Master's Thesis

Spatial Error Concealment in Ad-hoc Audio Conferencing Systems

Reza Parhizkar

NOKIA Research Center-Lausanne
Audiovisual Communications Laboratory (LCAV)
Ecole Polytechnique Fédérale de Lausanne

Supervisors

Matti Hämäläinen
Mihailo Kolundžija
Prof. Martin Vetterli

August 2009

Abstract

In this work we consider an ad-hoc audio conferensing system based on VoIP services in which the participants connect to the conference using mobile communication devices with wireless connectivity. To overcome possible quality problems in the wireless link in this configuration, we propose improvements to the existing conferencing systems.

Some networking modifications are suggested to increase the channel capacity and robustness from the conference server to multiple clients. On the other hand, for the improvement of the uplink quality, we suggest a new spatial error concealment method, where a backup device captures and sends the audio signals to the server together with the primary device. In the server the lost frames from the primary channel are estimated based on the backup signal.

Several methods for estimating the primary signal based on the backup signal are studied. The results of the methods are evaluated by a psychoacoustic error metric based on Zwicker's loudness model. An informal subjective test is also performed to compare the results of these methods in order to chose one for implementing on the real-time conferencing setup.

Both objective and subjective tests show consistent results and confirm that usage of spatial error concealment improves significantly the audio quality in the primary signal.

Acknowledgments

I would like to express profound gratitude to my advisor in Nokia Research Center-Lausanne, Matti Hämäläinen. Thank you for your continuous support during the project and for teaching me how to deal with the technical difficulties in industry and thank you for always being there in the difficulties.

I would like to thank my advisor in EPFL, Mihailo Kolundžija. Without your guidance and persistent help this thesis would not have been finished.

I would like to express the deepest appreciation to Prof. Martin Vetterli. It has been a great pleasure for me to know you and work with you during my several projects in your lab. Thank you for accepting me in LCAV and for the enthusiasm and inspiration you have been always giving me.

I am thankful to all my colleagues in Nokia Research Center-Lausanne for accepting me warmly and providing me the opportunity to work, to learn and to have fun. Special thanks to my friend, Mahdad Hosseini Kamal, who accompanied me on these project during this six months and left me unforgettable memories.

I am grateful to Jussi Mutanen, Jussi Virolainen, and Kai Samposalo, for their kind meetings and helpful guidance during this work. Thank you for sharing you work with us and for the hours you spent on answering every single question we had.

I would also like to thank Ville Myllylä for providing the implementation of adaptive variable step-size method. I am also thankful to Henri Toukomaa for sharing the subjective test software and for computing the results.

I am as ever, especially indebted to my family. Thank you for your unconditional love, support and understanding throughout my studies. I would like to dedicate this thesis to you.

Contents

List of Figures

Chapter 1

Introduction

Audio conferencing systems serve as an efficient communication tool in different applications such as teleconferencing systems or the communication systems in auditoriums. Most of these systems are centralized on a server (also called conference bridge) to which all the participants in the conference connect [1]. Traditional teleconferencing systems use circuit switched technologies for carrying the voice, but with development of the internet services, we are witnessing the migration of these applications to the voice over internet protocol (VoIP) services.

Moreover, with the developments in the mobile communications, wide range of capabilities, including wireless connectivity, are now integrated into the handheld communication devices. This ability allows these devices to contribute in the paradigm shift in basing the communications on IP, and makes them potential clients for such audio conferencing systems.

The advantage of employing existing handheld devices in the audio conferencing systems is the cost saving due to utilization of existing hardware and the cost-efficiency of VoIP services in general. Furthermore, in audio conferencing systems based on the wired implementations, the installation costs increase nonlinearly with the number of potential participants in the conference; on the other hand if one could implement such systems using wireless ad-hoc networks, not only the installation costs would decrease significantly, but also the scalability of the system would be mainly dependant on the capabilities of the conference server and the capabilities of the wireless network.

Our goal is to construct an audio conferencing system, based on VoIP services and wireless LAN connectivity of mobile communication devices. We call this configuration an ad-hoc audio conferencing system. The challenge is that these audio communication systems demand high-quality services, whereas the wireless IP networking techniques applied to real-time commu-

nications introduce errors and packet losses. Thus, the problem changes to the implementation of high quality communications in unreliable media.

In this thesis, we consider audio conferences between two or more acoustic spaces, because shared acoustic space systems demand accurate delay management which is not in the scope of this project. We will address the problem where groups of participants join a conference and the system servers as an inter-group communication tool. An example of such system is illustrated in Fig. 1.1. In this figure, there are participants in two (or more) rooms and an audio conference is established between these groups while room internal connectivity is wireless. The task is to provide a high quality audio between the participants in this configuration. In this work an existing audio

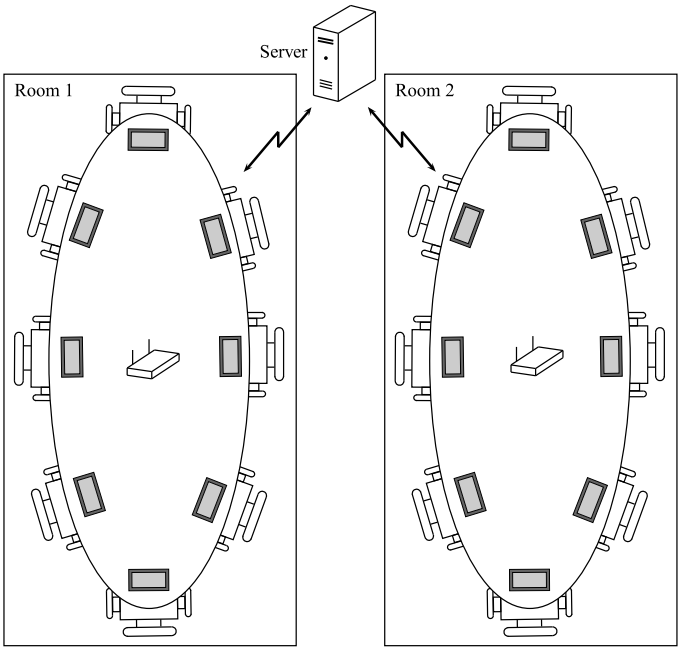


Figure 1.1 – Prototype of an ad-hoc audio conferencing setup. Two groups of participants are in two separate acoustic spaces. In each room, there is a wireless access point which connects the clients to the server.

conferencing system is adopted and improvements are applied on it to allow loss-free wireless conferencing. We will divide the problem to two separate parts: the *downlink* which is the data flow from the conference server towards the devices of the conference participants, and the *uplink* which is the data flow from the mobile devices to the conference server.

To improve the quality of downlink, networking approaches are introduced to guarantee low rate of packet loss in the downlink. On the other

hand, for the uplink problem, we propose to use multiple devices for capturing the voice and sending it to the server. In addition to a primary device, a backup device also captures the voice sends it to the server. In the server the lost frames in the primary signal are replaced by estimations based on the backup signal. this is graphically represented in Fig. 1.2. For estimating the primary signal from the backup signal, the transfer function from the backup signal to the primary signal needs to be found.

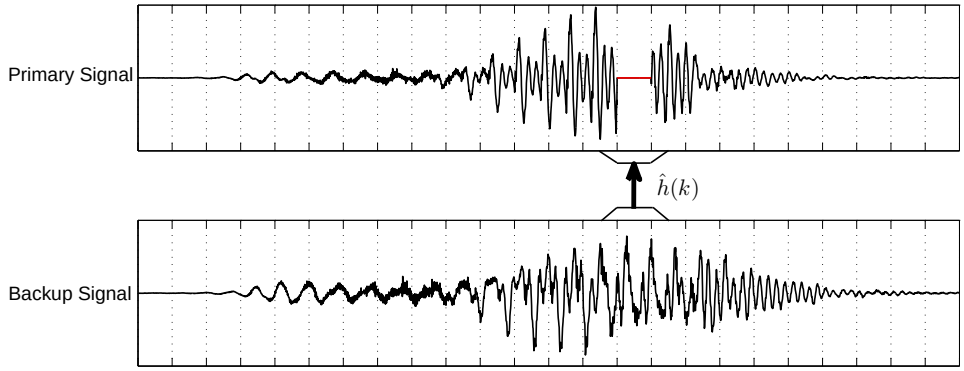


Figure 1.2 – The primary signal with a lost frame and the backup signal. Frames are separated by dashed lines. The goal is to replace the missing frame with a modified one from the backup signal.

The thesis is organized as follows; in Chapter 2 the system components are introduced in more details. The downlink problem is also addressed and brief solutions are given. Then error concealment in the uplink is mentioned in more details.

In Chapter 3, a thorough investigation of the transfer function estimation techniques is done and some discussions are provided for evaluating the performance of each method. Some of these methods approximate the transfer functions by a signal delay-and-gain and some try to give a more realistic estimation of the transfer function.

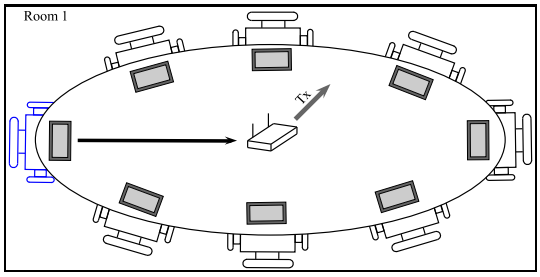
In Chapter 4, studied methods in Chapter 3 are used in the proper context for spatial error concealment. Performances of the methods are evaluated with objective psychoacoustic tests in this chapter. The results of an MOS subjective test are also provided in this chapter.

Finally in chapter 5, the summary and conclusion is given and some future work is proposed.

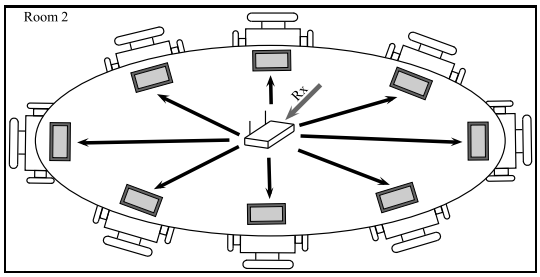
Chapter 2

Audio Conferencing System

The used audio conferenceing system is based on VoIP services and consists of a server and clients which are the running on the devices held by the participants in the conference. Packet exchange is done using SIP and RTP protocols, which are described in detail in appendices A and B, respectively. Let's consider the problem setup in a simple case where we have only two rooms as in Fig. 2.1.



(a) Uplink



(b) Downlink

Figure 2.1 – Data flow in the conference system prototype. A participant in room 1 is speaking, the packets are forwarded to the server which forwards them to the participants in room 2.

Every participant in the conference holds a Nokia N810 device which captures the speech signal and communicates the packets to the mixing server. The mixing server mixes different signals coming from the participants in the conference. The aim of this project is to guarantee a high quality uplink from the speaker device to the mixing server and a high quality and scalable downlink from the mixing server to the clients.

2.1 System components

The audio conferencing system consists of two key components; the client application and the mixing server which are briefly described in the following sections.

2.1.1 Mixing Server

A mixing server in an audio conferencing system, is an application to which all the participants in the conference connect. The duty of the mixing server can be summarized as receiving the packets from several clients, extract the audio contents from each packet, mix the packets according to a mixing policy and forward them to the destined listeners in the conference.

For the work done in this thesis, an existing audio teleconferencing server is used as the basis for the extensions. This system is developed by Nokia Devices R&D and is able to connect clients using VoIP. The system functionality is based on SIP and RTP packet exchange.

Several participants in the same or different acoustic space make independent calls (unicast connection) to the conference mixing server. For each client application a default room number (which corresponds to the acoustic space the user is in) is associated. The room number can be changed anytime during the conference by the user via the client application. If the room numbers for two clients are the same, the mixing server assumes that the participants are in the same acoustic space, thus the mixing policy acts in a way that no acoustic feedback comes from the devices in the same room, i.e. packets are not forwarded to the room they are coming from.

The mixing server components are depicted in Fig. 2.2. In this architecture, the SIP stack takes care of initializing the connection and signaling packets. The RTP processor receives (or sends) the RTP packets and extracts (or encapsulates) the audio contents of the packet. The control unit in association with the user interface manages the SIP and RTP processing. The audio pre-processor, which is added to the mixing server in this thesis, is responsible for the acoustic error concealment. Finally, the audio mixer com-

bines the received signals according the rules defined by the mixing policy module.

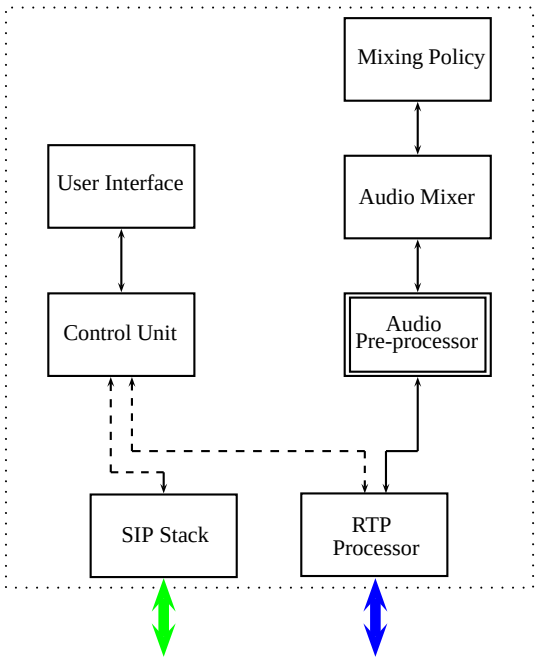


Figure 2.2 – Audio conferencing mixing server components. The audio pre-processor module is added as an extension to the mixing server and The RTP processor component is modified for the intended improvements.

In the prototype system, the mixing server is running on Red Hat Enterprise Linux on a low performance IBM T41 laptop.

2.1.2 VoIP Client

The client is a VoIP application which is able to have Dual-tone multi-frequency (DTMF) signaling with the server or other clients. For this thesis, the standard integrated VoIP client in Nokia N810 internet tablets is used.

Different modules of this application is shown in Fig. 2.3. The application consists of three modules; The user interface takes care of the application logic, the stream engine is responsible for audio streaming logic and the connection manager implements the signalling logic. These three modules communicate to each other via the D-Bus[1]. For the intended changes in this thesis we modify the stream engine module of this software.

[1]D-Bus (Desktop Bus) is a message bus system, a simple way for applications to

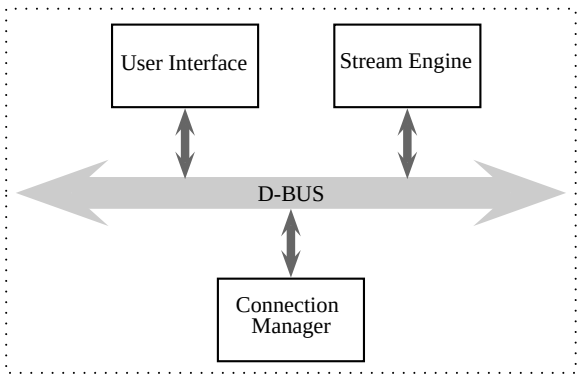


Figure 2.3 – Audio conferencing client application components. The stream engine module is modified in this thesis to be consistent with the changes in the server application.

2.2 Telconferencing system extensions

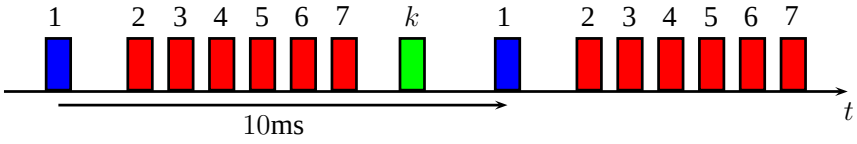
The teleconferencing system mentioned in the previous section is designed for VoIP teleconferencing applications and no extra processing is included in the software for possible packet drops due to the wireless connectivity. In other words, in these applications, packet loss is not compensated. In this sense, to improve the system, two extensions to this system are proposed; Firstly *radio packet loss avoidance* in the downlink and secondly *acoustic error concealment* in the uplink. These methods are briefly mentioned in the sequel. Note that in this thesis, it is assumed that at each time instance, from the participants who share the same acoustic space, only one is speaking, the reason for this assumption is described in the next chapter.

2.2.1 Radio packet loss avoidance in downlink

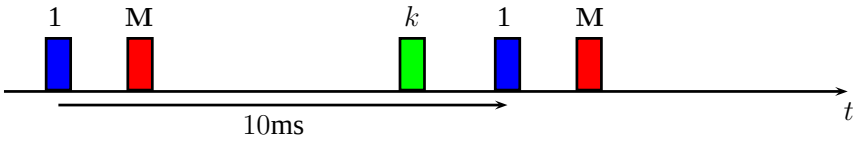
One way of improving the quality of service is to prevent the system to lose the packets in the downlink. Since we assume that at each time instance only one participant is speaking, all the others in the conference are listening to the same speech signal and they are all in the same network, so it is possible to change the multiple unicast downlink connections between the server and each client to a multicast connection to all of them. Some signaling can

talk to one another. It is primarily developed by Red Hat, as part of the freedesktop.org project and released under the terms of the GNU General Public License and the Academic Free License. D-Bus is free software. For more information refer to http://www.freedesktop.org/wiki/Software/dbus.

be done to avoid the acoustic feedback problem in the configuration. This modification is depicted graphically in Fig. 2.4. Client number 1 sends an audio frame (blue) to the mixing server and the server forwards the packet to clients 2 to 7 (red). Once in a time there is a signaling between the server and client number k (green) and the procedure is repeated after for every audio frame (10ms long in this example).



(a) Unicast downlink



(b) Multicast downlink

Figure 2.4 – Packet flow over time for unicast downlink versus multicast downlink from the mixing server to the clients. By changing the downlink to multicast, one can save bandwidth of the radio channel.

With this process, large amount of the bandwidth in the channel and buffer in the server stack is saved and this improves the scalability of the system. Moreover, the saved bandwidth can be utilized to improve robustness against packet losses by sending multiple multicast packets, so that if one packet is lost, the repeated one is reached to the client. With applying redundant packets, the packet flow will be as in Fig. 2.5.

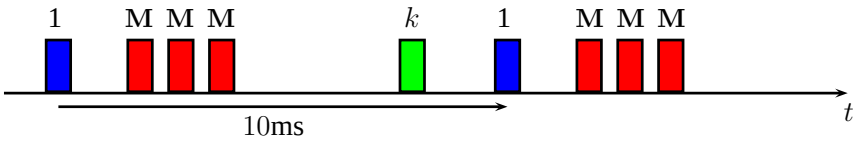


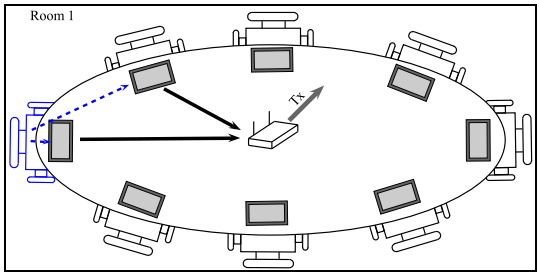
Figure 2.5 – Packet flow over time for multiple multicast packets. Multiple multicast packets are sent in the downlink in order to recover packet losses.

In this case, if none of the multiplications are lost, multiple copies of the same packet reach the client. To distinguish the duplication and remove

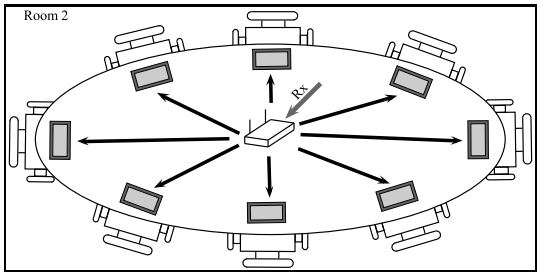
redundant frames, we compare the sequence numbers in the RTP packet headers (refer to appendix B). If the sequence number of an arriving packet is the same as the previous one, the packet is detected as redundant and is removed from the buffer.

2.2.2 Acoustic error concealment in uplink

In contrary to the downlink problem, utilization of multicast communication is not feasible in the uplink problem. due to the assumption of many devices in an acoustic space, it is possible to capture the speaker signal by two (or more) microphones and send the audio content by multiple different channels. In other words, a spatial error concealment is proposed for the uplink problem. This is illustrated in Fig. 2.6.



(a) Uplink



(b) Downlink

Figure 2.6 – Data flow in the conference system prototype with spatial error concealment. A participant in room 1 is speaking, two devices capture packets and forward them to the server which forwards them to the participants in room 2.

With this configuration, a primary channel carries the speaker signal and a backup channel serves in the cases that a frame loss happens in the primary channel. Two channels transport the audio signals to the mixing server and there, whenever a frame is missing from the primary microphone signal, an

estimation of it which is obtained from the backup microphone signal is put in its place. The packet flow is illustrated in Fig. 2.7. In the figure client number 2 is acting as the backup channel for the uplink signal.

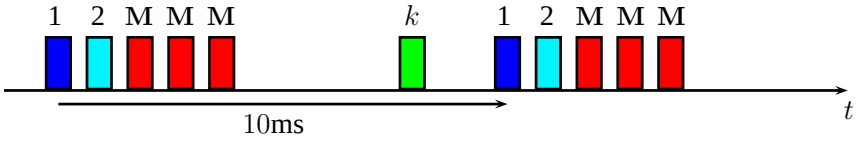


Figure 2.7 – Packet flow over time with multiple multicast downlink packets and two channels for the uplink.

For keeping two channels synchronous, the time stamping of the RTP packets described in appendix B is used. Now the question is how to estimate a lost frame of the primary signal from backup signal. Obviously in free filed the relation between two signals is a simple delay and a gain factor, whereas in reverberant environments this assumption is not true and assuming that this relation is linear the difference between two signals is the room impulse response from the backup to the primary microphone, which has infinite length and an approximation of it has to be made to estimate the relation between two signals. In the next chapter different techniques are proposed for transfer function estimation.

Chapter 3

Transfer Function Estimation

As mentioned before the goal is to conceal the errors occurring during the transmission of the signals in the wireless LAN. For this purpose we suggest that for each speaker at least two microphones capture the speech and send it to the mixing server; in this case one device acts as the primary microphone and the other one provides a backup channel. Losing a packet in the transmission is then equal to loss of an audio frame which will be compensated by a frame from a processed backup channel. Therefore the setup for capturing the speaker signal would be as in Fig. 3.1. In this figure $x_p(k)$ and $x_b(k)$ are the speaker signal reaching the primary and backup microphones respectively.

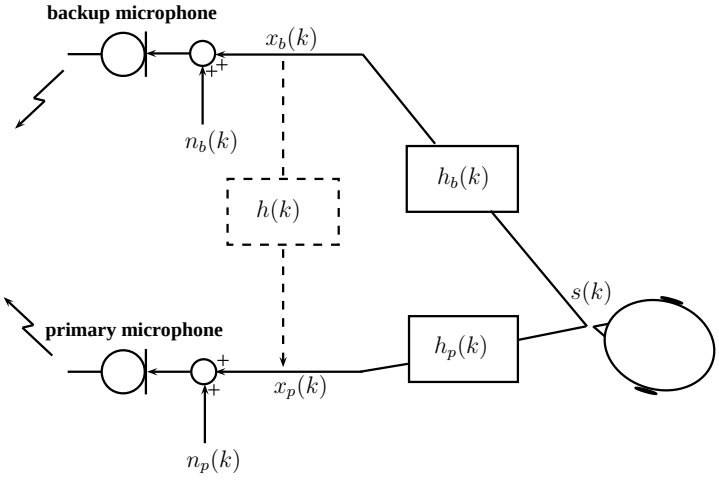


Figure 3.1 – Signal model for capturing the speaker signal by two microphones.

Note that for reconstructing some parts of $x_p(k)$ from $x_b(k)$, we need to

have an estimate of the filter $h(k)$. It should be also mentioned that with this setup, it is not expected to estimate the relation between the two signals in multiple-speaker mode, namely if we assume that two sources, $s_1(k)$ and $s_2(k)$, are active at the same time in different locations and denote their Fourier transform by $S_1(e^{j\omega})$ and $S_2(e^{j\omega})$, we have

$$\begin{aligned} X_p(e^{j\omega}) &= H_{p_1}(e^{j\omega})S_1(e^{j\omega}) + H_{p_2}(e^{j\omega})S_2(e^{j\omega}) \\ X_b(e^{j\omega}) &= H_{b_1}(e^{j\omega})S_1(e^{j\omega}) + H_{b_2}(e^{j\omega})S_2(e^{j\omega}), \end{aligned} \tag{3.1}$$

and therefore,

$$\begin{aligned} H(e^{j\omega}) = \frac{X_p(e^{j\omega})}{X_b(e^{j\omega})} &= \frac{H_{p_1}(k)S_1(e^{j\omega}) + H_{p_2}(e^{j\omega})S_2(e^{j\omega})}{H_{b_1}(e^{j\omega})S_1(e^{j\omega}) + H_{b_2}(e^{j\omega})S_2(e^{j\omega})} \\ &= \frac{H_{p_1}(e^{j\omega}) + H_{p_2}(e^{j\omega})\frac{S_2(e^{j\omega})}{S_1(e^{j\omega})}}{H_{b_1}(e^{j\omega}) + H_{b_2}(e^{j\omega})\frac{S_2(e^{j\omega})}{S_1(e^{j\omega})}}. \end{aligned} \tag{3.2}$$

As can be seen in (3.2), the relation between the channels is dependent on the speech signals, and this makes it theoretically impossible to exactly find the source signals and the relation between them; In the past two decades some related works have been done in the signal processing community on blind signal separation (BSS) [2] and independent component analysis (ICA) [3], which address a similar problem and solve it in the case that two source signals are mutually statistically independent or decorrelated. For simplicity we have assumed that only one speaker is active at a time and in this thesis source separation methods (BSS and ICA) are not addressed.

For estimating $h(k)$, the room impulse response between the two microphones, one may think of a simple delay-and-amplitude estimator and neglect the fact that in real environments room reverberation would also affect the received signals. In this case, algorithms based on cross correlation are studied. Another possible way to think of the problem can be to take into account the reverberation but still estimate only the delay and relative amplitude between the signals, which is also introduced in an algorithm in the following sections. Finally one could also estimate the whole or part of the impulse response assuming that the reverberation is present; this will be addressed here by two adaptive algorithms, which estimate adaptively the impulse response between the two microphone signals.

In this chapter, first some simple delay estimation techniques will be discussed and then some sophisticated methods will be used to estimate the relation between the primary and backup microphones more accurately.

3.1 Cross correlation based methods for time delay estimation

These methods are used under the assumption that the two microphone signals satisfy [4]:

$$x_p(t) = s(t) + n_p(t), \tag{3.3a}$$
$$x_b(t) = \alpha s(t + D) + n_b(t), \tag{3.3b}$$

where $s(t)$, $n_p(t)$ and $n_b(t)$ are real, jointly stationary random processes and moreover $s(t)$ is uncorrelated with $n_p(t)$ and $n_b(t)$.

3.1.1 Cross correlation

A common method for determining the time delay D is to compute the cross correlation function

$$R_{x_p x_b}(\tau) = \mathbb{E}\left[x_p(t)x_b(t-\tau)\right], \tag{3.4}$$

where $\mathbb{E}[\cdot]$ denotes expectation. With this notation, D can be found as

$$D = \underset{\tau}{\operatorname{argmax}}\, R_{x_p x_b}. \tag{3.5}$$

Using the model defined in (3.3) and the assumption that $n_p(t)$ and $n_b(t)$ are uncorrelated, we have

$$\begin{aligned} R_{x_p x_b}(\tau) &= \alpha R_{ss}(\tau - D) \\ &= \alpha R_{ss}(\tau) \circledast \delta(\tau - D), \end{aligned} \tag{3.6}$$

where $\circledast$ is convolution. As can be seen in (3.6) the cross correlation is actually a delta function which is smoothed by the autocorrelation of the source signal. Although the signal shape is smoothed in this case, due to the properties of autocorrelation function ($\operatorname{argmax}_\tau R_{ss}(\tau) = 0$), the maximum of the cross correlation is at $t = D$. But what happens when there is multiple delays in the signal, i.e., when there is reverberation in the room? Assuming mutual uncorrelatedness, the cross correlation becomes

$$R_{x_p x_b}(\tau) = R_{ss}(\tau) \circledast \sum_i \alpha_i \delta(\tau - D_i). \tag{3.7}$$

In this case two smoothed delta functions may spread into each other and make it impossible to detect the delays or at least the first delay. this is where some modifications to the cross correlation, such as "*generalized cross correlation*" methods, are introduced.

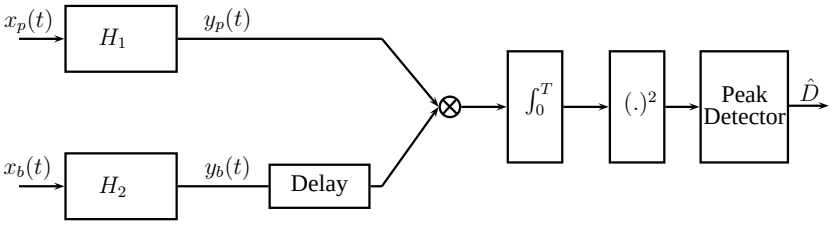


Figure 3.2 – Signal model for generalized cross correlation methods. Captured signals are filtered before computing cross correlation.

3.1.2 Generalized cross correlation

Apparently, the cross power spectrum and the cross correlation of $x_p(t)$ and $x_b(t)$ are related as

$$R_{x_p x_b}(\tau) = \int_{-\infty}^{\infty} G_{x_p x_b}(f) e^{j2\pi f\tau} df. \tag{3.8}$$

Let's assume that before maximizing the cross correlation, we filter both signals $x_p(t)$ and $x_b(t)$ as in Fig. 3.2.

The cross power spectrum of signals $y_p(t)$ and $y_b(t)$ will be

$$G_{y_p y_b}(f) = H_1(f) H_2^*(f) G_{x_p x_b}(f). \tag{3.9}$$

Calling $\Psi(f) = H_1(f)H_2^*(f)$, the cross correlation between filter outputs will be

$$R_{y_p y_b}(\tau) = \int_{-\infty}^{\infty} \Psi(f) G_{x_p x_b}(f) e^{j2\pi f\tau} df. \tag{3.10}$$

Note that the role of $\Psi(f)$ is to ideally make the peaks sharper in order to prevent them from spreading into each other. On the other hand, sharp peaks are more sensitive to errors introduced by finite observation time [4]. Different choices for $\Psi(f)$ for this tradeoff construct different varieties of generalized cross correlation methods. In the sequel we will present these alternative weighting algorithms.

3.1.2.1 The Roth weighting

Roth proposed in [5] the following weighting for $\Psi(f)$

$$\Psi(f) = \frac{1}{G_{x_b x_b}(f)}. \tag{3.11}$$

By substituting (3.11) in (3.10), one gets

$$R_{y_p y_b}(\tau) = \int_{-\infty}^{\infty} \frac{G_{x_p x_b}(f)}{G_{x_b x_b}(f)} e^{j2\pi f\tau} df. \tag{3.12}$$

Note that the values for $G_{x_p x_b}(f)$ and $G_{x_b x_b}(f)$ are computed in limited time and are approximations of the real power spectra. A close look at (3.12) reveals the relation of this formulation with optimum (Wiener) linear filter

$$H(f) = \frac{G_{x_p x_b}(f)}{G_{x_b x_b}(f)}, \tag{3.13}$$

which approximates the mapping from $x_b(t)$ to $x_p(t)$ [6].

3.1.2.2 The Smoothed Coherence Transform (SCOT)

Carter et al. in [7] used the complex coherence function $\gamma(f)$ between two signals $x_p(t)$ and $x_b(t)$, given by

$$\gamma(f) = \frac{G_{x_p x_b}(f)}{\sqrt{G_{x_p x_p}(f) G_{x_b x_b}(f)}}, \tag{3.14}$$

to formulate the modified cross correlation. SCOT is defined as the inverse Fourier transform of the weighted coherence ($\gamma(f)$):

$$C(\tau) = \int_{-\infty}^{\infty} W(f)\gamma(f) e^{j2\pi f\tau} df, \tag{3.15}$$

where $W(f)$ is a smooth weighting function. In our case $W(f) = 1$. With this definition, the weighting function with SCOT will be

$$\Psi_s(f) = \frac{1}{\sqrt{G_{x_p x_p}(f) G_{x_b x_b}(f)}}, \tag{3.16}$$

and cross correlation becomes

$$R_{y_p y_b}(f) = \int_{-\infty}^{\infty} \gamma(f) e^{j2\pi f\tau} df. \tag{3.17}$$

When $G_{x_p x_p} = G_{x_b x_b}$, the SCOT is equivalent to the Roth weghting.

Moreover, one can separate $\Psi_s(f)$ into two filters and construct the filters $H_1(f)$ and $H_2(f)$ as in Fig. 3.2 such that

$$H_1(f) = \frac{1}{\sqrt{G_{x_p x_p}(f)}} \tag{3.18}$$

$$H_2(f) = \frac{1}{\sqrt{G_{x_b x_b}(f)}}. \tag{3.19}$$

3.1.2.3 The Phase Transform (PHAT)

For the previous two methods, by rewriting the formula for $R_{y_p y_b}(\tau)$, one gets

$$R_{y_p y_b}(\tau) = \delta(\tau - D) \circledast \int_{-\infty}^{\infty} F(f) e^{j2\pi f\tau} df. \tag{3.20}$$

Therefore, the delta function is still affected by spreading. To avoid this spreading, Knapp et al. presented in [4] another weighting called the Phase Transform (PHAT),

$$\Psi_p(f) = \frac{1}{|G_{x_p x_b}(f)|}, \tag{3.21}$$

Which yields

$$R_{y_p y_b}(\tau) = \int_{-\infty}^{\infty} \frac{G_{x_p x_b}(f)}{|G_{x_p x_b}(f)|} e^{j2\pi f\tau} df. \tag{3.22}$$

If the noise signals $n_p(t)$ and $n_b(t)$ in the model (3.3) are uncorrelated, we have

$$|G_{x_p x_b}(f)| = \alpha G_{ss}(f). \tag{3.23}$$

Therefore,

$$\frac{G_{x_p x_b}(f)}{|G_{x_p x_b}(f)|} = e^{j2\pi f D}, \tag{3.24}$$

and finally

$$R_{y_p y_b}(\tau) = \delta(\tau - D). \tag{3.25}$$

Note that this only happens in the ideal case where the values for $G_{x_i x_j}$ are computed correctly, and in practice, for finite number of samples, this is not completely true. But between the methods presented above, this one seems to have better performance in narrowing the peaks.

In the implementation, in order to approximate the expectation in $G_{x_p x_b}(f)$, we use an smoothing factor, γ which computes the power spectrum as

$$G^{n+1}_{x_p x_b}(f) = \gamma \mathbf{X}_p \odot \mathbf{X}_b^* + (1 - \gamma) G^n_{x_p x_b}(f)\,, \tag{3.26}$$

where $G^n_{x_p x_b}(f)$ is the cross power spectrum for frame number n and $\odot$ represents element-wise multiplication. Also an speech activity threshold is defined to make sure that for low signal activity the cross correlation does not result in a bad estimation of the delay. The algorithm is summarized in the following;

Algorithm 1 GCC-PHAT method for TDE

Fix γ and *activityThreshold*
for n over frames **do**
    $G^n_{x_p x_b}(f) \Leftarrow \gamma \mathbf{X}_p \odot \mathbf{X}_b^* + (1-\gamma) G^{n-1}_{x_p x_b}(f)$

    $R^n_{y_p y_b}(\tau) \Leftarrow \mathbf{IFFT}\left\{ \frac{G^n_{x_p x_b}(f)}{|G^n_{x_p x_b}(f)|} \right\}$

    $D^n \Leftarrow \text{argmax}_\tau \, R^n_{y_p y_b}$
    $R_{\max} \Leftarrow \max_\tau R^n_{y_p y_b}$

    if $R_{\max} <$ *activityThreshold* **then**
        $D^n \Leftarrow D^{n-1}$
    end if
end for

There are also other techniques for weighting or prefiltering, which are not in the scope of this thesis. For more information about these techniques, the reader is referred to Eckart filter [8], and Hannan-Thomson (HT) processor [9].

As described above, cross correlation based methods do not take into account the reverberation of the room and this makes these methods less efficient for time delay estimations in the reverberant environments, which are considered in this thesis. This inability to adapt to the environment made us to look for more sophisticated, but also useful, methods for estimating the relation between $x_p(t)$ and $x_b(t)$ as modeled in Fig. 3.1.

3.2 Adaptive eigenvalue decomposition for time delay estimation

Adaptive eigenvalue decomposition was originally proposed by Benesty in [10] for passive acoustic source localization. In this thesis the method is used to estimate the time delay between the primary and backup microphone signals according to the model in Fig. 3.1.

In contrary to the previous model for cross correlation based methods, in this case the signals are modeled as

$$x_p(n) = h_p(n) \circledast s(n) + n_p(n)\,, \tag{3.27a}$$
$$x_b(n) = h_b(n) \circledast s(n) + n_b(n)\,, \tag{3.27b}$$

where $s(n)$ is the active source.

Let's first consider the noiseless case ($n_p(n) = n_b(n) = 0$). Assuming that the room impulse responses are finite with length M. It follows that

$$\mathbf{x}_p{}^{\mathrm{T}}(n)\,\mathbf{h}_b(n) = \mathbf{x}_b{}^{\mathrm{T}}(n)\,\mathbf{h}_p(n)\,, \tag{3.28}$$

where

$$\mathbf{x}_i(n) = [x_i(n) \quad x_i(n-1) \quad \cdots \quad x_i(n-M+1)]\,, \tag{3.29a}$$
$$\mathbf{h}_i(n) = [h_i(n) \quad h_i(n-1) \quad \cdots \quad h_i(n-M+1)]\,, \tag{3.29b}$$

for $i = p,\, b$.

The covariance matrix of vectors $\mathbf{x}_p(n)$ and $\mathbf{x}_b(n)$ is defined as

$$\mathbf{R} = \begin{bmatrix} \mathbf{R}_{x_p x_p} & \mathbf{R}_{x_p x_b} \\ \mathbf{R}_{x_b x_p} & \mathbf{R}_{x_p x_p} \end{bmatrix}\,, \tag{3.30}$$

where

$$\mathbf{R}_{x_i x_j} = \mathbb{E}\left[\mathbf{x}_i(n)\mathbf{x}_j{}^{\mathrm{T}}(n)\right]\,, \qquad i,j \in p,b\,. \tag{3.31}$$

If one defines the $2M \times 1$ vector

$$\mathbf{u} = \begin{bmatrix} \mathbf{h}_b \\ -\,\mathbf{h}_p \end{bmatrix}\,,$$

from (3.28) and (3.30), it is apparent that $\mathbf{R}\,\mathbf{u} = 0$. This means that the vector $\mathbf{u}$ is the eigenvector of the covariance matrix $\mathbf{R}$ corresponding to the eigenvalue 0.

Since the covariance matrix $\mathbf{R}$ is positive semi-definite, it would be enough to simply find the eigenvector corresponding to its minimum eigenvalue. This is equivalent to minimizing $\mathbf{u}^{\mathrm{T}}\,\mathbf{R}\,\mathbf{u}$ with respect to $\mathbf{u}$ and subject to $\|\,\mathbf{u}\,\|^2 = \mathbf{u}^{\mathrm{T}}\,\mathbf{u} = 1$. Therefore if one defines the error as

$$e(n) = \frac{\mathbf{u}^{\mathrm{T}}(n)\,\mathbf{x}(n)}{\|\,\mathbf{u}(n)\|}\,, \tag{3.32}$$

where $\mathbf{x}(n) = \left[x_p{}^{\mathrm{T}}(n) \quad x_b{}^{\mathrm{T}}(n)\right]^{\mathrm{T}}$, then minimizing the mean squared value of $e(n)$ solves the above problem. This is done adaptively by LMS algorithm [10], where the adaptation is given by

$$\mathbf{u}(n+1) = \frac{\mathbf{u}(n) - \mu e(n) x(n)}{\|\,\mathbf{u}(n) - \mu e(n) x(n)\|}\,. \tag{3.33}$$

It is very important to note that in this method the goal is not to estimate (either accurately or approximately) the impulse responses, but rather the

time delay between two signals. So the adaptation starts with a vector $\mathbf{u}$ with only one non-zero element in the middle of its first half. The procedure is depicted below:

Algorithm 2 Adaptive Eigenvalue Decomposition for TDE

1: $\mathbf{h}_b \Leftarrow [0,\ 0,\ \cdots,\ 0,\ 1,\ 0,\ \cdots,\ 0,\ 0]$: one at position $M/2$
2: $\mathbf{h}_p \Leftarrow [0,\ 0,\ \cdots,\ 0,\ 0,\ 0,\ \cdots,\ 0,\ 0]$
3: $\mathbf{u} \Leftarrow \begin{bmatrix} \mathbf{h}_b{}^{\mathrm{T}}(n) & -\mathbf{h}_p{}^{\mathrm{T}}(n) \end{bmatrix}^{\mathrm{T}}$
4: Adjust adaptation parameters so that peak at position $M/2$ is always dominant.
5: **for** n over frames **do**
6: $\quad e(n) \Leftarrow \mathbf{u}^{\mathrm{T}}(n)\,\mathbf{x}(n)$
7:
8: $\quad \mathbf{u}(n+1) \leftarrow \frac{\mathbf{u}(n)-\mu e(n)x(n)}{\|\,\mathbf{u}(n)-\mu e(n)x(n)\|}$
9:
10: $\quad D \Leftarrow \operatorname{argmin} \mathbf{u}[M+1, \cdots, 2M] - M/2$
11: **end for**

Doblinger in [11] argues that a significantly greater computational efficiency can be achieved by using an FFT-based algorithm requiring only four FFTs per frame. In this algorithm, the normalization of vector $\mathbf{u}$ is eliminated. Actually this normalization, as argued in [10], may avoid an error propagation if the algorithm runs for a long period of time, thus in the FFT-based algorithm from [11], in order to ensure tracking, the adaptive algorithm is periodically restarted in order to eliminate the error propagation over time.

This FFT-based algorithm from [11] is depicted in the following structure ($M = 2L$ is the frame size and adaptive filter length is L):

Algorithm 3 Adaptive Eigenvalue Decomposition for TDE in Fourier domain

$\mathbf{h}_b \Leftarrow [0,\ 0,\ \cdots,\ 0,\ 1,\ 0,\ \cdots,\ 0,\ 0]$: one at position $M/2$
$\mathbf{h}_p \Leftarrow [0,\ 0,\ \cdots,\ 0,\ 0,\ 0,\ \cdots,\ 0,\ 0]$
$FrameCounter \Leftarrow 0$
Fix $ResetCounter$ and μ
loop
    $FrameCounter \Leftarrow FrameCounter + 1$

    $X_b(m,k) \Leftarrow \sum_{n=0}^{M-1} x_b(mL+n)e^{-j\frac{2\pi}{M}nk}$
    $X_p(m,k) \Leftarrow \sum_{n=0}^{M-1} x_p(mL+n)e^{-j\frac{2\pi}{M}nk}$

    $e(m,\ n) \Leftarrow \frac{1}{M}\sum_{k=0}^{M-1}\left[H_b(m,k)X_b(m,k) + H_p(m,k)X_p(m,k)\right]e^{j\frac{2\pi}{M}nk}$
    $E(m,k) \Leftarrow \sum_{n=0}^{M-1} e(m,n)e^{-j\frac{2\pi}{M}nk}$

    $S_{x_bx_b}(m,k) \Leftarrow \gamma|X_b(m,k)|^2 + (1-\gamma)S_{x_bx_b}(m-1,k)$
    $S_{x_px_p}(m,k) \Leftarrow \gamma|X_p(m,k)|^2 + (1-\gamma)S_{x_px_p}(m-1,k)$

    $H_b(m+1,k) \Leftarrow H_b(m,k) - \mu\frac{X_b^*(m,k)E(m,k)}{S_{x_bx_b}(m,k)+\epsilon}$
    $H_p(m+1,k) \Leftarrow H_p(m,k) - \mu\frac{X_p^*(m,k)E(m,k)}{S_{x_px_p}(m,k)+\epsilon}$

    if mod ($FrameCounter$, $ResetCounter$) $= 0$ **then**
        reset everything
        $h_p(m,n) \Leftarrow \frac{1}{M}\sum_{k=0}^{M-1} H_p(m,k)e^{j\frac{2\pi}{M}nk}$
        $D \Leftarrow \text{argmin}_n\, h_p(m,n)$ - M/2
    end if
end loop

3.3 Adaptive variable step-size transfer function estimation

All methods that we have studied by now estimate the time delay between the two signals and none of them is concerned about estimating at least an approximation of the impulse response between two microphones. Since the backup signal is often farther from the speaker than the primary one, the level of room reverberation for the desired source is higher compared to the primary microphone signal. Thus the output of these algorithms suffers from the fact that extra room reverberation is non-realistically added

to the primary signal. Moreover, estimating the delay is not enough for replacing a signal with the other one, but the relative level difference is also of significant importance. Even with accurate delay estimate between two signals, because of reverberation in the room and small frame sizes, it is not possible to calculate an exact gain factor for the transform and this with the above effects cause some artifacts in the output of these algorithms. This justifies the reason to study ways for estimating the impulse response for replacing the missing packets in the primary signal according to the backup signal.

An adaptive algorithm for estimating impulse responses is originally proposed by Myllylä [12] for enhancing acoustic echo control. The algorithm uses a residual echo filter to reduce the amount of the residual echo after the main adaptive filter. The method needed spme modifications to be applied for transfer function estimation between the backup and primary signals. In the echo cancellation schemes, the echo power attenuates from the loudspeaker to the microphone, however in our problem, the backup signal is weaker in power compared to the primary signal. Furthermore, since usually the backup microphone is farther away from the speaker than the primary one, the estimated filter from the backup microphone to the primary microphone is not causal. We will assume that the devices are placed in such a way that the maximum direct delay between the signals do not exceed size of the audio processing buffer (which is either 10 ms or 20 ms corresponding to 3.4 m and 6.8 m). To compensate these two differences, we intentionally apply a delay to the primary signal and a gain factor to the backup signal before they are processed . For the simplicity in the notations we will still call the resulted signals $x_p(k)$ and $x_b(k)$.

For the rest of the section, we will use the following notation for the signals and filters;

- $\mathbf{h}(k)$ – The room impulse response from the backup to the primary signal.
- $\hat{\mathbf{h}}(k)$ – Estimation of the room impulse response from the backup to the primary signal.
- $\mathbf{\Delta}(k) = \mathbf{h}(k) - \hat{\mathbf{h}}(k)$ – The system mismatch vector.
- $x_b(k)$ – Captured signal by the backup microphone.
- $x_p(k)$ – Captured signal by the primary microphone.
- $n_p(k)$ – Modeled interference close to the primary microphone.

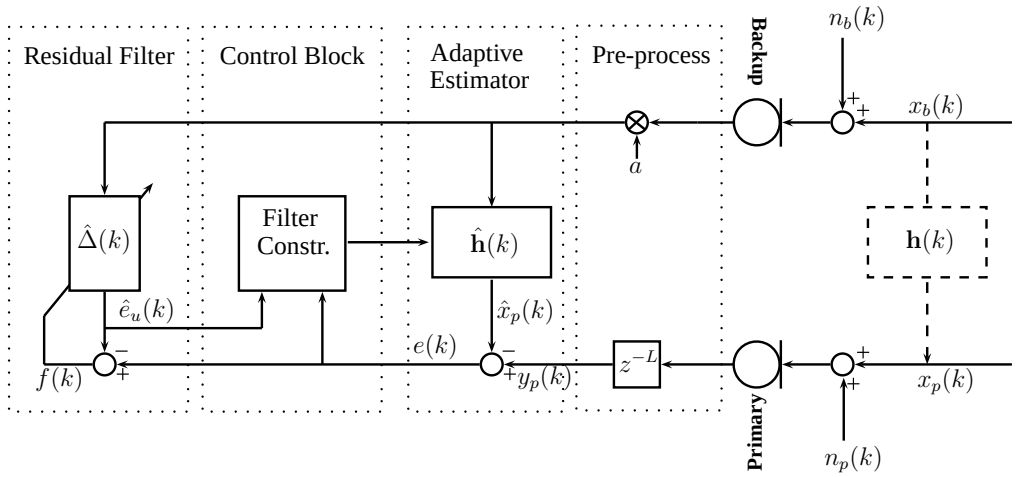


Figure 3.3 – Variable step size adaptive filter for transfer function estimation.

- $n_b(k)$ – Modeled interference close to the backup microphone.
- $\hat{x}_p(k) = \hat{\mathbf{h}}(k)\,\mathbf{x}(k)$ – signal estimating $x_p(k)$.
- $e(k) = y_p(k) - \hat{x}_p(k)$ – Adaptive filter error.
- $e_u(k) = x_p(k) - \hat{x}_p(k)$ – Undistorted error.

We will also use the subscripts m for the main filter and r for the residual filter.

The signal model for this specific adaptive filter is illustrated in Fig. 3.3. The main idea in this work is to use an additional adaptive filter called the residual filter, placed after the main adaptive filter for estimating $e_u(k)$ through the system coupling factor defined as

$$\beta_m(k) = \mathbb{E}\{\| \mathbf{h}(k) - \hat{\mathbf{h}}(k)\|^2\} = \mathbb{E}\{\| \mathbf{\Delta}(k)\|^2\} \tag{3.34}$$

Note that since we have interference combined in $x_p(k)$, it is not possible to have an exact value for $e_u(k)$ and that's why we are trying to estimate it. The system coupling factor can be interpreted as an energy transfer factor that describes what portion of $x_b(k)$ is still present in $e(k)$ after the main adaptive filter.

3.3.1 Main filter control

For the main filter control the common Normalized Least Mean Square (NLMS) filter [13] is used as follows

$$e(k) = y_p(k) - \hat{\mathbf{h}}^{\mathrm{T}}(k)\,\mathbf{x}_b(k)\,, \tag{3.35a}$$

$$\hat{\mathbf{h}}(k+1) = \hat{\mathbf{h}}(k) + \mu_m(k)e(k)\frac{\mathbf{x}_b(k)}{\|\,\mathbf{x}_b(k)\|^2}\,, \tag{3.35b}$$

with $0 \leq \mu_m \leq 1$ being the step size. This method also benefits from variable step size, in a way that if there is some local activity close to the primary microphone, which is not present in the backup microphone (here modeled as $n_p(k)$), step size should be reduced and if the position of one of the microphones changes (and consequently $\mathbf{h}(k)$), the step size should get larger to allow following the changes fast enough. This procedure is done in [14] as follows:

Assuming that the system is time invariant, namely

$$\mathbf{h}(k+1) = \mathbf{h}(k)\,, \tag{3.36}$$

then (3.35b) can be written as

$$\mathbf{\Delta}(k+1) = \mathbf{\Delta}(k) - \mu_m(k)\frac{e(k)\,\mathbf{x}_b(k)}{\|\,\mathbf{x}_b(k)\|^2}\,. \tag{3.37}$$

The aim of the adaptive filter is to minimize the expected square norm of the system mismatch vector $\mathbf{\Delta}(k)$. Using (3.37) and the fact that $e(k) = e_u(k) + n_p(k) = \mathbf{\Delta}^{\mathrm{T}}(k)\,\mathbf{x}_b(k) + n_p(k)$, one gets

$$\begin{aligned}\mathbb{E}\left\{\|\,\mathbf{\Delta}(k+1)\|^2\right\} = \mathbb{E}\left\{\|\,\mathbf{\Delta}(k)\|^2\right\} - 2\mu_m(k)\,\mathbb{E}\left\{\frac{e(k)e_u(k)}{\|\,\mathbf{x}_b(k)\|^2}\right\} + \\ \mu_m^2(k)\,\mathbb{E}\left\{\frac{e^2(k)}{\|\,\mathbf{x}_b(k)\|^2}\right\}\,.\end{aligned} \tag{3.38}$$

This quantity is minimized for

$$\frac{\partial\,\mathbb{E}\left\{\|\,\mathbf{\Delta}(k+1)\|^2\right\}}{\partial\mu_m(k)} = 0\,, \tag{3.39}$$

which results in

$$\mu_{m,\mathrm{opt}}(k) = \frac{\mathbb{E}\left\{\frac{e(k)e_u(k)}{\|\,\mathbf{x}_b(k)\|^2}\right\}}{\mathbb{E}\left\{\frac{e^2(k)}{\|\,\mathbf{x}_b(k)\|^2}\right\}} \approx \frac{\mathbb{E}\{e_u^2(k)\}}{\mathbb{E}\{e^2(k)\}}\,. \tag{3.40}$$

In the above equation it is assumed that the norm of the backup vector, $\| \mathbf{x}_b(k)\|^2$ can be approximated by a constant and signals $n_p(k)$ and $x_b(k)$ are uncorrelated. By applying another approximation for the expectation, the step size becomes

$$\mu_{m,\text{opt}} = \frac{\overline{e_u^2(k)}}{\overline{e^2(k)}}\,, \tag{3.41}$$

where the overline represents the short term smoothing function.

$$\overline{e^2(k)} = \gamma e^2(k) + (1-\gamma)\overline{e^2(k-1)} \quad 0 < \gamma \leq 1\,. \tag{3.42}$$

Because of all the assumptions and simplifications done in deriving the formula, $\mu_{m,\text{opt}}$ is called the pseudo-optimal step size. However, the problem with this step size is that the value of $e_u(k)$ is not directly available. Therefore, the following is used instead as the step size in this algorithm

$$\mu_m = \frac{\mathbb{E}\left\{e_u^2(k)\right\}}{\mathbb{E}\left\{e^2(k)\right\}} = \frac{\mathbb{E}\left\{\left[\boldsymbol{\Delta}^{\mathrm{T}}(k)\,\mathbf{x}_b(k)\right]^2\right\}}{\mathbb{E}\left\{e^2(k)\right\}} \approx \frac{\beta_m(k)\overline{x_b^2(k)}}{\overline{e^2(k)}}\,, \tag{3.43}$$

where it is assumed that $x_b(k)$ and the system mismatch vector are independent and the backup signal $x_b(k)$ is white.

Now it remains to find the update formula for the coupling factor, $\beta_m(k)$. For the coupling factor as defined in (3.34) there is no direct access to $\mathbf{h}(k)$. To compute this quantity, when the system is time invariant as in (3.36) the coupling factor can be approximated as [14]

$$\beta_m(k+1) \approx \left(1 - \frac{\mu_m(k)(2-\mu_m(k))}{M}\right)\beta_m(k) + \frac{\mu_m^2(k)}{M}\frac{\sigma_{n_p}^2}{\sigma_{x_b}^2} \tag{3.44a}$$

$$\approx \left(1 - \frac{\mu_m(k)(2-\mu_m(k))}{M}\right)\beta_m(k) + \frac{\mu_m^2(k)}{M}\frac{1}{\text{esnr}}\,, \tag{3.44b}$$

where M is the filter length and, since we do not have the real SNR between $x_b(k)$ and $n_p(k)$, it is replaced by a fixed value called the expected SNR (esnr).

3.3.2 Residual filter control

For updating the residual filter an NLMS filter is used as well. Following the same procedure for the main filter, we have

$$f(k) = e(k) - \hat{\boldsymbol{\Delta}}^{\mathrm{T}}(k)\,\mathbf{x}_b(k) \tag{3.45a}$$

$$\hat{\boldsymbol{\Delta}}(k+1) = \hat{\boldsymbol{\Delta}}(k) + \mu_r(k) f(k) \frac{\mathbf{x}_b(k)}{\| \mathbf{x}_b(k)\|^2}\,. \tag{3.45b}$$

And similar to the main filter

$$\mu_r(k) = \frac{\beta_r(k)\overline{x_b^2(k)}}{\overline{f^2(k)}}\,. \tag{3.46}$$

The coupling factor for the residual filter will be

$$\begin{aligned}\beta_r(k) &= \mathbb{E}\left\{\| \boldsymbol{\Delta}(k) - \hat{\boldsymbol{\Delta}}(k)\|^2\right\} \\ &= \mathbb{E}\left\{\| \boldsymbol{\Delta}(k)\|^2\right\} - 2\,\mathbb{E}\left\{\boldsymbol{\Delta}^{\mathrm{T}}(k)\,\hat{\boldsymbol{\Delta}}(k)\right\} + \mathbb{E}\left\{\| \hat{\boldsymbol{\Delta}}(k)\|^2\right\}\,.\end{aligned} \tag{3.47}$$

The second term in (3.47) is small enough to be neglected, thus the coupling factor for the residual filter becomes

$$\beta_r(k) \approx \beta_m(k) + \overline{\| \hat{\boldsymbol{\Delta}}(k)\|^2}\,. \tag{3.48}$$

If the path between the primary and the backup microphones is stable for some time, $\beta_m(k)$ becomes small as the main filter adapts and the same happens to $\overline{\| \hat{\boldsymbol{\Delta}}(k)\|^2}$. As a consequence, $\beta_r(k)$ becomes and stays small. In this case, if the path between two microphones changes, the frozen system cannot adapt to the new path, unless there is a mechanism to restart the adaptation procedure. For this purpose, Myllylä in [12] uses the residual filter as

$$\beta_m(k+1) = \begin{cases}\overline{\| \hat{\boldsymbol{\Delta}}(k+1)\|^2}\,, & \overline{e^2(k)} - \overline{f^2(k)} > \mathrm{dth} \\ \beta_m(k+1)\,, & \text{otherwise}\end{cases} \tag{3.49}$$

where dth is the detection threshold for the path between two microphones and is fixed in the algorithm to some constant.

To summarize this section we present the entire process in the following algorithm.

Algorithm 4 Adaptive variable step-size algorithm for transfer function estimation

Fix esnr and dth
$\hat{\mathbf{h}} \Leftarrow \mathbf{0}$
$\hat{\mathbf{\Delta}} \Leftarrow \mathbf{0}$
$\beta_m(0) \Leftarrow 0$
$\beta_r(0) \Leftarrow 0$
for k **do**
    $e(k) \Leftarrow y_p(k) - \hat{\mathbf{h}}^{\mathrm{T}}(k)\,\mathbf{x}_b(k)$
    $f(k) \Leftarrow e(k) - \hat{\mathbf{\Delta}}^{\mathrm{T}}(k)\,\mathbf{x}_b(k)$
    $\mu_m(k) \Leftarrow \frac{\beta_m(k)\overline{x_b^2(k)}}{\overline{e^2(k)}}$
    $\mu_r(k) \Leftarrow \frac{\beta_r(k)\overline{x_b^2(k)}}{\overline{f^2(k)}}$
    $\hat{\mathbf{h}}(k+1) \Leftarrow \hat{\mathbf{h}}(k) + \mu_m(k)e(k)\frac{\mathbf{x}_b(k)}{\|\mathbf{x}_b(k)\|^2}$
    $\hat{\mathbf{\Delta}}(k+1) \Leftarrow \hat{\mathbf{\Delta}}(k) + \mu_r(k)f(k)\frac{\mathbf{x}_b(k)}{\|\mathbf{x}_b(k)\|^2}$
    $\beta_m(k+1) \Leftarrow \left(1 - \frac{\mu_m(k)(2-\mu_m(k))}{M}\right)\beta_m(k) + \frac{\mu_m^2(k)}{M}\frac{1}{\mathrm{esnr}}$
    $\beta_r(k) \Leftarrow \beta_m(k) + \overline{\|\hat{\mathbf{\Delta}}(k)\|^2}$
    $\beta_m(k+1) \Leftarrow \begin{cases} \overline{\|\hat{\mathbf{\Delta}}(k+1)\|^2}\,, & \overline{e^2(k)} - \overline{f^2(k)} > \mathrm{dth} \\ \beta_m(k+1)\,, & \text{otherwise} \end{cases}$
end for

One implementation issue should be noted here that the algorithm assumes that the microphone signals, $x_b(k)$ and $x_p(k)$, have flat spectra. To achieve this property, fixed first-order whitening is used before the algorithm starts:

$$x_{p,w}(k) = \mathbf{g} \circledast x_p(k)\,, \tag{3.50a}$$
$$x_{b,w}(k) = \mathbf{g} \circledast x_b(k)\,, \tag{3.50b}$$

where $\mathbf{g} = [1,\, \alpha]$ is the whitening filter.

As seen in this section, this algorithm implements two parallel adaptive filters which are both benefitting from a variable step size close to the pseudo-optimal one in (3.41). In the calculations, it is assumed that the filter is time invariant, but [12] solves the problem of the path change between the microphones by a simple check based on the residual filter energy.

3.4 Perceptually motivated transfer function estimation

In this section, we present a rather simple method that is based on the perceptual measures of the human auditorial system. This method uses time-frequency analysis, and the two signals are processed as

$$X_p(m,k) = \sum_{n=0}^{M-1} w(n)x_p(mL+n)e^{-j\frac{2\pi}{M}nk}, \quad k = 0, 1, \cdots, M-1, \tag{3.51a}$$

$$X_b(m,k) = \sum_{n=0}^{M-1} w(n)x_b(mL+n)e^{-j\frac{2\pi}{M}nk}, \quad k = 0, 1, \cdots, M-1, \tag{3.51b}$$

where $w(n)$ is a window with length M ($M = 4L$):

$$w(n) = \begin{cases} 0 & n \in [0, L-1] \\ \sin^2(\frac{\pi(n-L)}{2L}) & n \in [L, 3L-1] \\ 0 & n \in [3L, 4L-1] \end{cases} \tag{3.52}$$

The window shape is illustrated in Fig. 3.4.

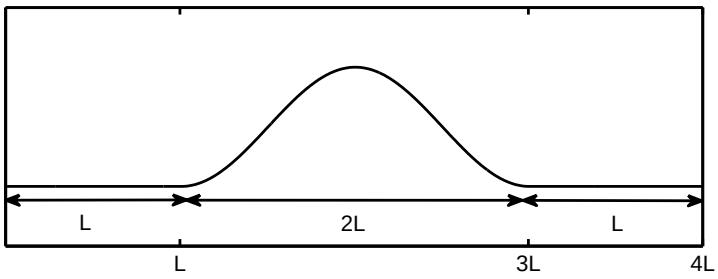


Figure 3.4 – The window used for time-frequency analysis in perceptually motivated transfer function estimation method.

This scheme adapts to the signal statistics in time and frequency and the choice for the time-frequency representation is the use of the critical bands as described in [15]. We will use ERB (Equivalent Rectangular Bandwidth) scale to model the critical bandwidth of the auditory system in each frequency. At moderate sound levels, the ERB in Hz is defined by [16]

$$\text{ERB}(f) = 0.108f + 24.7, \tag{3.53}$$

where f is the center frequency in Hz. But with this definition, finding the corresponding frequency bands is not easy, so in [16] the ERB scale is defined as the number of ERBs below each frequency:

$$\mathrm{ERBS}(f) = 21.4 \log_{10}(0.00437f + 1). \tag{3.54}$$

By using the inverse of the above formula, the boundary frequencies are determined for the frequency bands used in the algorithm. Moreover, the signals are assumed to be stationary in each time-frequency tile. We assume that the primary microphone signal $X_p(m,k)$ can be written as

$$X_p(m,k) = H(m,k)X_b(m,k) + N_p(m,k), \tag{3.55}$$

where $H(m,k)$ is a time and frequency dependent gain factor. This model is motivated in [17]. It is also assumed that all the signals are zero mean and $X_p(m,k)$ is independent from $N_p(m,k)$. The goal of the method is to estimate the relation between the primary microphone signal, $X_p(m,k)$, and the backup microphone signal, $X_b(m,k)$.

By multiplying both sides of (3.55) and taking expectation, one obtains

$$\mathbb{E}\left[X_p(m,k)X_b^*(m,k)\right] = H(m,k)\,\mathbb{E}\left[|X_b(m,k)|^2\right]. \tag{3.56}$$

It follows that

$$H(m,k) = \frac{\mathbb{E}\left[X_p(m,k)X_b^*(m,k)\right]}{\mathbb{E}\left[|X_b(m,k)|^2\right]}, \tag{3.57}$$

which is the Wiener filter solution of the above problem. We will also simplify the expectation by short time averaging operation for estimating the mean in each time-frequency band. In particular,

$$\begin{aligned}\mathbb{E}\left[X_p(m,k)X_b^*(m,k)\right] &= \gamma\left(X_p(m,k)X_b^*(m,k)\right) \\ &\quad + (1-\gamma)\,\mathbb{E}\left[X_p(m-1,k)X_b^*(m-1,k)\right], \qquad (3.58a)\\ \mathbb{E}\left[|X_b(m,k)|^2\right] &= \gamma\left(|X_b(m,k)|^2\right) + (1-\gamma)\,\mathbb{E}\left[|X_b(m-1,k)|^2\right]. \qquad (3.58b)\end{aligned}$$

where $0 < \gamma \leq 1$.

To summarize, the algorithm is depicted below:

Algorithm 5 Perceptually motivated transfer function estimation

Fix γ
for m **do**
    $X_p(m,k) \Leftarrow \sum_{n=0}^{M-1} w(n)x_p(mL+n)e^{-j\frac{2\pi}{M}nk}$
    $X_b(m,k) \Leftarrow \sum_{n=0}^{M-1} w(n)x_b(mL+n)e^{-j\frac{2\pi}{M}nk}$

    $\mathbb{E}\left[X_p(m,k)X_b^*(m,k)\right] \Leftarrow \gamma\left(X_p(m,k)X_b^*(m,k)\right) + (1-\gamma)\,\mathbb{E}\left[X_p(m-1,k)X_b^*(m-1,k)\right]$
    $\mathbb{E}\left[|X_b(m,k)|^2\right] \Leftarrow \gamma\left(|X_b(m,k)|^2\right) + (1-\gamma)\,\mathbb{E}\left[|X_b(m-1,k)|^2\right]$

    $H(m,k) = \frac{\mathbb{E}\left[X_p(m,k)X_b^*(m,k)\right]}{\mathbb{E}[|X_b(m,k)|^2]}$
end for

As seen above, this method is simple and the time-frequency resolution is chosen according to the perceptual criteria. The only parameter that should be fixed for this method is γ, which determines the effect of the previous frames in estimation of the current impulse response. If γ is chosen to be small, one has a rather smoothly changing impulse response over time frames, but is not able to follow the changes very fast. On the other hand, large γ incorporates less from previous frames and the estimated transfer function changes very fast from frame to frame (especially when having interference in $x_p(n)$). Therefore, choosing γ should be done carefully to have a good compromise between these two cases.

3.5 Discussion

In the previous sections, we presented the mathematical basis for some estimation methods. Here we will have a quick overview on the implementation issues and some estimation experiments will be presented.

As mentioned before, our estimation methods are divided into two categories, methods which only estimate the direct delay, and methods which estimate the transfer functions. For the time delay estimation methods, the relative estimated delay is compared to the original one, whereas in the transfer function estimation methods, the distance between the estimated transfer function and the original is calculated.

Two speech signals, shown in Fig. 3.5, are used in the experiments. They are 10 seconds long and have sampling rate of 16kHz. Each of these signals is first passed through filter h_b to construct the backup signal, x_b. h_p is selected to be the convolution of a known filter h with h_b. In this case the estimation algorithms should give an estimation, $\hat{h}$, of the known filter h.

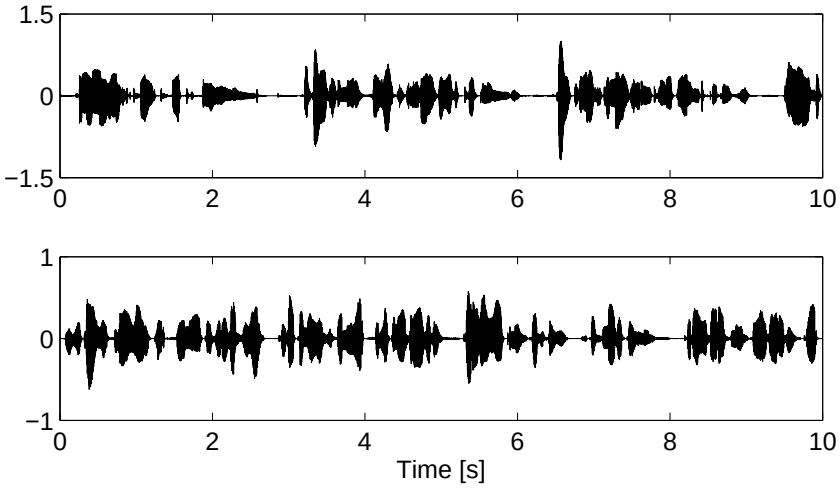


Figure 3.5 – Two signals used for the empirical evaluation of transfer function estimation algorithms.

Furthermore, we use three different filters, h_1, h_2 and h_3 within the duration of each signal (or in other words the position of primary microphone is changing, and so the transfer function h). Therefore, the signals are divided to three equal parts and each part is convolved with its corresponding filter. Filters h_b, h_1, h_2 and h_3 are presented in Fig. 3.6.

The direct acoustic delay for the mentioned filters are:

filter	direct delay [ms]
h_1	4.6250
h_2	3.3125
h_3	6.0625

The primary signal and the parts corresponding to h_1, h_2 and h_3 are shown in Fig. 3.7.

3.5.1 GCC-PHAT

For implementing this method, we used two fixed parameters; one is the smoothing factor for the approximation of expectation, and the other is the activity threshold. These two are the parameters which determine the accuracy of the method for different signals. Apparently if the smoothing factor γ is small, the method cannot follow the changes fast enough and if it is large, the fluctuations in the estimated delays will be high. In Fig. 3.8 two

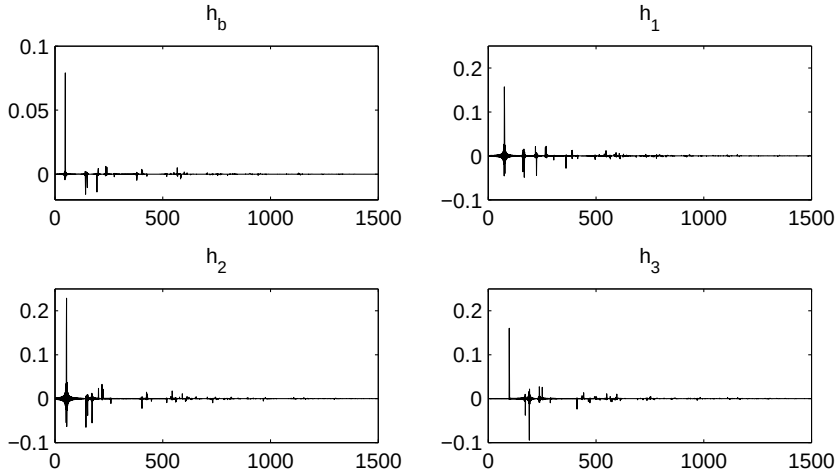


Figure 3.6 – Filters used for the experiments on the proposed algorithms. h_b generates the backup signal and h_1, h_2 and h_3 convolved with h_b generate different parts of the primary signal.

experiments with this method are presented, and the expected results are confirmed. For these experiments, the frame length is $10ms$ (which corresponds to 160 samples) and the history length (the buffer size for determining GCC over the past received samples) is taken to be 5 times the frame length.

3.5.2 Adaptive eigenvalue decomposition

For this method, the FFT based implementation is used. Similar to GCC-PHAT method for the estimating the expectation, a smoothing factor γ is

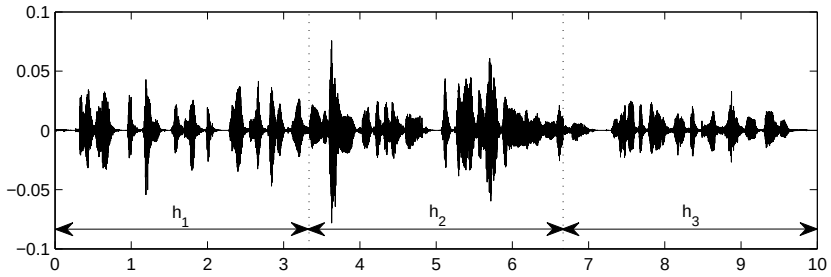
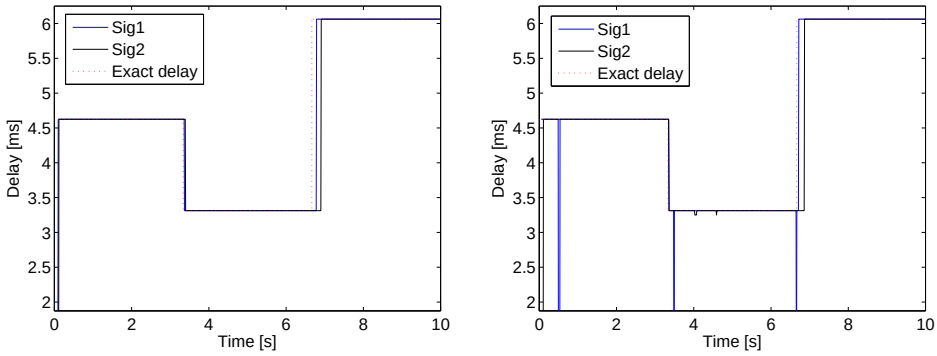


Figure 3.7 – Primary signal, generated by three different filters h_1, h_2 and h_3.



(a) $\gamma = 0.2$ (b) $\gamma = 0.9$

Figure 3.8 – GCC-PHAT method: estimated delays for two different values of γ. As the figure shows, the estimated delay has more fluctuations for large γ and small value of γ results in late adaptation to the changes.

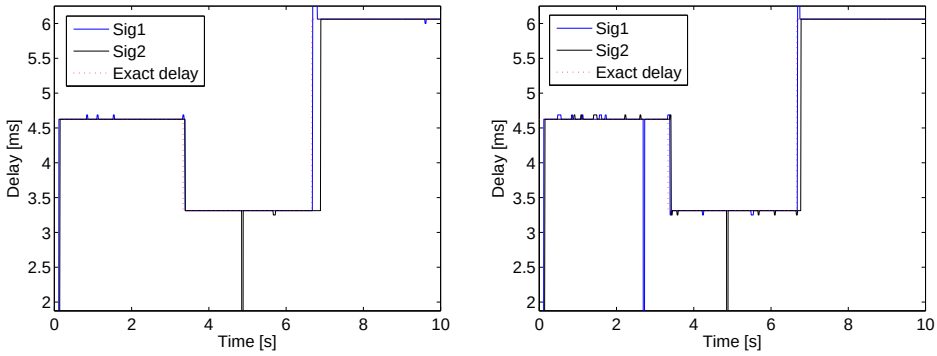
introduced. The same argument as for GCC-PHAT method applies here. If the value for γ is large, more fluctuations are seen in the results, whereas if it is small, late path changes are observed in the algorithm outputs. The results for two different signals and two different values of γ is presented in Fig. 3.9. In these experiments, the used frame size is 160 samples and the estimated filter size is 5 times the frame size.

3.5.3 Adaptive variable step-size

For evaluating this method, we introduce a measure as the accuracy of the results. It is defined as the distance between the real transfer function and the estimated one. It should be mentioned that the real transfer function had 3052 taps, whereas here we are estimating only 5 times the frame size (800) of them. So to compare the two, we calculate their Fourier transforms with the larger number of samples and find the system distance in the Fourier domain:

$$\Delta = \frac{\left\| \mathbf{DFT}(\hat{\mathbf{h}}) - \mathbf{DFT}(\mathbf{h}) \right\|}{\|\mathbf{DFT}(\mathbf{h})\|}, \tag{3.59}$$

In Fig. 3.10 the outputs of the method for two different signals are presented. As expected, the algorithm behaves very well with the path changes and the experiment shows that he adaptation is signal-dependent.



(a) $\gamma = 0.2$ (b) $\gamma = 0.8$

Figure 3.9 – Adaptive eigenvalue decomposition method: estimated delays for two different values of γ. As the figure shows, the estimated delay has more fluctuations for large γ and small value of γ results in late adaptation to the changes.

3.5.4 Perceptually motivated method

Since the signals are first windowed before estimating the transfer function in this method, it is not possible to compare the estimated transfer function with the actual one. Therefore, for evaluating this method, for each frame, the next frame of the primary signal is estimated based on the estimated transfer function for the current frame and the error for each frame is calculated as

$$error(n) = \frac{\left\| \left\| \mathbf{X}_p^{(n+1)} - \hat{\mathbf{X}}_p^{(n+1)} \right\| \right\|}{\| \mathbf{X}_p^{(n+1)} \|}, \tag{3.60}$$

where $\mathbf{X}_p^{(n+1)}$ represents the Fourier transform of frame $n+1$. Algorithm output is presented in Fig. 3.11. As expected, the method adaptively reduces the amount of error, even when the transfer functions are changed.

In the next chapter we will use these methods for estimating the lost frames in the primary signal based on the backup signal.

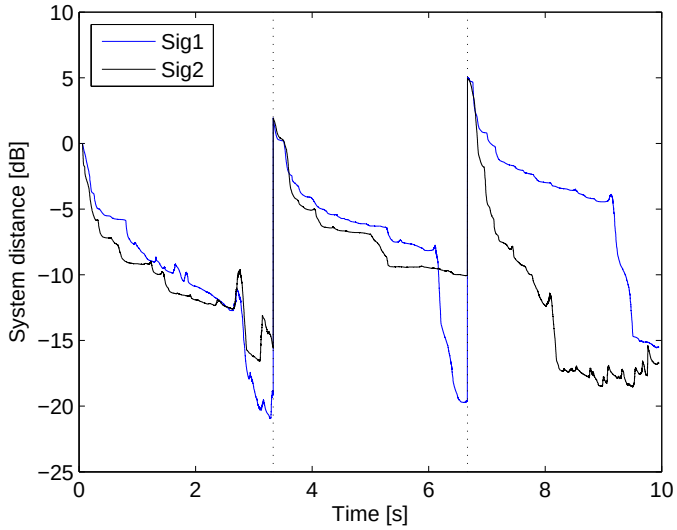


Figure 3.10 – Adaptive variable step-size method: system distance for two different signals. The time instances where the transfer functions are changed are shown with dashed lines.

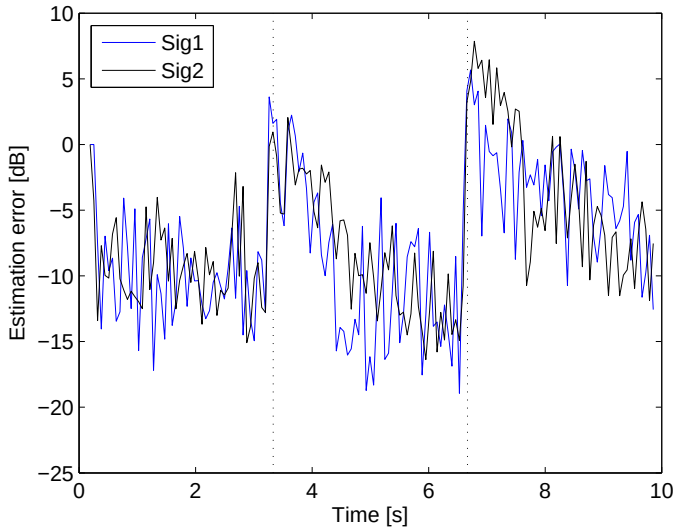


Figure 3.11 – Perceptually motivated method: estimation error for two different signals. The time instances where the transfer functions are changed are shown with dashed lines. Estimation error is computed by (3.60).

Chapter 4

Spatial Error Concealment

In Chapter 2, we showed that in the uplink from the user devices towards the mixing server, some audio packets may be lost and we presented the idea to use an extra acoustic channel as backup for estimating the missing frames in the primary signal.

To sum up the structure of the problem, a block diagram as in Fig. 4.1 would help. As the figure implies, the acoustic signal is sent through two separate channels which are assumed to be wireless. The packets are received by the server. While extracting the audio frames from RTP packets, the server checks if a frame is lost or not. If the frame is not lost (which happens in most of the cases), it is forwarded to the listening clients and the next frame is requested from the buffer, but if a frame is lost, an estimation of the relation between two channels is requested and using this estimation and the corresponding frames of the backup signal, the primary frame is reconstructed and sent to the downlink.

In this procedure, since the reconstructed frame introduces a discontinuity in the primary signal, a cross fading is applied during the replacement. This is illustrated in Fig. 4.2. The window $w(n)$ has length $2L$, where L is the frame length and obeys the following equation:

$$w(n) = \begin{cases} 1 & 0 \leq |n| \leq \frac{L}{2} \\ \frac{1}{2}\left(1 + \cos\left(\pi \frac{|n| - \frac{L}{2}}{L}\right)\right) & \frac{L}{2} \leq |n| \leq L \end{cases} \tag{4.1}$$

In real time implementations, $w(n)$ can be approximated with a linear cross-fading window.

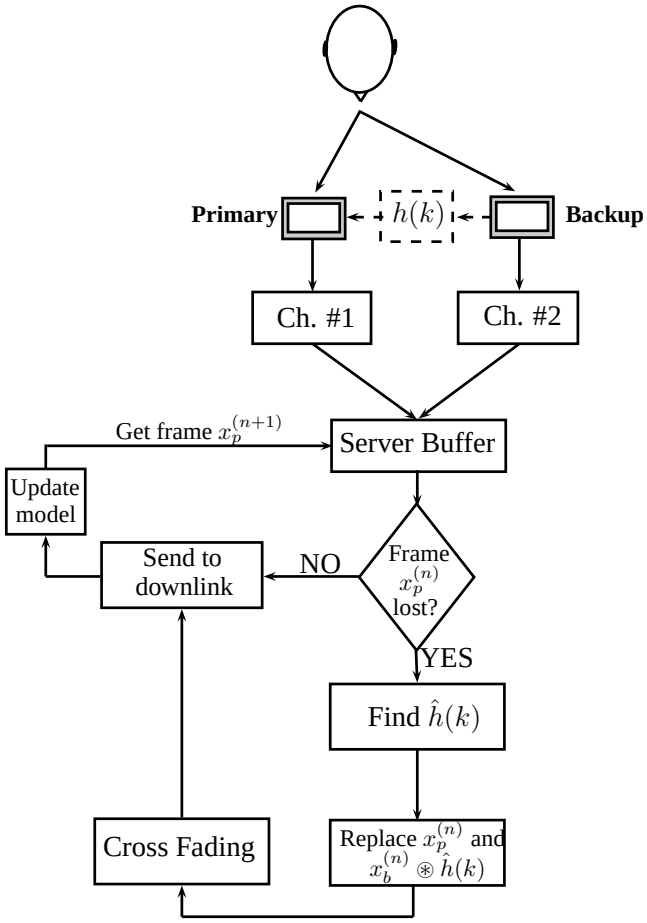


Figure 4.1 – Block diagram for functionality of acoustic error concealment.

4.1 Evaluation

Two sets of signals are used for evaluating the methods for acoustic error concealment in the audio conferencing system. The materials for each of these sets are presented below:

- **SET 1 (Simulated data):** This set consists of four studio recorded speech signals, each 6 seconds long, sampled at 16kHz and quantized to 16 bits. Two of the speakers are female and two are male. These signals are shown in Fig. 4.3. Each signal is divided to frames of each $10ms$ long (160 samples).

  Artificial primary and backup signals are generated using these signals as follows:

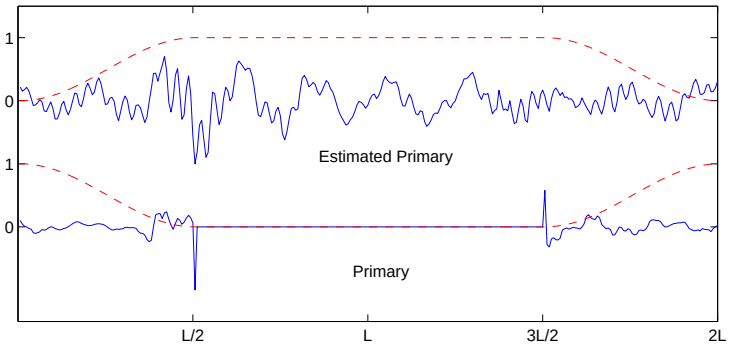


Figure 4.2 – Cross fading during the replacement of a missing frame.

It is assumed that speaker, the primary, and the backup devices are all in a room with the following characteristics:

- Room dimension: $7m \times 6m \times 3m$ (x, y, z)
- Speaker position: $[1, 2, 1.5]$
- Primary microphone position: $[2, 2, 1.5]$
- Backup microphone position: $[1, 4, 1.5]$
- Sound speed: 340 m/s
- Reverberation time[1], T60: 0.2 seconds

The room setup is illustrated in Fig. 4.4.

Two artificial room impulse responses from the speaker to the two microphones are calculated[2]. Let's denote the impulse response from the speaker to the primary microphone $h_p(k)$ and the one to the backup microphone $h_b(k)$. These two impulse responses are shown in fig. 4.5.

- **SET 2 (Recorded test signals):** In this set, the original signals are the same signals as in **SET 1** with the same specifications (Fig. 4.3). Real primary and backup signals are generated using these data as follows:

[1]Reverberation time is the time for reflections of a direct sound to decay by a fixed amount after the sound source is stopped. T60 is the time for which the decay amount is 60 dB below the level of direct sound [18].

[2]The calculations are done using image-source method and the codes are adopted from http://www.eric-lehmann.com/.

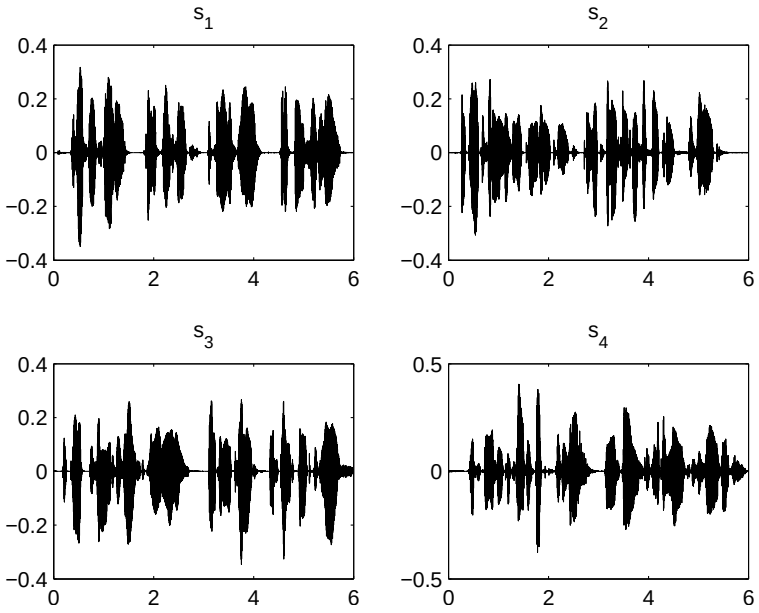


Figure 4.3 – Speech signals used for the evaluation of acoustic error concealment methods. s_1 and s_4 are male speakers and the rest are female speakers.

A Genelec 8020A speaker and two N810 devices are placed in a room with metal walls and absorbing ceil with dimensions $3.5m \times 4.5m \times 3m$. The distances of the devices from the speaker are $50cm$ and $150cm$. The speech samples of Fig. 4.3 are played through the Genelec speaker and are recorded by a voice recording software running on the N810 devices. The closer device is taken as the primary microphone and the other one serves as the backup.

Therefore, for each of **SET 1** and **SET 2**, we have eight signals (four primaries and four backups). In the primary signals, frame losses are simulated using a random packet number generator. In our experiments, 10% of the frames are dropped from the primary channel. In the dropped frame, all sample values are set to zero. The first one second of each signal is forced error-free to let the adaptive algorithms adapt.

4.1.1 Objective tests

The four methods presented in the discussion section of previous chapter are applied on the primary and backup signals of **SET 1** and **SET 2** to

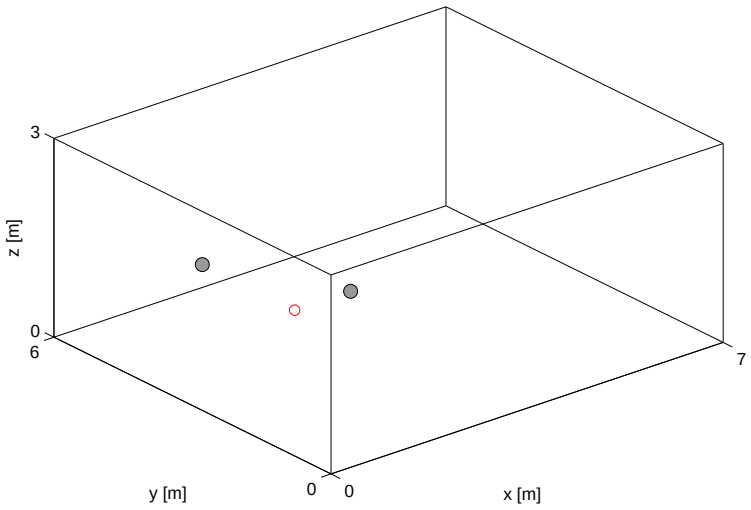


Figure 4.4 – Room setup for generating data for **SET 1** in the evaluation of acoustic error concealment methods. Red ring represents the speaker position and gray circles show the microphone positions. The primary microphone is closer to the source.

estimate the lost frames in the primary signals. By this process, for each set we have 24 signals for judging the quality (four signals and for each signal, six variations).

We can compare the quality of processed signals with the original one by calculating their loudness[3] using Zwicker's loudness model for stationary signals. This model assumes that the loudness is not directly related to overall stimulus intensity but is related to the sound pressure level within different auditory filters. The output of these auditory filters is converted into an excitation pattern. From these excitation patterns, specific loudness, (i.e., loudness per critical band) can be deduced using a nonlinear power law relationship. Moreover, total loudness is obtained by integrating these specific loudness values across frequency bands. For more about this model, the reader is referred to [20, 21].

A window is defined which moves along the primary signal and the estimated signal and looks for the lost frames. Whenever a lost frame is detected, the specific loudness of the primary signal and its difference with the estimated one (distortion) is found. At the values for specific loudness are

[3]Loudness is the magnitude of the physiological sensation produced by a sound, which varies directly with the physical intensity of sound but also depends on frequency of sound and waveform [19].

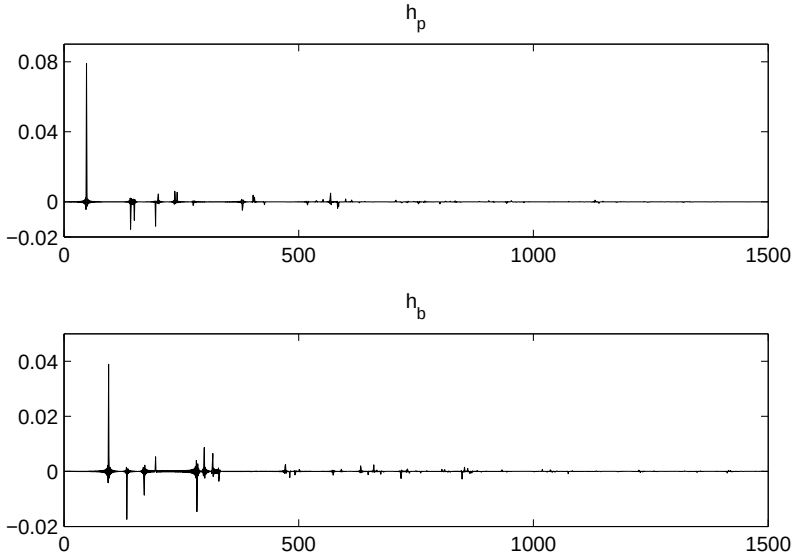


Figure 4.5 – Artificial room impulse responses for generating data for **SET 1** in the evaluation of acoustic error concealment methods. h_p and h_b are the transfer functions from the speaker to the primary and backup microphones respectively.

averaged over the lossy frames for both the primary signal and the distortion.

This is also repeated to find the specific loudness of the distortion for unprocessed signal (primary signal with lost frames). The resulted specific loudness for the data in **SET 1** are presented in Fig. 4.6. For a perfect reconstruction of the primary signal, the distortion loudness would be zero. Therefore, if the loudness values of the distortion are small, the method has found a better approximation of the primary signal. This explains the reason why the average loudness distortion is so close to the primary signal loudness in the case where no frame estimation is applied on the signal (Fig. 4.6(e)).

To compare the performance of the different methods, one can define a performance measure for them as the difference of the average specific loudness for the primary signal (SL_p) and the distortion signal (SL_d). We call this measure specific loudness difference (SLD):

$$SLD = SL_p - SL_d. \tag{4.2}$$

The results for specific loudness differences are shown in Fig. 4.7. In this case higher the curve for a method is, higher is the performance of that method in estimating the lost frames. These results show that the performance of the

variable step-size method is better than the other methods and it is close to the performance of perceptually motivated method. Also the two time delay estimation methods have similar performances, but eigenvalue decomposition method is slightly better. From the results also it is obvious that processing the signal improves the signal quality significantly.

The same procedure is applied for the real recorded data in **SET 2** and they are presented in Fig. 4.8. This shows that perceptually motivated method has a better performance than the variable step-size method in the lower frequencies but the behavior is worse in higher frequencies. In general the methods perform similarly for the artificial data and real recordings.

Also as the model suggests, the total loudness difference for each method would be the integral of the curves in Fig. 4.7 and Fig. 4.8. These values are shown in Table 4.1. These values also show that the overall performance of the methods does not change from artificial data to real recordings and the order of performance of the methods is the same as deduced from Fig. 4.7.

Method	**Total loudness difference**	
	SET 1	**SET 2**
Eigenvalue decomposition	4.0742	4.0407
Variable step-size	**7.3981**	**6.5777**
GCC-PHAT	3.7914	4.0871
Perceptually motivated	6.4326	6.1269
No processing	1.8983	1.9231

Table 4.1 – Total loudness difference for the results of different estimation methods applied to **SET 1** and **SET 2**. Higher total loudness differences suggest higher performance of the method.

4.1.2 Subjective Test

In order to verify the test results, an informal subjective test is also performed. For this test, the data in **SET 1** was used. This set is used to make sure that the participants are scoring the signals based on their quality in estimating the primary signal, not based on the noise components. As before, for each signal six possible outcomes are available; primary signal, unprocessed signal, and the outputs of the four algorithms.

For the subjective evaluation, Mean Opinion Score (MOS)[4] measure is used. In this test the sound quality was to be assigned with a number between

[4]In multimedia, the mean opinion score (MOS) provides a numerical indication of the perceived quality of received media after compression and/or transmission. The MOS is

1 and 5, 1 corresponding to a bad quality and 5 to an excellent quality. The test consisted of 24 sound samples (four signals and for each signal six variations) of length 6 seconds. First, two test samples were played for the participants in the test for the training purposes before the start of the actual listening test. Each test lasted approximately 6 to 10 minutes. Sound samples were repeatable and after each sound sample, the participant could score the quality and after scoring, the next sample was played. Sound samples were played on an IBM T41 laptop with SoundMAX Integrated Digital Audio sound card and the participants wore a set of Sennheiser HD650 headphones for listening to the samples. The samples were randomly permuted and based on the order of permutation, two sample tests were generated. Participants who had odd day of birth listened to set one and the rest to set two. Totally 24 people participated in the test and the measurements from two tests were merged. Three subjects were removed from the test due to inconsistent behavior in scoring undistorted (ideal) and unprocessed samples. Listening test statistics were calculated for 21 subjects.

These results are presented in Table 4.2. The mean value and the 95% confidence intervals are found for each method. A graphical illustration is shown in Fig. 4.9.

signal	**CIU**	**Mean**	**CIL**	**CI**
primary signal with loss	1.74	1.60	1.45	0.1413
primary signal without loss	4.82	4.71	4.61	0.1027
adaptive eigenvalue decomposition	2.99	2.83	2.67	0.1586
GCC-PHAT	3.28	3.11	2.94	0.1693
adaptive variable step-size	3.60	**3.45**	3.31	0.1462
perceptually motivated	3.35	3.17	2.98	0.1873

Table 4.2 – MOS subjective test results for 21 participants in the test. CIL is the lower bound of the confidence interval and CIU is the higher bound.

By comparing the subjective test results in Table 4.2 and Fig. 4.9 with objective test results in Table 4.1 and Fig. 4.10, we can see that spatial error concealment improves audio quality but the estimation errors are still noticeable. According to the test results, based on the performance, the methods can be ranked as: adaptive variable step-size, perceptually motivated, GCC-PHAT and adaptive eigenvalue decomposition.

Furthermore, comparing the performances of adaptive variable step-size

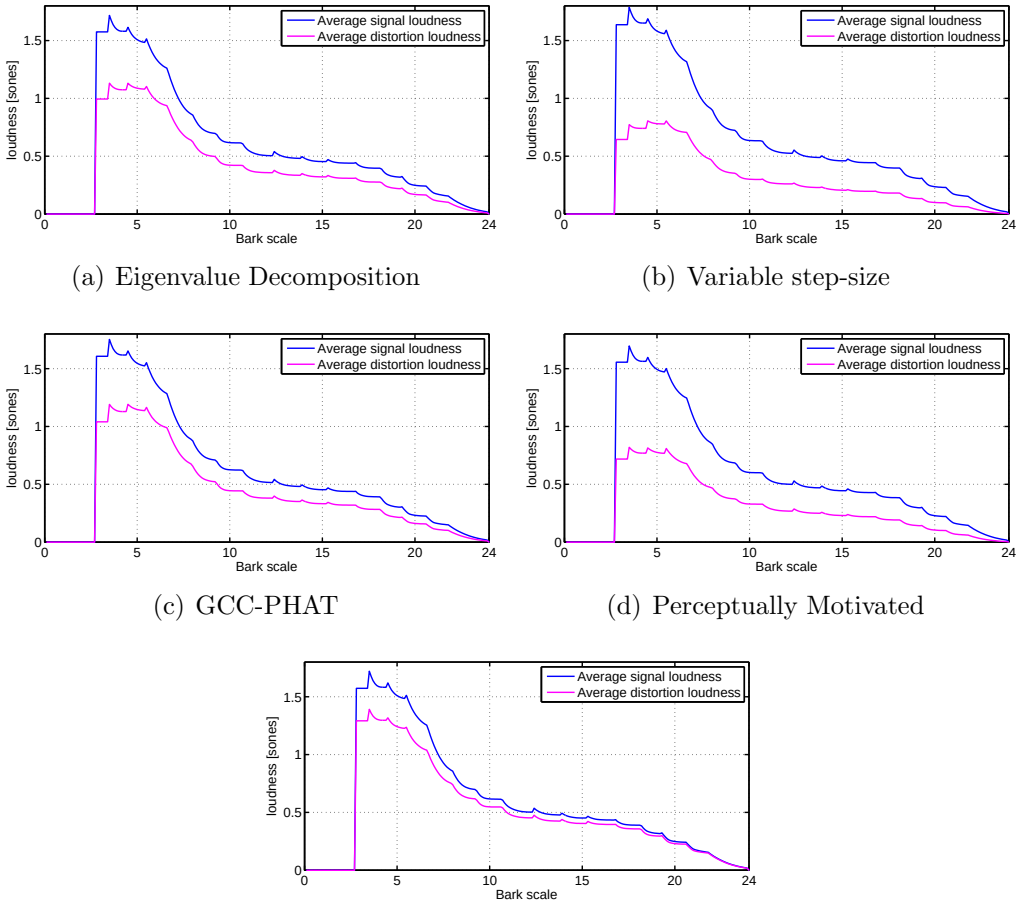
expressed as a single number in the range 1 to 5, where 1 is lowest perceived audio quality, and 5 is the highest perceived audio quality measurement [22].

and GCC-PHAT methods, shows that the difference in the performances is statistically significant (at significance level $P < 0.05$). This argument holds also for the difference in the performance of adaptive variable step-size method and adaptive eigenvalue decomposition method. In other words adaptive variable step-size method significantly (statistically) outperforms these two methods[5]. However, although the mean value for the adaptive variable step-size method is higher than the one for perceptually motivated algorithm, we cannot claim that they are significantly different in performance.

Figure 4.9 also suggests the similar findings from the objective tests that the variable step size method has the best performance comparing to the others, next is the perceptually motivated method and then GCC-PHAT and eigenvalue decomposition methods perform very similar to each other.

In this chapter we implemented four different transfer function estimation techniques in order to estimate the lost frames of the primary signal based on the backup signal and both the objective and subjective tests show that the adaptive variable step size method gives better estimations of the missing frames than the other methods. Moreover, we saw that post-processing the lost frames definitely improves the perceptual quality of the signals.

[5]This is because their confidence intervals do not overlap.



(a) Eigenvalue Decomposition

(b) Variable step-size

(c) GCC-PHAT

(d) Perceptually Motivated

(e) No Processing

Figure 4.6 – Average specific loudness for different estimation methods. In all the figures the above curve is the average specific loudness of the primary signal without loss. The lower curve in (a, b, c, d) represent the average loudness of the distortion after applying estimation methods. In (e) the lower curve represents the average specific loudness of the distortion when no error concealment is done. Higher the difference between two curves, better the method works.

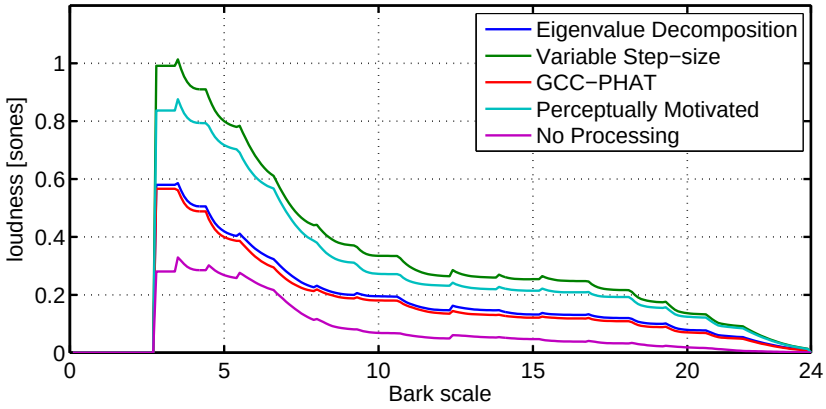


Figure 4.7 – Specific loudness difference for different methods applied to **SET 1** (simulated data). Higher values for SLD suggest higher performance of the method. This shows that the methods in the order of better performance are: variable step-size, perceptually motivated, eigenvalue decomposition, GCC-PHAT and no processing has the worst perceptual quality.

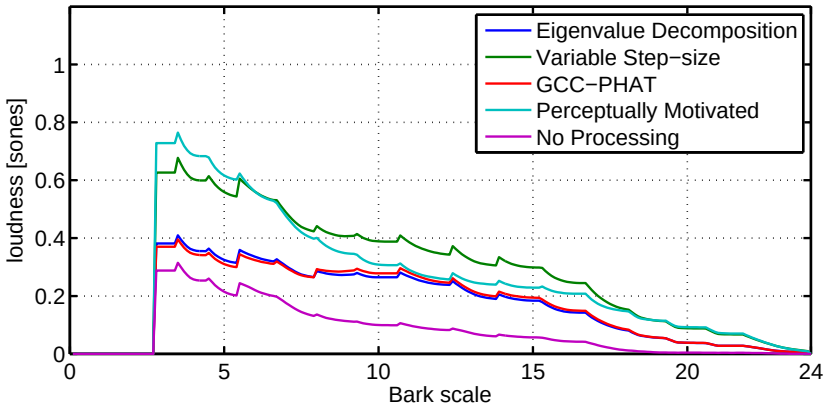


Figure 4.8 – Specific loudness difference for different methods applied to **SET 2** (real recordings). Higher values for SLD suggest higher performance of the method. This shows that perceptually motivated method performs better than the variable step-size method in lower frequencies. The overall performance is similar to the artificial data.

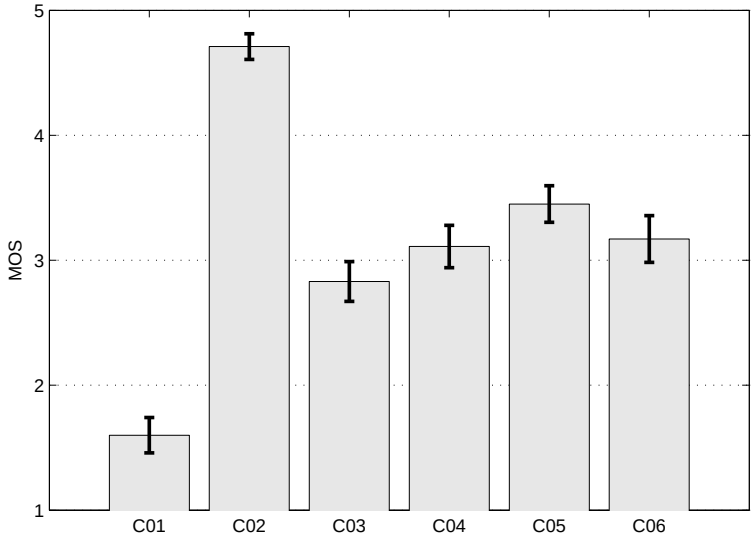


Figure 4.9 – Quality measure for different algorithms based on MOS subjective test performed on 21 participants. C01: primary signal with loss, C02: primary signal without loss, C03: adaptive eigenvalue decomposition method, C04: GCC-PHAT method, C05: adaptive variable step-size method, C06: perceptually motivated method.

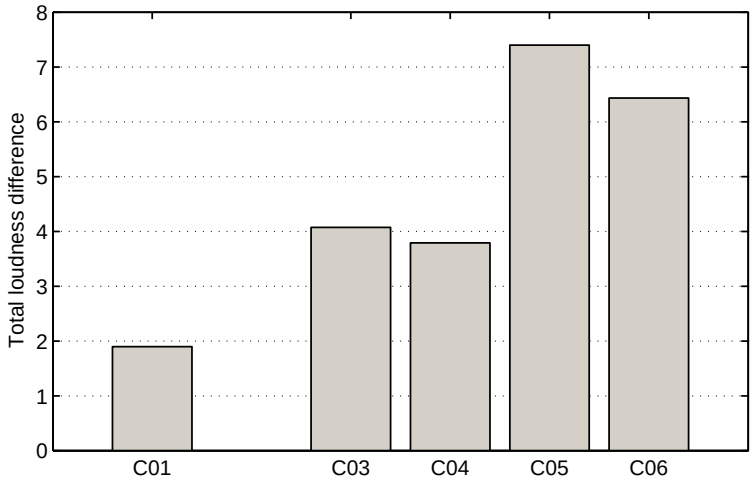


Figure 4.10 – Psychoacoustic measure for different algorithms based on objective test for data in **SET 1**. C01: primary signal with loss, C03: adaptive eigenvalue decomposition method, C04: GCC-PHAT method, C05: adaptive variable step-size method, C06: perceptually motivated method.

Chapter 5

Conclusions

In this work we have considered an ad-hoc audio conferencing setup, where the participants hold mobile communication devices with wireless connectivity. In this system, the clients are running on the mobile devices and connect to the conference server by VoIP services through a wireless link. Existing softwares for the conference mixing server and standard VoIP client for Nokia N810 devices are taken as the basis of the development and extensions to them have been proposed in order to solve the packet loss problem in the wireless audio conferencing.

To improve the quality of the downlink from the mixing server to the clients, the connection type is changed from unicast to multicast and saving the bandwidth has enabled us to send several copies of the same packet in order to increase the probability that at least one reaches the destination. To improve the quality of the uplink from the clients to the mixing server, we have proposed spatial error concealment techniques where instead of only one microphone, two (or more) devices capture the sound and communicate it to the mixing server. In this way, a device constructs the primary channel and the other devices serve as backup channels. If a frame is lost from the primary signal, an estimation of the frame is replaced based on the backup channel.

We have applied several methods for estimating the relation between the primary and backup signals. We have made an objective evaluation based on the Zwicker's loudness model on both simulated voice samples and real recordings to find the amount of distortion present in the signals before and after applying the methods. We also performed an informal MOS subjective test to determine the performance of each method. The results of the evaluations were consistent in the objective and subjective tests and they showed that spatial error concealment definitely improves the perceptual quality of the speech signals.

Evaluation results also showed that between the utilized methods, transfer function estimation methods perform better than time delay estimation methods and that the adaptive variable step-size method performs the best among the others. The spatial error concealment technique based on this method will be integrated in the real-time setup of the audio conferencing system.

In this work we considered the case where at each time instance there is only one active speaker, but as a future work one could address the problem with multiple active speakers.

We also assumed that the primary and backup devices are selected manually, however, one could associate algorithms for automatic channel ranking and device selection based on the characteristics of the captured signals by the devices.

Appendix A

Session Initiation Protocol

The Session Initiation Protocol (SIP) is a signaling protocol, used for controlling multimedia communication sessions such as voice and video calls over IP, video and voice conferencing and instant messaging. The protocol can be used for creating, modifying and terminating two-party (unicast) or multi-party (multicast) sessions. The modification can involve changing addresses or ports, inviting more participants, adding or deleting media streams, etc.

SIP was originally designed by Henning Schulzrinne and Mark Handley starting in 1996[1]. The latest version of the specification is RFC 3261[2] from the IETF Network Working Group.

The SIP protocol is an application layer protocol. SIP is designed to be independent of the underlying transport layer; it can run on Transmission Control Protocol (TCP) or User Datagram Protocol (UDP). SIP is not a vertically integrated communications system, it is a component that can be used with other IETF protocols to build a complete multimedia architecture. It is compatible with IPv4 and IPv6.

According to the RFC 3261, some of the elements of the SIP and their definitions are:

- **User Agent:** A logical entity that can act as both a user agent client and user agent server, i.e. that is capable of creating a new request and sending it through the client transaction machinary, or generating a response to a SIP request. Each resource of a SIP network, such as a User Agent, is identified by a Uniform Resource Identifier (URI). an example of such URI is "`sip:user1@192.168.1.1`".

- **Proxy Server:** "An intermediary entity that acts as both a server and a client for the purpose of making requests on behalf of other clients.

[1]Session Initiation Protocol-Wikipedia, the free encyclopedia.
[2]http://tools.ietf.org/html/rfc3261

A proxy server primarily plays the role of routing, which means its job is to ensure that a request is sent to another entity "closer" to the targeted user. Proxies are also useful for enforcing policy (for example, making sure a user is allowed to make a call). A proxy interprets, and, if necessary, rewrites specific parts of a request message before forwarding it."

- **Registrar:** "A registrar is a server that accepts `REGISTER` requests and places the information it receives in those requests into the location service for the domain it handles."

The SIP also consists of a set of requests which are listed below;

- `REGISTER`: Used by a UA to notify its current IP address and the URLs for which it would like to receive calls.
- `INVITE`: Used to establish a media session between user agents.
- `ACK`: Confirms reliable message exchanges.
- `CANCEL`: Is used to cancel a previous request sent by a client.
- `BYE`: Terminates a session between two users in a conference.
- `OPTIONS`: Requests information about the capabilities of a caller, without setting up a call.

A simple example of the establishment of a SIP call from RFC 3261 is presented. This scenario is the basic SIP trapezoid, U1 → P1 → P2 → U2. U1 sends:

```
INVITE sip:callee@domain.com SIP/2.0
Contact: sip:caller@u1.example.com
```

to P1. P1 is an outbound proxy. P1 is not responsible for domain.com, so it looks it up in DNS and sends it there. It also adds a Record-Route header field value:

```
INVITE sip:callee@domain.com SIP/2.0
Contact: sip:caller@u1.example.com
Record-Route: <sip:p1.example.com;lr>
```

P2 gets this. It is responsible for domain.com so it runs a location service and rewrites the Request-URI. It also adds a Record-Route header field value. There is no Route header field, so it resolves the new Request-URI to determine where to send the request:

```
INVITE sip:callee@u2.domain.com SIP/2.0
Contact: sip:caller@u1.example.com
Record-Route: <sip:p2.domain.com;lr>
Record-Route: <sip:p1.example.com;lr>
```

The callee at u2.domain.com gets this and responds with a `200 OK`:

```
SIP/2.0 200 OK
Contact: sip:callee@u2.domain.com:
Record-Route: <sip:p2.domain.com;lr>
Record-Route: <sip:p1.example.com;lr>
```

The callee at u2 also sets its dialog state's remote target URI to `sip:caller@u1.example.com` and its route set to:

```
(<sip:p2.domain.com;lr>,<sip:p1.example.com;lr>)
```

This is forwarded by P2 to P1 to U1 as normal. Now, U1 sets its dialog state's remote target URI to `sip:callee@u2.domain.com` and its route set to:

```
(<sip:p1.example.com;lr>,<sip:p2.domain.com;lr>)
```

Since all the route set elements contain the lr parameter, U1 constructs the following `BYE` request:

```
BYE sip:callee@u2.domain.com SIP/2.0
Route: <sip:p1.example.com;lr>,<sip:p2.domain.com;lr>
```

As any other element (including proxies) would do, it resolves the URI in the topmost Route header field value using DNS to determine where to send the request. This goes to P1. P1 notices that it is not responsible for the resource indicated in the Request-URI so it does not change it. It does see that it is the first value in the Route header field, so it removes that value, and forwards the request to P2:

```
BYE sip:callee@u2.domain.com SIP/2.0
Route: <sip:p2.domain.com;lr>
```

P2 also notices it is not responsible for the resource indicated by the Request-URI (it is responsible for domain.com, not u2.domain.com), so it doesn't change it. It does see itself in the first Route header field value, so it removes it and forwards the following to u2.domain.com based on a DNS lookup against the Request-URI:

```
BYE sip:callee@u2.domain.com SIP/2.0
```

The important characteristic of SIP is that it is compatible with multicast communications which enables us to use it for the ad-hoc audio conferencing system.

Appendix B

Real-time Transport Protocol

The Real-time Transport Protocol (RTP) was developed by the Audio-Video Transport Working Group of the IETF published as RFC 3550[1] in 2003. the RTP provides end-to-end delivery services for real-time data, such as interactive audio and video; services such as payload type identification, sequence numbering, time stamping and delivery monitoring. The RTP also supports data transfer in multicast mode.

The RTP defines a standard packet format for delivering audio and video. The packet header format which is of most interest for us is presented below

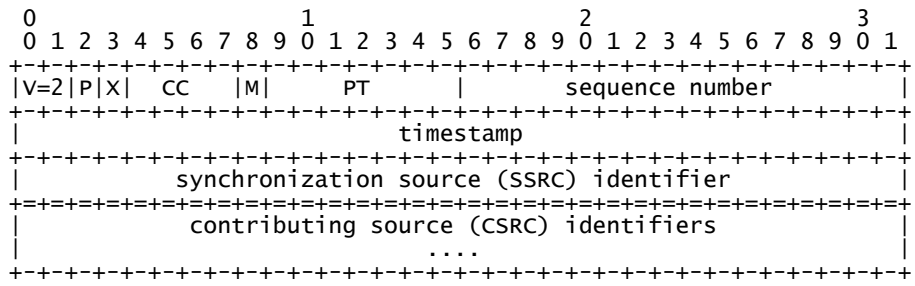


Figure B.1 – RTP packet header.

Some important fields of the header and their definitions are taken from RFC 3550 as follows:

- `CSRC count (CC)`: 4 bits
  The CSRC count contains the number of CSRC identifiers that follow the fixed header.

[1] http://tools.ietf.org/html/rfc3550

- `payload type (PT)`: 7 bits
  This field identifies the format of the RTP payload and determines its interpretation by the application. For example `PT=0` means that the payload is audio encoded as PCMU or `PT=26` means that the payload is video encoded as JPEG, etc[2].

- `sequence number`: 16 bits
  The sequence number is the counter of the sent RTP data packets. Receivers may use this field for packet loss detection. The initial value of the sequence number is chosen randomly by the RTP application.

- `timestamp`: 32 bits
  The timestamp reflects the sampling instant of the first octet in the RTP data packet. The sampling instant must be derived from a clock that increments monotonically and linearly in time to allow synchronization and jitter calculations. For example if an audio application is dividing the audio signal in 10ms frames and the sampling frequency is 16kHz, then the timestamp would be increased by 160 for each such block, regardless of whether the block is transmitted in a packet or dropped as silent. The initial value of the timestamp is also chosen randomly.

- `SSRC`: 32 bits
  The SSRC field identifies the synchronization source. This identifier should be chosen randomly, with the intent that no two synchronization sources within the same RTP session will have the same SSRC identifier.

- `CSRC` list: 0 to 15 items, 32 bits each
  The CSRC list identifies the contributing sources for the payload contained in this packet. The number of identifiers is given by the `CC` field. For example in an audio conference, the `SSRC`s of the contributing sources in a frame is put in the `CSRC` field of the packet containing that frame.

Again it is worth repeating that the RTP also supports multicast communication, and `sequence number` is the responsible filed for detecting multiple or lost frames.

[2]For more information refer to RFC 3551: http://tools.ietf.org/html/rfc3551.

Bibliography

[1] A. Dutta-Roy, Virtual Meetings with Desktop Conferencing, IEEE Spectrum **35**, 47–56 (1998).

[2] P. Comon, C. Jutten, and J. Herault, Blind separation of sources, Part II: problems statement, Signal Processing, Elsevier **24**(1), 11–20 (1991).

[3] P. Comon, Independent Component Analysis: a new concept?, Signal Processing, Elsevier **36**(3), 287–314 (1994).

[4] C. H. Knapp and G. C. Carter, The Generalized Correlation Method for Estimation of Time Delay, IEEE Transactions on Acoustics, Speech and Signal Processing **24**(4), 320– 327 (1952).

[5] P. R. Roth, Effective Measurements Using Digital Signal Analysis, IEEE Spectrum **8**(4), 62– 70 (1971).

[6] A. Zaknich, *Principles of Adaptive Filters and Self-learning Systems (Advanced Textbooks in Control and Signal Processing)*, pages 159–172, Springer, 2005.

[7] G. C. Carter, A. H. Nuttal, and P. G. Cable, The Smoothed Coherence Transform, Proceedings of the IEEE **61**(10), 1497–1498 (1973).

[8] C. Eckart, Optimal Rectifier Systems for the Detection of Steady Signals, SIO Ref 52-11 (1952).

[9] E. J. Hannan and P. J. Thomson, Estimating Group Delay, Biometrika **60**, 241–253 (1973).

[10] J. Benesty, Adaptive Eigenvalue Decomposition Algorithm for Passive Acoustic Source Localization, The Journal of the Acoustical Society of America **107**(1), 384–391 (1999).

[11] E. Hänsler and G. Schmidt, editors, *Topics in Acoustic Echo and Noise Control, Selected Methods for the Cancellation of Acoustic Echoes, the Reduction of Background Noise, and Speech Processing*, pages 90–122, Springer, 2006.

[12] V. Myllylä, Residual Echo Filter for Enhanced Acoustic Echo Control, Signal Processing, Elsevier **86**(6), 1193–1205 (2006).

[13] S. Haykin, *Adaptive Filter Theory*, pages 432–438, Prentice Hall, 3rd edition, 1996.

[14] E. Hänsler and G. Schmidt, *Acoustic Echo and Noise Control*, pages 73–88 and 303–307, John Wiley & Sons, 2004.

[15] J. O. Smith and J. S. Abel, Bark and ERB Bilinear Transforms, IEEE Transactions on Speech and Audio Processing **7**(6), 697–708 (1999).

[16] B. Moore and B. Glasberg, A Revision of Zwicker's Loudness Model, Acta Acustica united with Acustica **82**(2), 335–345 (1996).

[17] C. Faller, A Highly Directive 2-Capsule Based Microphone System, in *123rd Convention of the Audio Engineering Society*, New York, NY, October 2007.

[18] J. G. Roederer, *The Physics and Psychophysics of Music: An Introduction*, pages 147–152, Springer, 4th edition, 2008.

[19] M. Schroeder, *Springer Handbook of Acoustics*, Springer, 2007.

[20] E. Zwicker, H. Fastl, U. Widmann, and K. Kurakata, Program for Calculating Loudness Accourding to DIN 45631 (ISO 532B), The Journal of Acoustical Society of Jepan **12**(1) (1991).

[21] E. Zwicker, Subdivision of the Audible Frequency Range into Critical Bands (Frequenzgruppen), The Journal of Acoustical Society of America **33**(2) (1961).

[22] ITU, Methods for Subjective Determination of Transmission Quality, (Recommendation P.800) (1996).