetaceans' signals; as they carry more redundancy, the print becomes clearer. Accordingly, broader signals in time and frequency produce better features, as the coherence of the signal prevails over the coherence of the noise.

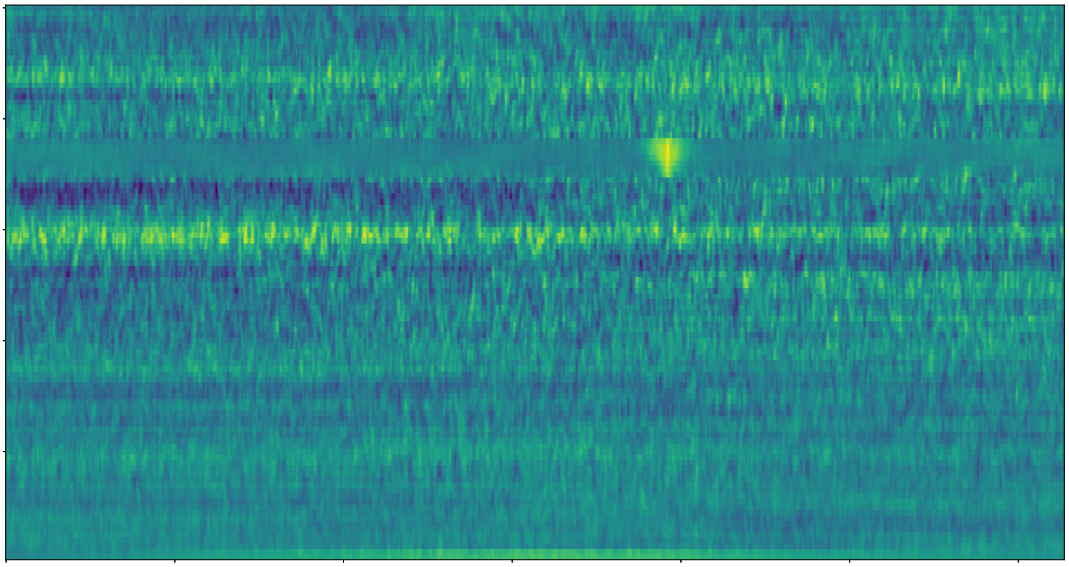


Figure 4-3: Local spectrogram cross-correlation (coherent-print features) of a narrow band whistle under favourable conditions (SNR = 0 dB)

Furthermore, we have implemented the whole procedure within the TensorFlow framework, providing an efficient processing with GPU-compatible through native APIs methods, such as usual convolutions[4].

4.2.2 Architecture

Even though many configurations have been evaluated, including several classes of recurrent networks, the fact is that none of them has improved the performance of an AlexNet-like convolutional network [20], except a fully convolutional network, where the fully-connected layers are replaced by convolutionals. We have compare both architectures, an AlexNet-like network, comprising convolutionals and fully connected layers, and a fully convolutional network.

4.2.3 AlexNet-like network

The input to our convolutional network is a fixed-size 1255 x 100 one channel image, which corresponds to coherence-print features sampled at 1ms. Even though lower resolution features, sampled at 2ms (627 x 100), show comparable performance, they are also less robust against noise.

The features image goes through a stack of four convolutional layers, where we apply filters with varying receptive field: 8 x 8 and 6 x 6 respectively for the first and second layer, and 4 x 4 for the two uppermost convolutional layers (Figure A-8a). The convolution stride is fixed to 1 x 1. In addition, the padding of all convolutional layers is such that the dimensions of the input image are preserved after convolution. The depth of the convolutional filters is 16, 32, 16, 8, from the first to the fourth convolutional correspondingly.

After each convolutional layer, spatial pooling is conducted by max-pooling layers. The first poling is performed only along the time dimension, over a 2 x 1 window, with a stride of 2 x 1. The remaining poolings have a window size of 2 x 2, and a stride of 2 x 2.

[4]TensorFlow convolutions, unlike Keras, are actually cross-correlations, since the filter is combined with the input window without reversing the filter

The stack of convolutional layers is followed by three fully-connected layers. The first layer have 1024 hidden units and the second 512. The last one is the output layer, with softmax activation (Equation 4.2).

$$\sigma(z)_j = \frac{e^{z_j}}{\sum_{k=1}^{K} e^{z_j}} \tag{4.2}$$

where,

$$z = W_{out}x + b_{out}$$

Besides, all hidden layers, both convolutional and fully-connected, are equipped with the rectification ReLu non-linearity (Equation 4.3).

$$ReLu(x) = max(x, 0) \tag{4.3}$$

Furthermore, previous to all ReLu non-linearities we add the batch-normalization transform [17]. Thus, the output of a layer is given by Equation 4.4.

$$z = ReLu(BN(Wx + b)) \tag{4.4}$$

where the batch-normalization transform (BN) is defined as in Equation 4.5, likewise, μ and σ are estimated via mini-batch statistics[5], while the scaling factor γ and the shift β are parameters to be learned.

$$BN(x) = \frac{\gamma(x - \mu)}{\sigma} + \beta \tag{4.5}$$

For our concerns, BN has proved very helpful in reducing the dependency on hyper-parameter tuning, such as the initialization of the learning rate and weights. Considering the novelty of the proposed input coherence-print features, it is very arduous to set a proper parameter initialization, so we easily got stuck in poor local minima without using BN.

[5]They do not only depend on the current mini-batch, rather we update μ and σ, mini-batch by mini-batch, through exponential moving average with a $decay = 0.99$

4.2.4 Fully convolutional network

The architecture is the same as below, but we replace the last two hidden FC for convolutional layers to produce a fully convolutional network with 6 hidden layers.

In addition, we have reduced the receptive field of the convolutional filters, 4 x 4 for the first and second layers, and 2 x 2 for the rest of layers (Figure A-8b). The depth of the convolutional filters is 16, 32, 16, 8, 4 and 1 from the first to the sixth convolutional respectively.

After each convolutional layer, except the uppermost, which is connected with the output layer, max-pooling layers are used. The first poling is performed over a 2 x 1 window, with a stride of 2 x 1. The remaining poolings have a window size of 2 x 2, and a stride of 2 x 2. The batch-normalization transform is also applied before each ReLu activation.

4.2.5 Training procedure

Unlike the procedure adopted for the data of [15], where a MSE cost was minimized, herein we use softmax regression because the classes are mutually exclusive even though the labels' probabilities are not. Thus, we constrain the model to learn exclusive classes with an order given by the probability distribution label.

Therefore, the loss is defined as the cross-entropy function between the softmax normalized predictions and the distribution labels (Equation 4.6). We also apply L2 regularization with a weight decay $\lambda = 0.004$ to all learned variables (Equation 4.7). Moreover, we calculate the moving average of the total loss and we use these averages during evaluation, executed together with train (in parallel).

$$L_{xentropy}(w) = \frac{1}{N}\sum_{n=1}^{N} H(p_n, l_n) = -\frac{1}{N}\sum_{n=1}^{N} \tau_n log\hat{\tau}_n + (1-\tau_n)log(1-\hat{\tau}_n) \tag{4.6}$$

$$L_{total}(w) = L_{xentropy}(w) + \frac{\lambda}{2N}\sum_{w} w^2 \tag{4.7}$$

4.2.5.1 Gradient optimization

The training is guided by the gradient descendent Adam (Adaptive moment estimation) optimizer [18]. In contrast to traditional stochastic gradient descent, where a constant learning rate is used for all learned parameters, Adam computes individual adaptive learning rates for each network weight. These learning rates are separately updated from estimates of first and second moments of the gradient.

The algorithm is initialized with an initial learning rate $\alpha = 0.12$, exponential decay rates $\beta_1 = 0.9$ and $\beta_2 = 0.999$, for the first and second moment respectively and $\epsilon = 10^{-8}$ that regularizes a division in the updating step.

4.2.5.2 Details of learning

We trained our models using a mini-batch size of 64 examples. On one hand, we initialized the weights in each layer from "Xavier" initializer [13], with scaled uniform distributed random initialization. On the other, we initialized the neuron biases in all convolutional layers, with the constant 0. The biases of the fully-connected hidden layers are initialized with the constant 0.1, while the output layer biases are initially set to 0.004.

4.2.6 Improving generalization

Below, we describe the primary ways in which we improve generalization, in addition to the already presented L2 regularization.

4.2.6.1 Data augmentation

The most common practice to reduce overfitting, especially on image data, is to artificially enlarge the dataset using transformations. We transform the data in two different ways:

Flip vertical it consists of generating image vertical reflections, while keeping the same time-delay label.

Flip horizontal this reflection is, however, non-label preserving. Therefore, data is horizontally reflected and its label array is reversed.

Both transformations are performed randomly, at the time that data is collected in batches. Thence, batches are quite dissimilar among epochs.

Furthermore, we have optimized our input pipeline with queues and multi-threading, such as, batching and data augmentation are performed on the CPU at runtime, and enqueued into a FIFO queue. Afterwards, data is dequeued as requested by the GPU training thread. Therefore, this data augmentation strategy is indeed computationally free (neither disk space nor GPU resources are required). Moreover, the GPU throughput is increased, since it is released from secondary laborious tasks.

4.2.6.2 Dropout

Combining the predictions of different models is a very successful practice. This is the case of random forest, where the prediction capability of the model is improved by training various decision trees [5]. The efficient version of model combination for deep learning is dropout.

It consists of setting to zero the output of each hidden neuron with a given probability. Thence, the neurons which are dropped out do not participate in backpropagation. So every time an input is presented, the neural network samples a different architecture, but all these architectures share weights.

At test time, we use all the neurons but multiply their outputs by the dropout probability, which is an approximation of taking the geometric mean of all the virtually trained models.

We use dropout in all fully connected layers of the AlexNet-like network, with a probability of 0.5, prior to implementing batch-normalization. However, as stated in [17], after adding batch-normalization, we have reduced the dropout probability to 0.3 without losing generalization. Thus, we have speeded up the training.

Alternatively, as we are not using dropout in the fully convolutional network, we have increased the weight decay of the L2 regularization to $\lambda = 0.025$, only for this architecture.

Chapter 5

Results

Firstly, we have studied the feasibility of a real-time design. Afterwards, we have evaluated our model for the datasets of Table 3.3.

Datasets 1-4 contain each 46100 examples, which are composed of 32000, 6400 and 7700 examples for train, validation and test respectively. Likewise, dataset 6 represents the most general case, it is composed of 31600, 6760 and 6780 examples for train, validation and test correspondingly.

Datasets 1-4 analyze the robustness of various estimators against noise. The signal-to-noise ratio is therefore the only degree of freedom among these datasets. Alternatively, dataset 5 simulates a real scenario, where several species are present and the signal-to-noise ratio is constantly changing.

5.1 Methodology

We have compare the supervised estimations developed in this work (Section 4.2), the AlexNet-like and the fully convolutional neural networks, named CNN1 and CNN2 respectively, with the methods derived from the GCC family, comprising the GCC (Section 2.2.2.2) and the PHAT (Section 2.2.2.3).

Though the AED method (Section 2.2.3) was originally intended to be tested, it has proved to perform poorly in a first overview. It offers the weakest results in terms of error, but it is also computationally intensive. This method was initially

designed for room acoustics applications, where the number of samples is limited and full advantage can be taken from a convolutive mixture. However, when the number of samples becomes as large as 192000 samples per channel, the iterative process takes more than two minutes per example to converge. In addition, as experimented in [15], AED may bring poor results because of the absence of convolutive mixture.

For the purpose of a fair comparison, we have restricted the search of the time-delay estimator to the maximum lag of 0.625s, for both cross-correlation transforms, GCC and PHAT. Besides, all the neural network models CNN1, from dataset 2 to dataset 5, are fine-tuned versions of a master model, trained with dataset 1. Alternatively, CNN2, trained with dataset 1, generalizes well enough, so it has not been fine-tuned for any dataset.

Moreover, a first analysis shows that the data augmentation transformations are not well-behaved. They degrade the estimation accuracy in the validation and prediction dataset partitions. Therefore, we have determined that the time-frequency distribution of the coherence-features provides relevant information for time-delay estimation and should not be transformed.

5.2 Computation time

We have tracked the computation time of our model through TensorFlow debugging tools. The batch size is fixed to one example and the FIFO queue size is set to zero, simulating real-time conditions. Figure 5-1 shows the relative computation time, it can be seen that the pre-processing node is not the bottleneck of the model. The most demanding task is indeed the computation of the gradients and the optimization of the parameters (Adam). The pre-processing takes between 258 μs and 327 μs, while the optimization takes around 187 ms.

Considering these computation times as a guideline, a real-time design seems to be feasible. On the other hand, the GCC and PHAT are very efficient algorithms, which demand fewer resources than our pre-processing.

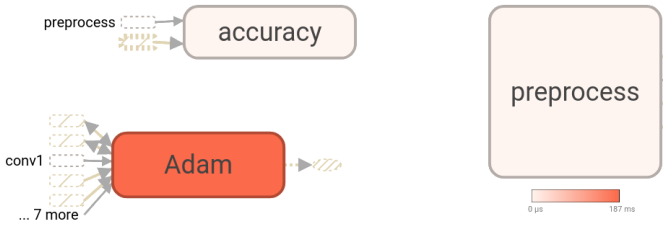


Figure 5-1: Part of the graph of the fully convolutional network. More demanding nodes, in terms of computation time, appears more reddish

5.3 First experiment

The performance of the GCC and PHAT estimators, as well as the the neural network approach, is evaluated with respect to noise.

5.3.1 First overview

Figure 5-2 shows the results of applying the GCC and CNN1 algorithms to rather narrowband whistles ($BW \approx 5kHz$), of dataset 2. The target and the output of the GCC and CNN1 are normalized to their respective maximums.

Upon initial inspection, the shape of the target distribution is much closer to the shape of the CNN1 than to that of the GCC. In fact, the CNN1 output is well-distributed around the target time-delay, which is one of our main objectives (Section 1.2). In addition, we have observed that this is the general behaviour. Actually, in some cases the output distribution is slightly skewed, pointing to the true time-delay. This provides an advantageous output for further post-processing.

Moreover, although the response of Figure 5-2 is normalized between 0 and 1, the softmax layer provide us valuable probability outputs. On one hand, when the estimation is poor, at very low SNR, CNN1 outputs very low probability values. On the other, as the conditions becomes more favourable (larger signals and higher SNR) CNN1 outputs greater values at its maximum peak. Therefore, a threshold time-delay detector can be implemented without extra-processing, testing between two simple hypotheses; the presence or absence of time-delay information.

The PHAT transform yields more noisy responses (Figure 5-3). However, PHAT

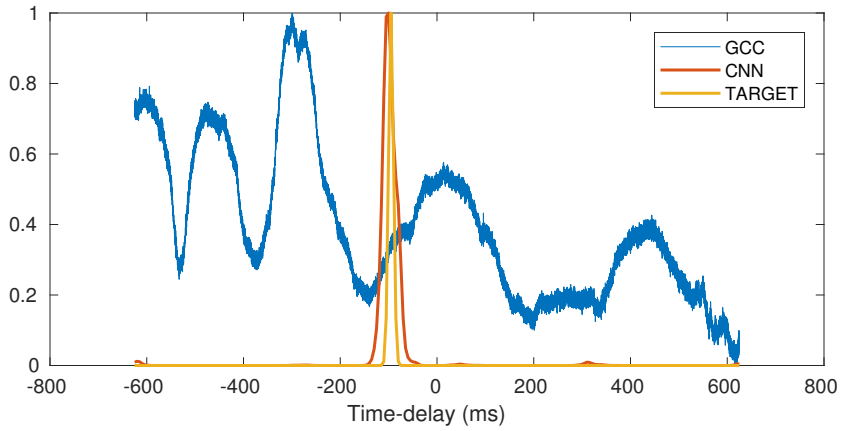


Figure 5-2: Output of GCC and CNN1 estimators with respect to the target time-delay. Input signals from dataset 2 at SNR = -6 dB, time-delay = -95 ms

resolves better the time-delay peak value, so it is also more accurate than the GCC in terms of mean error, as we will see.

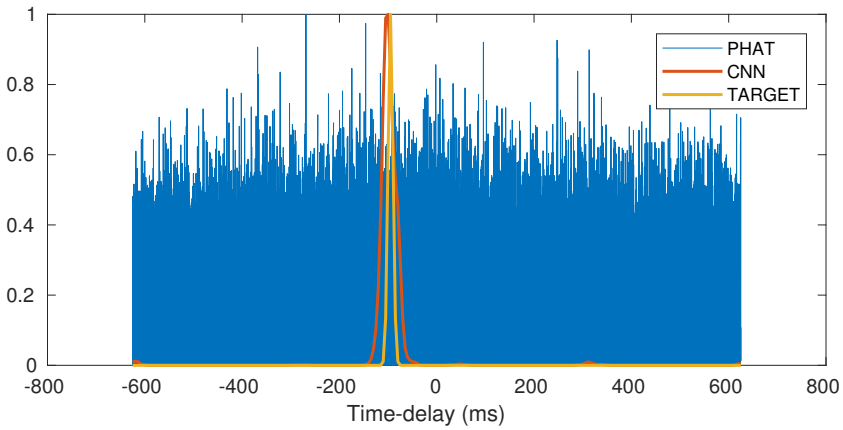


Figure 5-3: Output of PHAT and CNN estimators with respect to the target time-delay. Input signals from dataset 2 at SNR = -6 dB, time-delay = -95 ms

5.3.2 Statistical significance

Table 5.1 summarizes the evolution of the mean error and its standard deviation, as the SNR decreases. Clearly, both supervised approaches and PHAT outperform the GCC in all the evaluated scenarios, in terms of both mean error and standard deviation. Likewise, at 0 dB the PHAT transform and CNN2 show similar results, however, the estimations of CNN1 are weaker. Alternatively, CNN1 and CNN2 keep

the error confined as the noise increase, while the estimations of the PHAT are less reliable at negative SNRs.

SNR		XCOR		PHAT		CNN1		CNN2	
(dB)	Boxplot	μ_e	σ_e	μ_e	σ_e	μ_e	σ_e	μ_e	σ_e
0	A-9a	194.0	318.1	24.4	123.0	46.2	202.3	24.0	123.3
-6	A-9b	347.9	358.9	81.1	200.5	25.9	149.2	20.4	110.1
-9	A-10a	419.6	347.8	178.5	262.4	78.0	243.6	76.9	223.0
-12	A-10b	454.4	338.3	264.8	280.5	177.9	338.0	192.9	324.4

Table 5.1: Mean μ_e and standard deviation of the error in milliseconds as a function of the SNR (dB)

Moreover, although we expected that the error would increase monotonically as the SNR decrease, that is not the case for CNN1 and CNN2 at -6 dB. Likely, it is because the signals of dataset 2 are better represented in the coherence-print features[1], than that of dataset 1.

5.4 Second experiment

This experiment evaluates the performance of the estimators under scrutiny with dataset 5, which is composed of 3 different species. In addition, the SNR is varying from -18dB to 18dB.

SNR		XCOR		PHAT		CNN1		CNN2	
(dB)	Boxplot	μ_e	σ_e	μ_e	σ_e	μ_e	σ_e	μ_e	σ_e
$\in [-18, 18]$	A-11	194.6	318.2	69	188.6	105.1	287.8	46.9	174.6

Table 5.2: Mean μ_e and standard deviation of the error in milliseconds as a function of the SNR (dB)

Table 5.2 outline the results of applying the GCC, PHAT, CNN1 and CNN2 algorithms to the 3 species covered in this work for varying SNR. The observed tendency is the same as in the previous experiment, both PHAT and neural networks

[1]As noted in 4.2.1, wider signals in time and frequency produce clearer coherence-print features

perform better than the GCC. Besides, the estimations of CNN2 are the most accurate in terms of both, mean and standard deviation of the error.

Chapter 6

Conclusions and future development

6.1 Conclusions

In general, we have fulfilled our main objectives (Section 1.2), designing a time-delay database for cetacean localization and training a model for time-delay estimation.

Besides, although we have not been able to train an end-to-end deep learning model, we have worked around the curse of dimensionality with high-level features. In addition, we have proved their robustness against low SNR, which represents more real scenarios. We have also realized that the fully convolutional neural network fit better our features, providing the best results in a general scenario.

On the other hand, we have not faced the special cases proposed in Section 3, comprising limit cases and multi-channel time-delay estimation (more than two channels). Although these cases are already implemented in our data generation application, we have not considered them for the proposed final model.

6.2 Future development

We have developed an estimator based on coherence-print features and convolutional neural networks, but the results herein presented can be potentially outperformed.

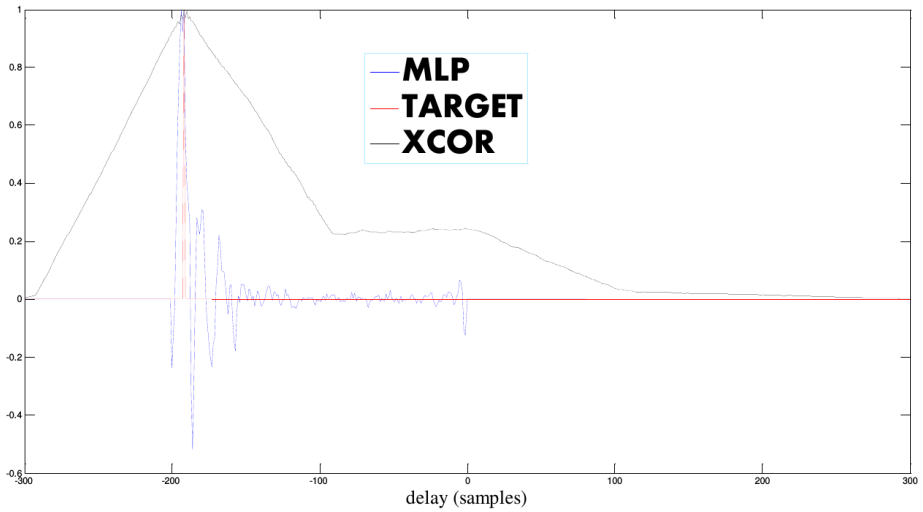
Therefore, we propose the following future development:

- To develop an end-to-end, low-features based, model.
- To adapt and evaluate the model for limit cases and multi-channel.
- To add a convolutive mixture in the signals.
- To move the time-delay data generation software from Matlab to TensorFlow, in such a way that the time-delay segments are generated at runtime. This procedure is not computing demanding, so a new batch of data can be generated by the pre-processing thread before being required by the training thread. However, in the current implementation, the learning capabilities of the model are limited by memory resources.

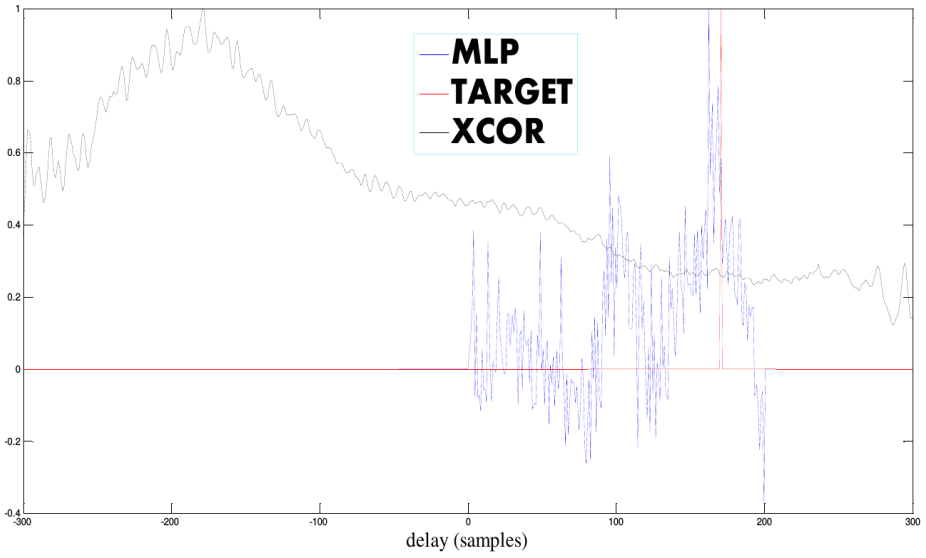
Mainly, we expect that an end-to-end low-features based model can make one step forward, outperforming the results herein presented in terms of accuracy and computation time.

Appendix A

Figures

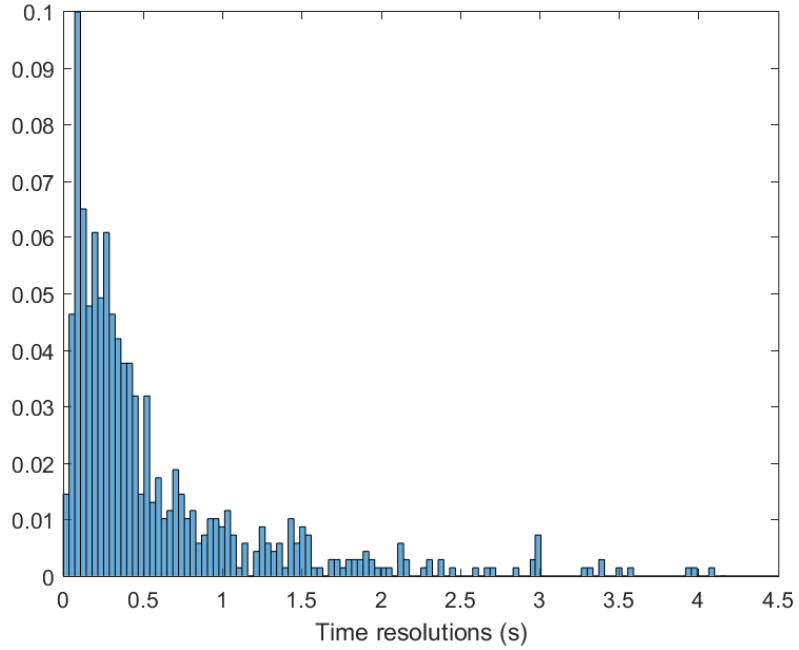


(a) $SNR = \infty$

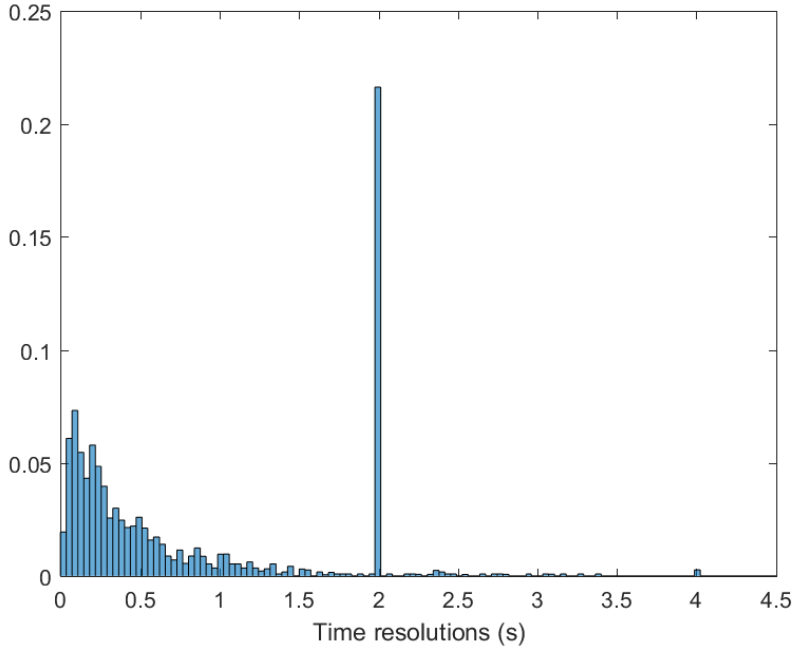


(b) $SNR \in [5, 1]$

Figure A-1: Output of the neural network and cross-correlation estimators with respect to target. Figure credit from [15]



(a) palmyra092007-070924-205305 dataset



(b) Qx-Tt-060814-123433 dataset

Figure A-2: Time resolutions pdf

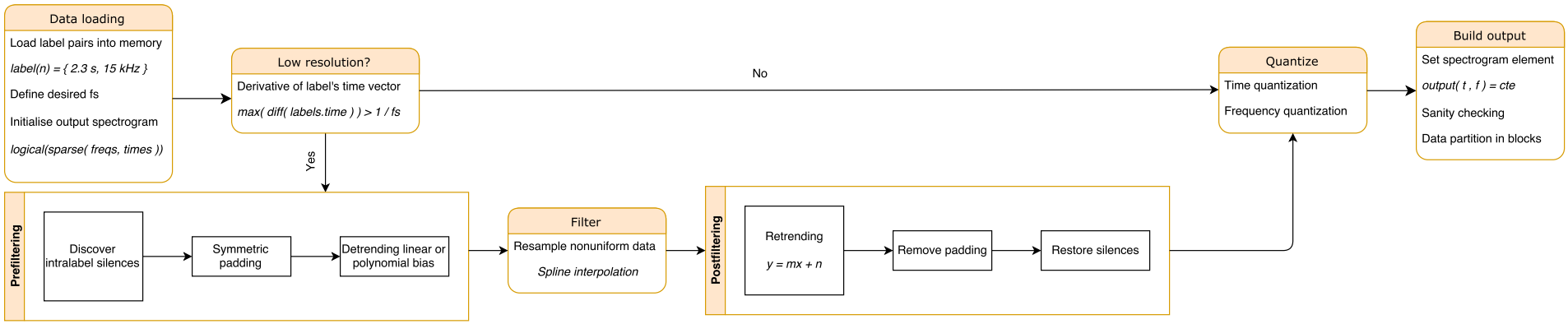


Figure A-3: Annotations to spectrogram

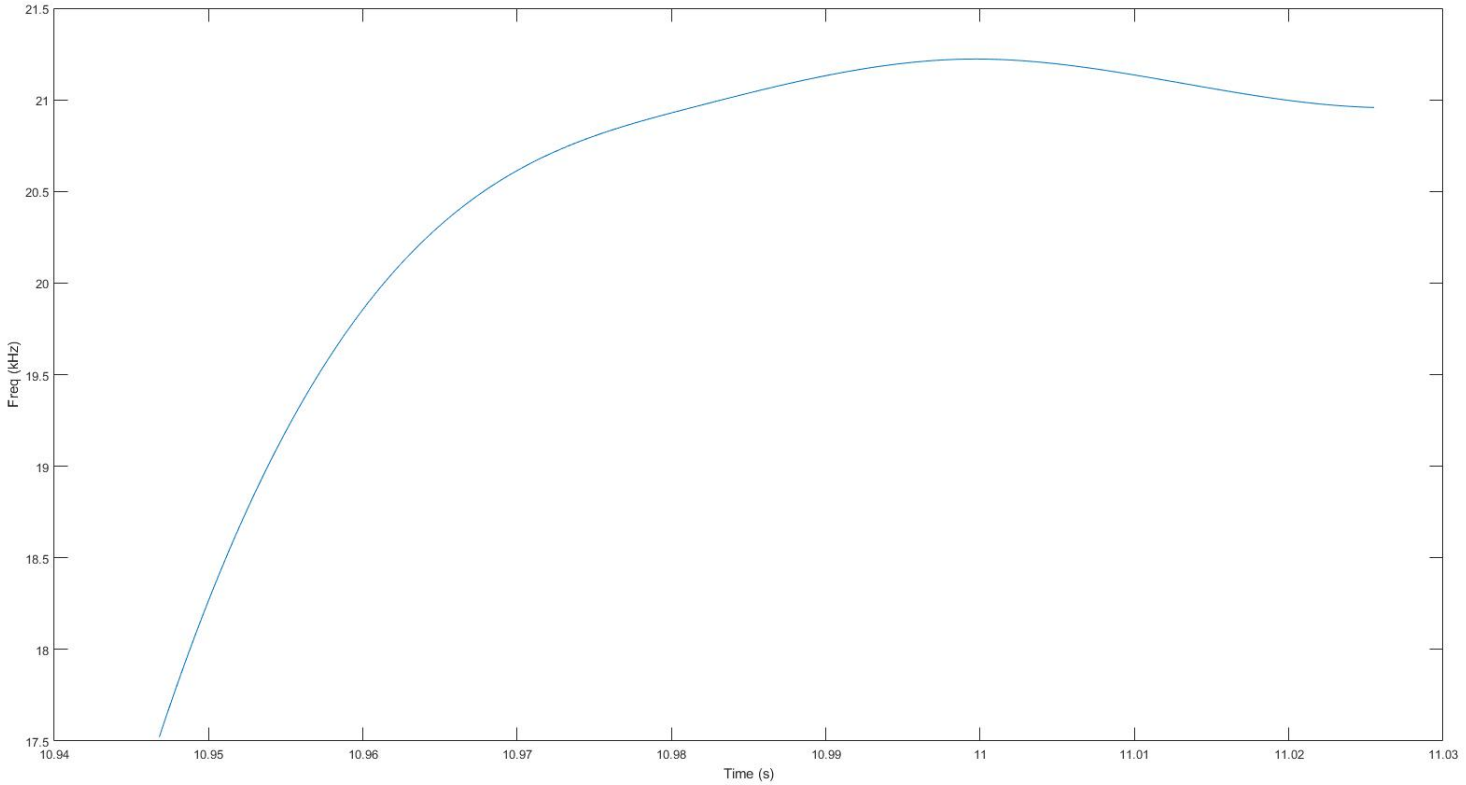


Figure A-4: Bottlenose dolphin's whistle labeled as a time-frequency contour

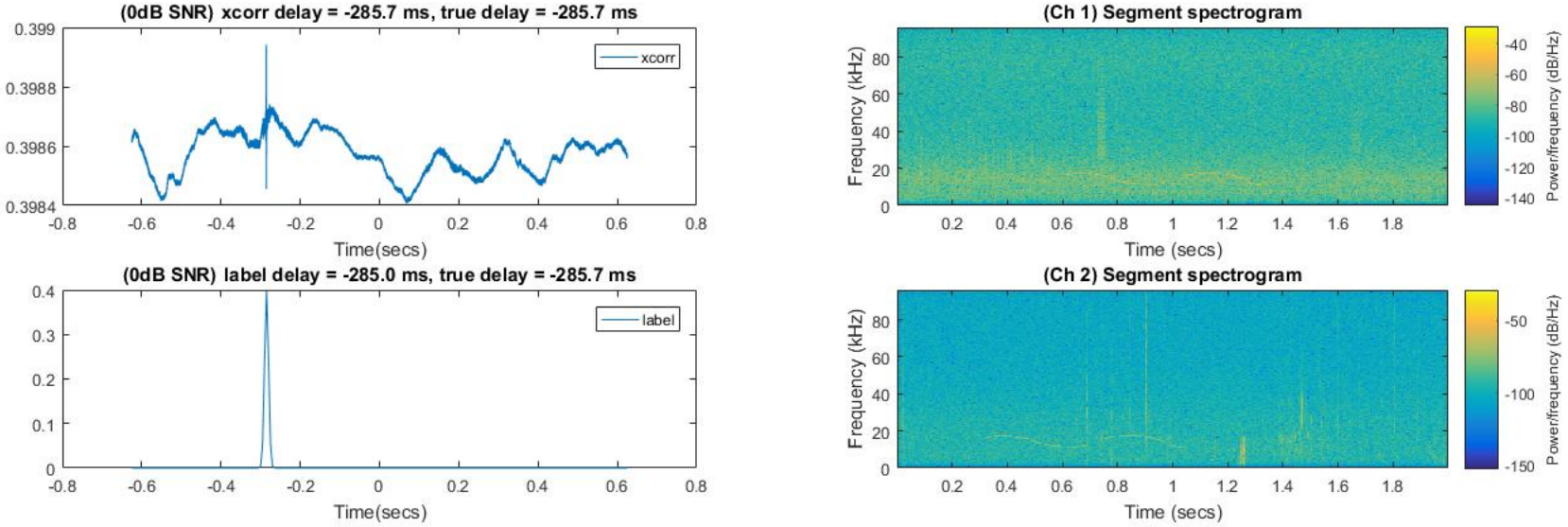


Figure A-5: Generated 2-channel time-delay segment

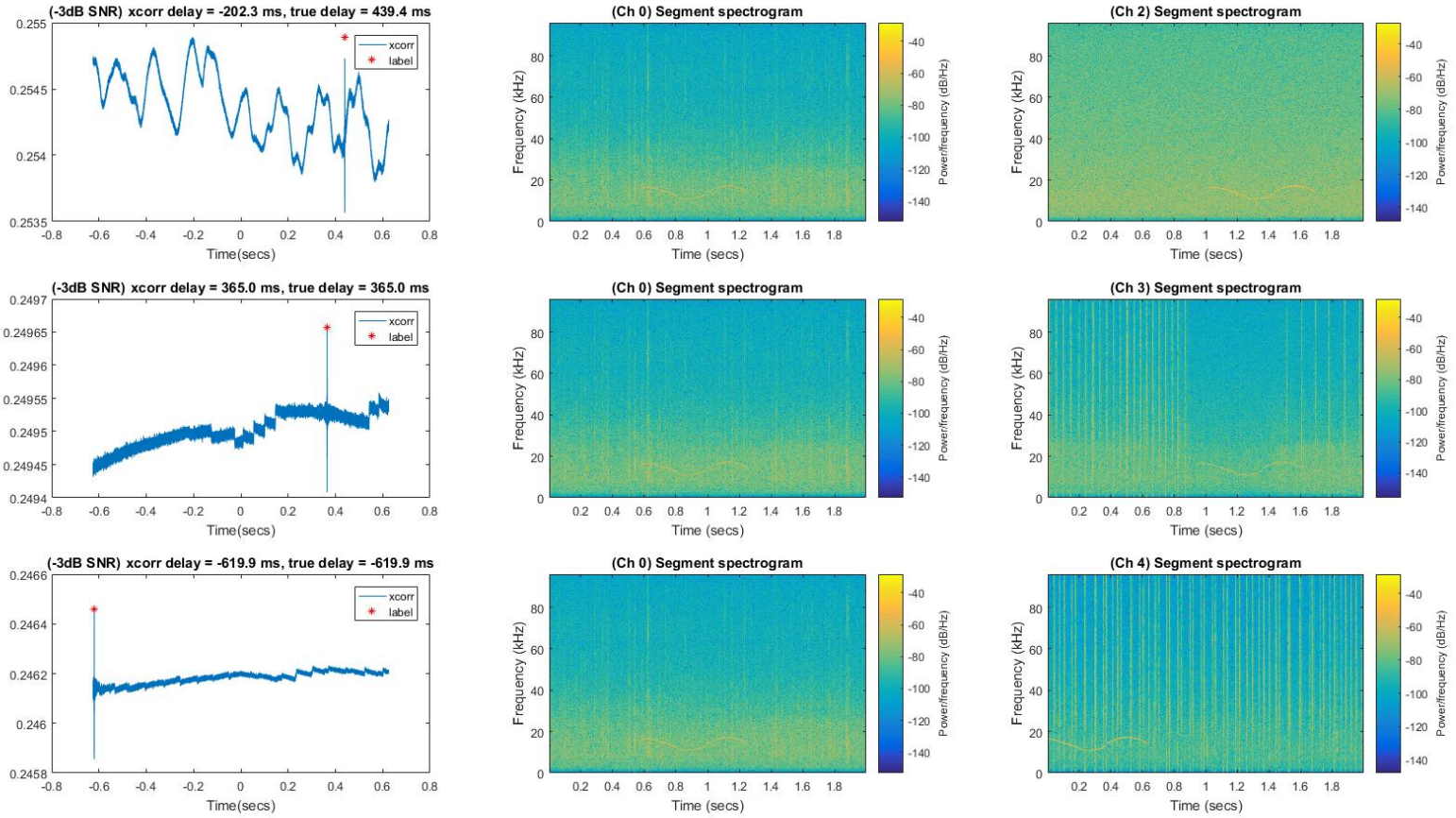


Figure A-6: Generated multi-channel time-delay segment

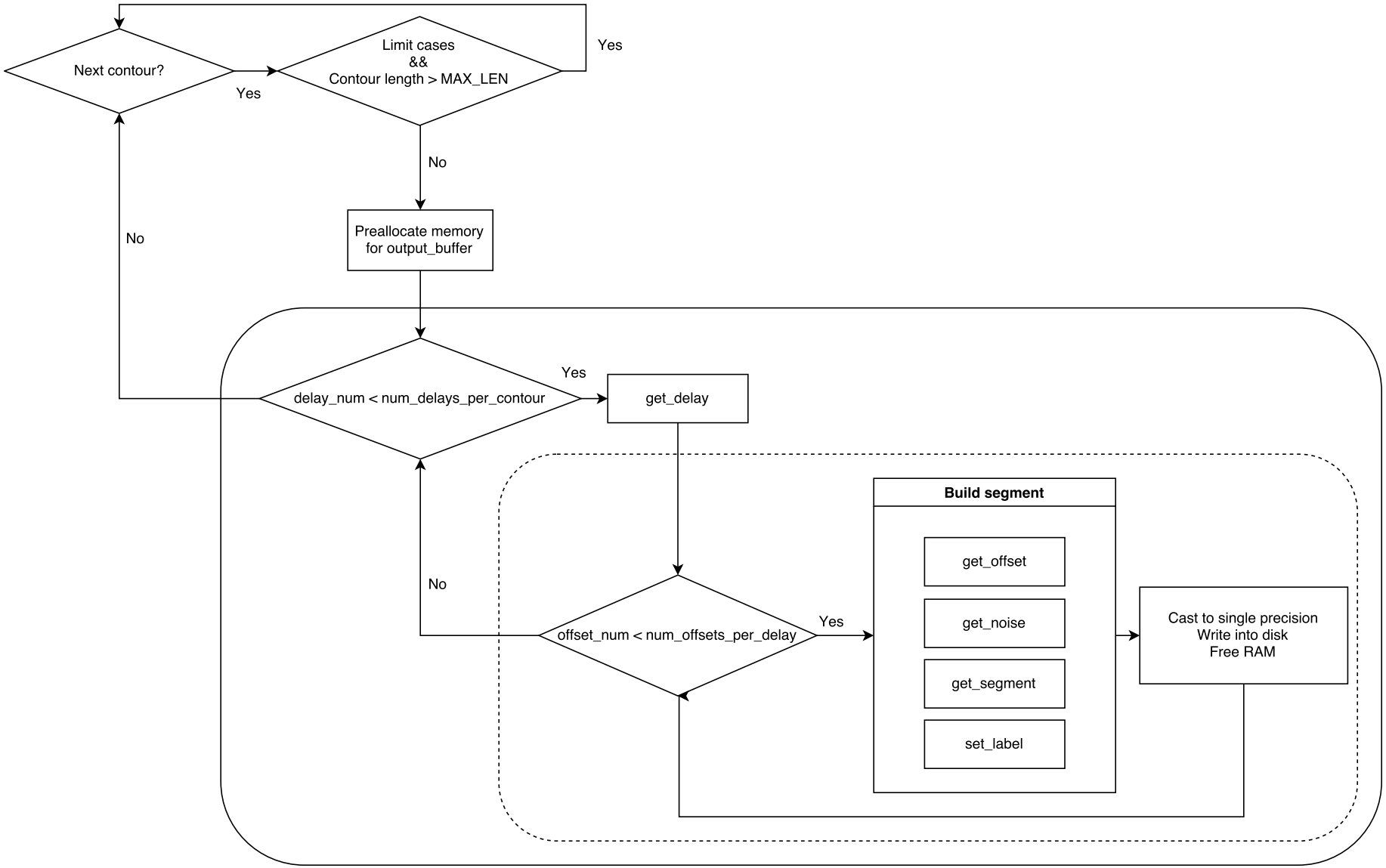
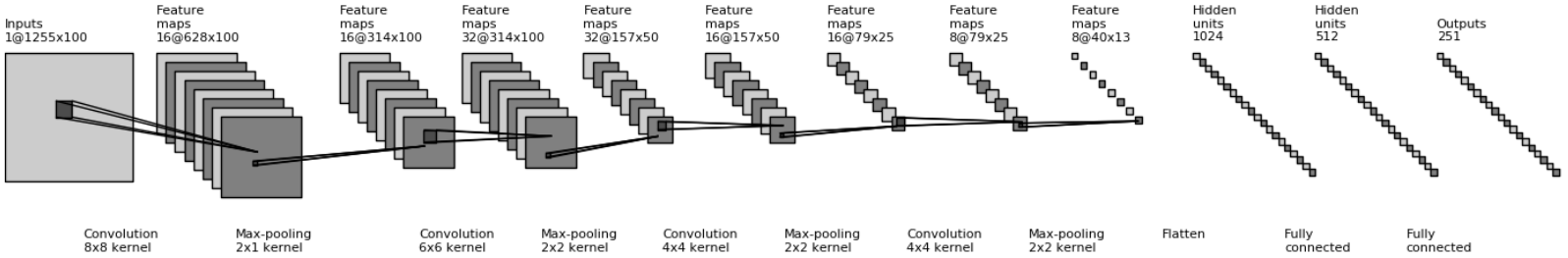
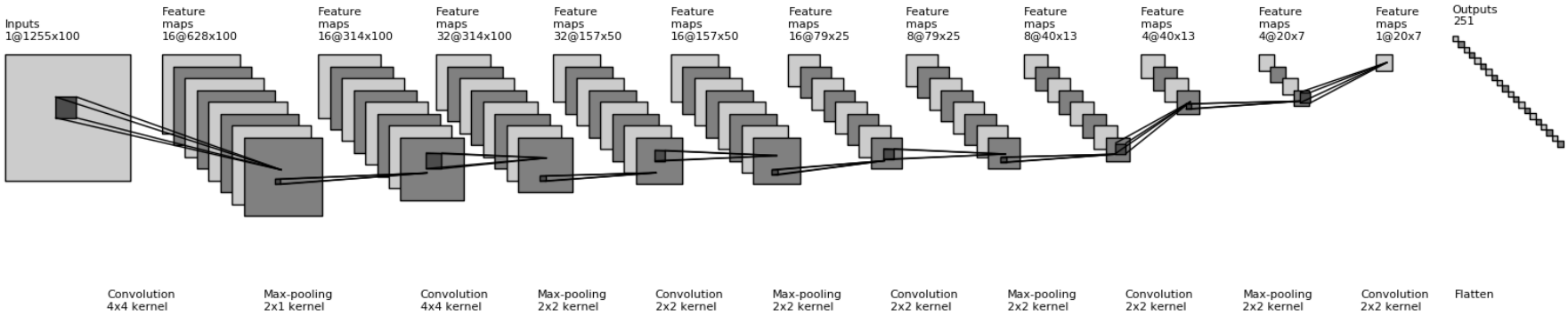


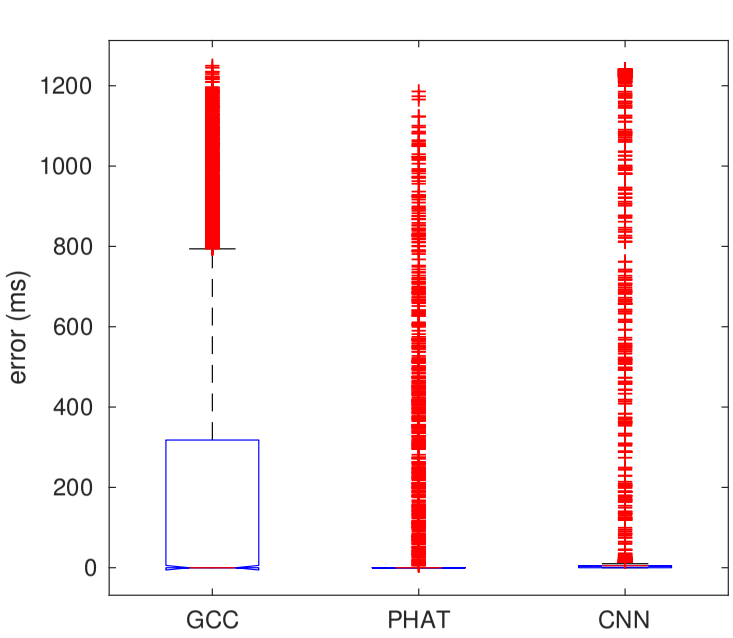
Figure A-7: Building segments



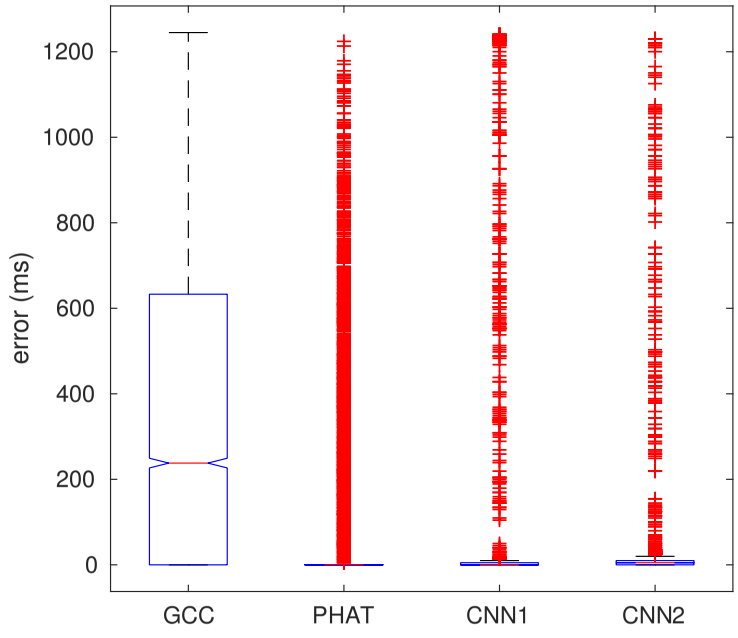
(a) Time-delay estimation AlexNet-like architecture



(b) Time-delay estimation fully CNN architecture

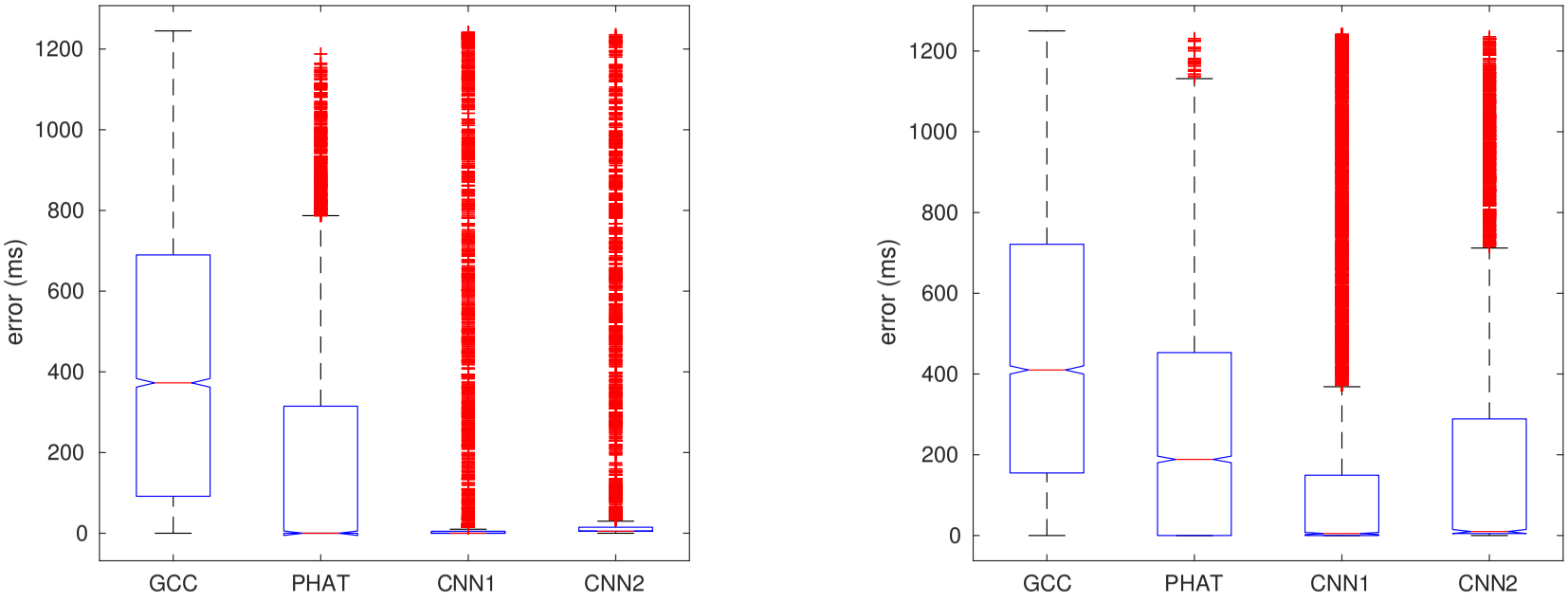


(a) Distribution of error for SNR = 0 dB



(b) Distribution of error for SNR = -6 dB

Figure A-9: Boxplot of the error of various estimators



(a) Distribution of error for SNR = -9 dB

(b) Distribution of error for SNR = -12 dB

Figure A-10: Boxplot of the error of various estimators

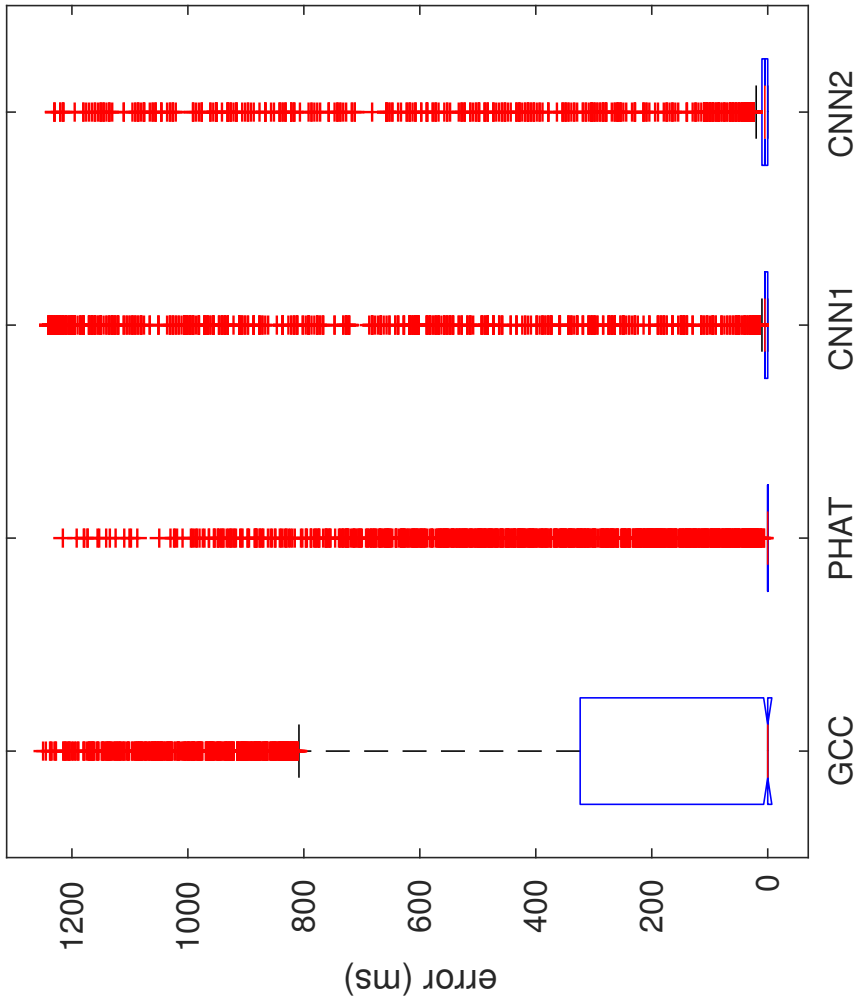


Figure A-11: Distribution of error for varying SNR

Bibliography

[1] Roland Aubauer. *Korrelationsverfahren zur Flugbahnverfolgung echoortender Fledermause*. VDI-Verl., 1995.

[2] Simone Baumann-Pickering, Sean M Wiggins, John A Hildebrand, Marie A Roch, and Hans-Ulrich Schnitzler. Discriminating features of echolocation clicks of melon-headed whales (peponocephala electra), bottlenose dolphins (tursiops truncatus), and grays spinner dolphins (stenella longirostris longirostris). *The Journal of the Acoustical Society of America*, 128(4):2212–2224, 2010.

[3] J. Benesty. Adaptive eigenvalue decomposition algorithm for passive acoustic source localization. *The Journal of the Acoustical Society of America*, 2000.

[4] J. Benesty, Y. Huang, and J. Chen. Time delay estimation via minimum entropy. *Signal Processing Letters, IEEE*, 14(3):157–160, 2007.

[5] Leo Breiman. Random forests. *Mach. Learn.*, 45(1):5–32, October 2001.

[6] Jane Bromley, Isabelle Guyon, Yann LeCun, Eduard Säckinger, and Roopak Shah. Signature verification using a "siamese" time delay neural network. In J. D. Cowan, G. Tesauro, and J. Alspector, editors, *Advances in Neural Information Processing Systems 6*, pages 737–744. Morgan-Kaufmann, 1994.

[7] G. Carter. Time delay estimation for passive sonar signal processing. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 29(3):463–470, Jun 1981.

[8] G. C. Carter. Coherence and time delay estimation. *Proceedings of the IEEE*, 75(2):236–255, Feb 1987.

[9] Soumitro Chakrabarty, Emanuël Habets, et al. Broadband doa estimation using convolutional neural networks trained with noise signals. *arXiv preprint arXiv:1705.00919*, 2017.

[10] Jingdong Chen, J. Benesty, and Yiteng Huang. Robust time delay estimation exploiting redundancy among multiple microphones. *IEEE Transactions on Speech and Audio Processing*, 11(6):549–557, Nov 2003.

[11] Jingdong Chen, Jacob Benesty, and Yiteng(Arden) Huang. Time delay estimation in room acoustic environments: An overview. *EURASIP Journal on Advances in Signal Processing*, 2006(1):026503, May 2006.

[12] Sara Heimlich et al. 5th workshop: Portland, 2011. `http://www.mobysound.org/workshops_p2.html`.

[13] X. Glorot and Y. Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS10). Society for Artificial Intelligence and Statistics*, 2010.

[14] M. Hadley. Tracking sperm whales using passive acoustics and particle filters. February 2011.

[15] Ludwig Houégnigan, Pooyan Safari, Climent Nadeu, et al. Neural networks for high performance time-delay estimation and acoustic source localization. *Journal of Computer Science and Information Technology*, 2017.

[16] Ludwig Houegnigan, Mike van der Schaar, Marta Solé Carbonell, Pablo Pla Caro, Alba Solsona Berga, and Michel André. Neural networks for the localization of biological and anthropogenic source at neutrino deep sea telescope. *OCEANS 2015-Genova. IEEE*, 2015.

[17] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.

[18] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2014.

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

[20] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 1097–1105. Curran Associates, Inc., 2012.

[21] J. Krolik, M. Joy, S. Pasupathy, and M. Eizenman. A comparative study of the lms adaptive filter versus generalized correlation method for time delay estimation. In *ICASSP '84. IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 9, pages 652–655, Mar 1984.

[22] Thomas A. Lampert and Simon E.M. O'Keefe. On the detection of tracks in spectrogram images. *Pattern Recognition*, 46(5):1396 – 1408, 2013.

[23] Breiman Leo. Statistical Modeling: The Two Cultures (with comments and a rejoinder by the author). *Statistical Science*, 16(3):199–231, 2001.

[24] David K Mellinger, Kathleen M Stafford, Sue E Moore, Robert P Dziak, and Haru Matsumoto. An overview of fixed passive acoustic observation methods for cetaceans. *Oceanography*, 20(4):36–45, 2007.

[25] Julie N. Oswald, Shannon Rankin, Jay Barlow, and Marc O. Lammers. A new tool for realtime acoustic species identification of delphinid whistles. *The Journal of the Acoustical Society of America*, 118(3):1909–1909, 2005.

[26] Samir Shaltaf. Neural-network-based time-delay estimation. *Eurasip Journal on Applied Signal Processing*, 2004:378–385, 2004.

[27] Samir Shaltaf and Ahmad A Mohammad. Neural networks based time-delay estimation using dct coefficients. *American Journal of Applied Sciences*, 6(4):703, 2009.

[28] Melissa S Soldevilla, E Elizabeth Henderson, Gregory S Campbell, Sean M Wiggins, John A Hildebrand, and Marie A Roch. Classification of rissos and pacific white-sided dolphins using spectral properties of echolocation clicks. *The Journal of the Acoustical Society of America*, 124(1):609–624, 2008.

[29] Jeanette A Thomas and Cynthia Moss. *Echolocation in bats and dolphins*. University of Chicago Press, 2004.

[30] San Diego State University. Tethys workbench software, 2012. `http://tethys.sdsu.edu/`.

[31] F.G. Wood. *Marine Mammals and Man: The Navy's Porpoises and Sea Lions*. R. B. Luce, 1973.

[32] Walter MX Zimmer. *Passive acoustic monitoring of cetaceans*. Cambridge University Press, 2011.