

REAL TIME INDIVIDUALIZATION OF INTERAURAL TIME DIFFERENCES FOR DYNAMIC BINAURAL SYNTHESIS

**Echtzeitindividualisierung von interauralen
Laufzeitunterschieden in der dynamischen
Binauralsynthese**

Diplomarbeit



durchgeführt am: Fachgebiet Audiokommunikation,
Institut für Sprache und
Kommunikation

vorgelegt von: Jorgos Estrella

Studiengang: Elektrotechnik

██████████ ██████████
Gutachter: Prof. Dr. Stefan Weinzierl

M.A Alexander Lindau

Abgabedatum: 17. Mai 2011

Abstract

Under the premise that all cues required for spatial orientation in an acoustic environment are present in the sound pressure at the ear drums, it is possible, with the aid of binaural techniques, to virtually reproduce and/or synthesize acoustical environments by recreating the sound pressure as it would occur in natural hearing.

Lord Rayleigh ¹ on his theory of sound perception, stated that two cues are mostly responsible for providing spatial acoustic orientation: The interaural level difference (ILD) and the interaural time difference (ITD), both are related to physiological characteristics, thus, different among individuals.

Head related impulse responses (HRIR) contain those cues and provide through a convolution process the auralized acoustic information. However, the use of non individual HRIRs in binaural synthesis can lead to severe degradation of the virtual acoustical experience.²

This work presents and discusses the practical implementation of a method to achieve individualization in the binaural synthesis by means of real-time ITD manipulation. Moreover it presents and discusses the development of an empirical formula to scale the foreign ITD based on anthropometric measures.

¹Strutt 1907

²Algazi et al. 1997; Wenzel et al. 1988, 1993

Eidesstattliche Erklärung

Die selbständige und eigenständige Anfertigung versichert an Eides statt.

Berlin, den 17. Mai 2011

JORGOS ESTRELLA

Scope of the present work

At this point it is important to mention that some aspects that support the methods described in this thesis were addressed theoretically by the author in a previous work (Estrella 2010). Therefore, these are only briefly explained here.

Much of this thesis' development work deal with technical implementation facets required to accomplish the software application. The media CD accompanying this work contains the code-sources and documentation of the mentioned software. These technical aspects will not be extensively treated in this Diploma thesis.

In order to facilitate the reading of the digital version of this document, the media-CD contains a copy with colored hyper-references.

Chapter 1 describes the problematic use of non-individual auditory cues in the binaural synthesis and introduces the proposed individualization model as well as its advantages.

Chapter 2 briefly explains several general aspects of binaural synthesis technology to finally describe some individualization approaches.

Chapter 3 explains, using system theory, the fundamentals of the proposed individualization model and its components.

Functional requirements for the practical implementation as well as the software components used and the most important implementation facets are explained in Chapter 4. A flowchart and schematics are also presented.

Once the software was developed a listening test was conducted, in order to establish a quantitative relationship between individual morphology and ITD manipulation. Chapter 5 explains this approach.

Chapter 6 presents a brief resume of this work and mentions the aspects that should be addressed in a future work.

The appendix contains brief resume in German language, the Matlab™ code for the data set preparation, the *Readme* files that explain the usage of the software, and other software applications written to support the development of this work.

Contents

Abstract	II
Eidesstattliche Erklärung	III
Scope of the present work	IV
List of Figures	VIII
1. Motivation	1
1.1. The use of non individualized HRIRs in the binaural synthesis	2
1.2. The proposed individualization model	3
2. State of the art	6
2.1. Fundamentals of binaural synthesis	6
2.1.1. Encoding binaural information	8
2.1.2. Decoding binaural information	8
2.1.3. HRTF and localization cues	9
2.1.3.1. Low and high frequency ITD	10
2.1.3.2. ITD estimation in binaural data sets	11
2.1.3.3. ILD as localization cue	13
2.1.3.4. SC as localization cue	14
2.1.4. Dynamic binaural auralization	15
2.2. Individualization using geometrical models	16
2.3. Anthropometric aided individualization	17
3. Methods	19
3.1. Chapter's overview	19
3.2. Separate processing of time and spectral characteristics	19
3.3. Extraction of minimum-phase impulse responses	20
3.4. Fractional delay filters	23
3.5. Chapter's resume	26
4. Implementation	28

4.1.	Data set pre-processing	28
4.2.	Functional requirements	29
4.3.	Software components	29
4.3.1.	Low latency high priority audio thread: The JACK Audio applica- tion programming interface (API)	30
4.3.2.	Delay-lines based on sample rate conversion (SRC)	32
4.3.3.	The libsamplerate API	34
4.3.4.	OSC control: The liblo API	36
4.3.5.	XML script parsing: The libxml++-2.x API	38
4.3.6.	GUI control: The GTK+2.0 Project	40
4.4.	Flowchart of the audio process	41
5.	Anthropometry-based ITD individualization	45
5.1.	Listening test: ITD Individualization by scaling	45
5.1.1.	Test setup	45
5.1.1.1.	Stimulus	47
5.1.1.2.	Listening-test's software	48
5.1.1.3.	Interface	48
5.1.2.	Listening test procedure	50
5.2.	Statistical analysis	50
5.3.	Anthropometry-based ITD individualization formula	51
6.	Summary and conclusions	54
	Bibliography	56
A.	Zusammenfassung in deutscher Sprache	i
A.1.	Motivation	i
A.1.1.	Die Verwendung von nicht-individuellen Lokalisationscues	i
A.1.2.	Das vorgeschlagene Individualisierungsmodell	ii
A.1.3.	Vorteile des Modells	iii
A.2.	Stand der Forschung	iv
A.2.1.	Individualisierung mit Hilfe von anthropometrischen Maßen	iv
A.3.	Methoden	v
A.3.1.	Ermittlung der ITD aus binauralen Datensätzen mittels Onset De- tektion	vi
A.3.2.	Extraktion von minimalphasigen Impulsantworten nach der Onset Detektion Methode	vii
A.3.3.	Fraktionale Zeitverzögerung	vii

Contents

A.4. Implementierung	viii
A.4.1. Data-Set Vorbereitung	ix
A.4.2. Softwarekomponenten	ix
A.5. Anthropometrie-basierte ITD Individualisierung	xi
A.5.1. Hörversuch zur Ermittlung des individuellen ITD-Skalierungsfaktors	xi
A.5.1.1. Hörversuchsaufbau	xi
A.5.1.2. Versuchsdurchführung	xiii
A.5.2. Statistische Auswertung	xiii
A.5.3. Anthropometrie-basierte ITD-Individualisierungsformel	xv
A.6. Zusammenfassung und Diskussion	xv
B. Matlab code for data set preparation	xvii
C. ITD-Individualizer usage	xxii
C.1. Installation	xxii
C.1.1. Dependencies	xxii
C.1.2. Compilation	xxii
C.2. Usage	xxii
D. A software for controlling multiple ITD individualizers	xxvi
E. A software to control the volume of audio streams	xxx
F. Audibility of Doppler effect due to head rotation	xxxii

List of Figures

1.1.	Kemar manikin type 45BA. Source G.R.A.S	2
1.2.	Simplified schematic of the proposed individualization model	4
2.1.	Graphical example of the binaural synthesis concept. Up: Perception of a real sound source. Down: reconstruction of the sound pressure at the eardrums through binaural synthesis.	7
2.2.	Binaural data set acquisition using the head and torso simulator FABIAN (Lindau 2006).	8
2.3.	H _p TF dependency on headphone positioning and the individual morphology. 10 successive headphone positions (Sennheiser D600) at the left ear of a representative subject are shown. The curves are shifted by 5 dB for legibility purposes. Note the differences for frequencies above 5 kHz. From Nicol (2010)	9
2.4.	Path length differences causing the interaural time difference	10
2.5.	ITD as a function of frequency on increasing azimuths (17° to 90°) computed using HRTFs of selected subjects. 6 sound source locations in the azimuth plane were considered. Note the low and high frequency ITD. From (Nicol 2010).	11
2.6.	ITD extracted using the onset detection method with 10x up-sampling, threshold -3dB. Data set: FABIAN's HRIRs (elevation: 0°, azimuth: -180° to +180°, resolution: 1°)	13
2.7.	Subjective ITD vs. ITD extracted with the Edge Detection method. Means of subject's answers and standard deviations are plotted with continuous lines. Dotted line represent the ITD estimation method. Only the horizontal plane is considered. From Busson et al. (2005)	14
2.8.	Acoustic shadowing causing the interaural level difference	15
2.9.	Woodworth and Schlosberg's ITD computation method based on a spherical head model. From (Kuhn 1977).	16
2.10.	Anthropometric measures used to find the optimal head radius in Algazi et al. (2001b)	17

List of Figures

3.1.	(up) HRTFs with and without phase component. (down) frequency response of both HRTFs. Source (Kulkarni et al. 1999)	20
3.2.	Results of ABX listening test of minimum-phase IRs (Hilbert method) vs. original impulse responses.	22
3.3.	Results of ABX listening test of minimum-phase IRs (extracted with the onset detection method) vs. original impulse responses.	23
3.4.	Extraction of quasi minimum-phase impulse responses with the onset detection method. Note that the envelope has slightly changed due to manipulation. It was these kind of differences that were assessed for audibility in the second listening test of section 3.3	24
3.5.	Basic discrete delay system.	25
3.6.	Ideal fractional delay approximation. Up: delay D is integer, sampling occurs at zero crossings. Down: delay D is non integer, sampling occurs between zero crossings. Infinite length impulse response is required in the ideal case. From Välämäki and Laakso (2000)	26
4.1.	Time stretching for achieving one sample delay ($22\mu\text{s}$). Note the use of sample rate conversion at the stretching region.	34
4.2.	Graphical user interface of the ITD individualizer developed using GTK+2.2	40
4.3.	Flowchart of the callback function that manages time stretching.	42
4.4.	Schematic description of the processing callback-function of the ITD individualizer	44
5.1.	Relevant anthropometric measures defining the individual ITD	46
5.2.	Listening test setup. Up: While using the reference speaker. Down: while using binaural system.	46
5.3.	Low-pass filter applied to the noise-burst stimulus to minimize the lateralization influence of ILD.	47
5.4.	Graphical user interface of the listening test audio application.	48
5.5.	Protocol of a selected user's listening test written in CSV format. Red arrow points to the column of the generated ITD scaling factors	49
5.6.	User interface for the listening test	49
5.7.	Distribution of the individually generated ITD scaling factors from 9 subjects. Note the big dispersions.	51
5.8.	Modeling of listening test results: The linear regression model over the intertragus distance is shown with hyperbolic 95% CIs.	53
A.1.	Data-Set Akquise unter Verwendung des HATS (Head and torso simulator) FABIAN (Lindau 2006).	ii

A.2. Vereinfachtes Schema des ITD- Individualisierungsmodells	iii
A.3. ITD ermittelt durch Onset Detektion bei 10facher Überabtastung, Detektionschwelle -3dB. Data Set: FABIAN's HRIRs (Elevation: 0°, Azimut: -180° bis +180°, Auflösung: 1°)	vi
A.4. Extraktion von minimalphasigen Impulsantworten mit der Methode der Onset Detektion. Die leichte Veränderung der Einhüllenden beruht auf dem Manipulationsvorgang an den BRIRs.	vii
A.5. Ideale fraktionale Zeitverzögerung. Oben: Delay D ist ganzzahlig, die Abtastung erfolgt an den Null-Übergängen. Unten: Delay D ist nicht ganzzahlig, Abtastung erfolgt zwischen den Null-Übergängen. Dem Idealfall entspricht eine Impulsantwort unendlicher Länge. Aus Välimäki and Laakso (2000)	viii
A.6. Grafische Benutzeroberfläche der ITD-I Software unter GTK+2.2	x
A.7. Relevante anthropometrische Maße für die Bestimmung der individuellen ITD	xi
A.8. Frequenzgang des Tiefpassfilters zur Minimierung des Lokalisationseinfluss der ILD.	xii
A.9. Verteilung der individuell hergestellten ITD-Skalierungsfaktoren von 9 Versuchsteilnehmern. Die großen Streuungen sind anzumerken.	xiv
A.10. Modellierung der Hörversuchsergebnisse: lineares Regressionsmodell dargestellt über die Intertragusdistanz mit 95% Konfidenzintervallen.	xiv
D.1. (up) Application developed to control multiple instances of the ITD individualizer. (down) Configuration file for determining the OSC ports to transmit commands to.	xxvii
E.1. Help file of an application to pre-set volume on audio streams.	xxxii
E.2. GUI of the software application to pre-set and modify volume.	xxxii

1. Motivation

The realization of virtual acoustical displays has been for many years subject of investigation. One approach to accomplish this goal is binaural synthesis. Binaural reproduction techniques are based on the assumption, that the sound pressure at the entrance of the ear canals contains the cues required for spatial perception of auditory environments. Thus, reproducing that sound pressure (i.e over headphones) recreates the acoustic scene.

Head-related transfer functions (HRTF) contain the temporal and spectral cues required for spatial orientation since they contain the position-dependent changes that occur when a sound wave propagates from a sound source to a listener's eardrum (Algazi et al. 1997). Already in the early years of the 20th century Lord Rayleigh on his duplex theory of hearing (Strutt 1907) mentioned two characteristics as being responsible for the localization of sound sources:

- The interaural time difference (ITD), which consists on the time difference of a sound wave arriving the left and right ears.
- The interaural level difference (ILD), being caused by diffraction of the sound wave due to reflections on the subjects head, torso and pinna³.

Further studies (Kuhn 1977; Mills 1958) demonstrated that the ITD is frequency independent up to 500 Hz and above 3 kHz. It also has a predominant influence on the localization in the horizontal plane and for frequencies up to 1.4 kHz. On the other hand, the ILD is a predominant localization cue in the median plane and for frequencies above 1.4 kHz.

The simplest way for achieving virtual environments is the direct reproduction of binaural recordings acquired with head and torso simulators (HATS) (see fig. 1.1). A more flexible approach is to convolve audio information with, either binaural room impulse responses (BRIR) recorded in real rooms, or mathematically computed impulse responses based on physical parameters.

³Lord Rayleigh's theory contemplates an ILD due to scattering on a sphere as head model, in this work reflections on torso and pinna are also referred to.



Figure 1.1.: Kemar manikin type 45BA. Source [G.R.A.S](#)

It has been proven that head movements significantly reduce the localization errors⁴ by reducing front-back confusion ([Begault et al. 2000](#)). Therefore another enhancement in the perceived plausibility of the acoustic display can be achieved using a head-tracked system to dynamically change the impulse responses according to the subject head's position.

1.1. The use of non individualized HRIRs in the binaural synthesis

As mentioned before ITD and ILD are related to the physical structure of the subject. The maximum ITD varies in proportion to the subject's head dimensions. The ILD relates to the form and size of head, torso and pinna. Thus, these are individual localization cues and can not be interchanged among subjects without audible consequences.

Unfortunately, in practice it is not feasible to conduct individual HRTF recordings⁵. They are usually acquired with the aid of HATS (see fig. 1.1), so they enclose the manikin's lo-

⁴"Localization error refers to the deviation of the reported position of the sound stimulus from the measured or synthesized target location" [Begault et al. 2000](#), pg. 905.

⁵In the special case of synthetic HRTFs, individualization can be achieved based on geometric models [Algazi et al. \(2001b\)](#).

calization cues.

The problematic in the use of non-individual head related impulse responses in the binaural synthesis was widely investigated and explained (Algazi et al. 1997; Wenzel et al. 1988, 1993). The most remarkable problems can be ordered in two categories:

- **Tone color variations** given by the non-individual ILD having different spectral characteristics. Though this is not critical since they are only noticeable in direct comparison with the real sound fields.(Møller and Hoffmann 2006)
- **Localization errors** due to a non-individual ITD are, on the contrary, more disturbing since they cause constant localization offsets and/or instability of sound sources on head tracked binaural systems. Algazi et al. (2001b) mentions the probably most disturbing issue of sound sources slightly moving in the opposite direction as the listener's head, if the artificial head used in the data acquisition had a bigger radius, or moving in the same direction of the subject's head movement, if the artificial head had a smaller radius.

1.2. The proposed individualization model

To overcome localization errors due to a non-individual ITD in the the data set, this work proposes an individualization method consisting on the **frequency-independent real-time manipulation of the ITD**.

The method can be resumed in three steps:

- The angle dependent $ITD(\theta, \phi)$ (azimuth, elevation) of the HRIR dataset to be used has to be first collected in a reference matrix.
- All impulse responses pairs of the dataset are replaced by it's minimum-phase representations at the convolution stage.
- The missing interaural time differences of the minimum phase HRIRs are reinserted in real-time as a subsample delay between the left and right ear signals using the ITD (collected in the previously mentioned reference matrix) scaled by an individualization factor.

Figure 1.2 shows a schematic description of the individualization model.

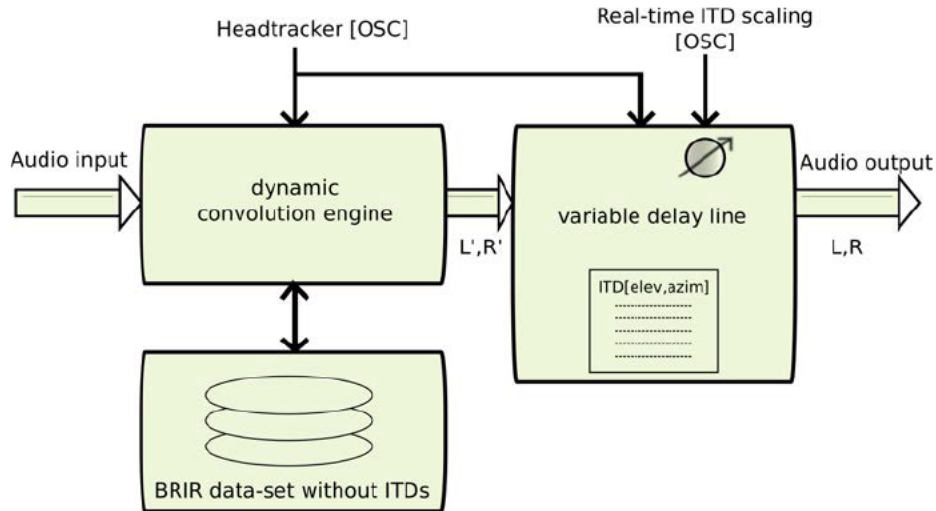


Figure 1.2.: Simplified schematic of the proposed individualization model

Advantages of the model

The advantages that arise as a consequence of the individualization methodology can be listed as follows:

Improvement of the simulation's authenticity The properly individualized ITD makes it possible to completely eliminate the instability of sound sources issue, when using generic HRIR datasets. It also improves the localization in the horizontal plane and in the end the overall plausibility of the simulation.

Latency and processing load reduction at the convolution stage Since the modified IR dataset, now containing minimum-phase impulse responses, has become shorter, there are less samples to process at the convolution stage. This reduces latency and processing load.

Artifact free cross-fading It is well known that addition of coherent, time-shifted sound sources leads to comb-filter artifacts. The use of IRs with extracted time difference in the cross-fading stage avoids degradation due to omission, repetition and comb filtering; thus improving the overall auditory quality.

Improved CPU load distribution The treatment of sound-source rendering on separate processes (i.e. one process for each source) makes it possible in multiprocessor systems, to keep the overall CPU load evenly distributed, provided that the operative system's kernel can efficiently parallelize tasks.

Preservation of the natural hearing characteristics The model presented on fig.1.2 foresees a time stretching algorithm for the reinsertion of the interaural delay. Such a behavior resembles, on a head-tracked system, the Doppler effect mechanism as it occurs in natural hearing⁶. This would be the case if the listener rotates his/her head, thus nearing one ear to the sound source while respectively moving the other ear away.

Reduction of resolution requirements at the dataset acquisition Since temporal and spectral cues are handled in separate processes they can be realized with different spatial resolutions. This could allow recording the head related impulse responses with coarser resolution, while the temporal characteristics are provided at a finer resolution (i.e. by means of interpolation). This could also reduce the impulse response storage requirements.

⁶Frequency shifts (ie. Doppler effect) due to head rotation fall under the audibility threshold. See appendix F

2. State of the art

This chapter reviews the state of the art of binaural systems and binaural individualization approaches. Section 2.1 explains briefly binaural techniques, section 2.2 explains some individualization approaches based on geometric modeling and section 2.3 covers an individualization method based on anthropometry.

2.1. Fundamentals of binaural synthesis

Like wave field synthesis (WFS), Ambisonics or stereophony, binaural synthesis is a technology for sound spatialization. In other words, it aims to convey the illusion of sound sources located in the space.

Binaural technology is based on the axiom that all cues required for spatial hearing are encoded in the sound pressure at the eardrums, thus, audio information convolved with those spatial cues recreates or synthesizes virtual auditory environments (VAEs). Figure 2.1 presents a typical setup of the sound wave reconstruction.

Natural cues of auditory localization result, as explained in the previous chapter, from the diffraction of the acoustic wave by the listeners morphology.

Rozenn Nicol in a recent publication (Nicol 2010) mentions some advantages of binaural technology.

- Compactness, in the sense that this system is very portable, being able to create virtual sounds anywhere around the listener (full 3D spatialization) with only 2 signals (ie. over headphones).
- Low cost, in terms of signal capture, storage and delivery.
- Compliant with loudspeaker reproduction, by using crosstalk cancellation.
- Good quality of sound spatialization in terms of immersion (Nicol 2010)

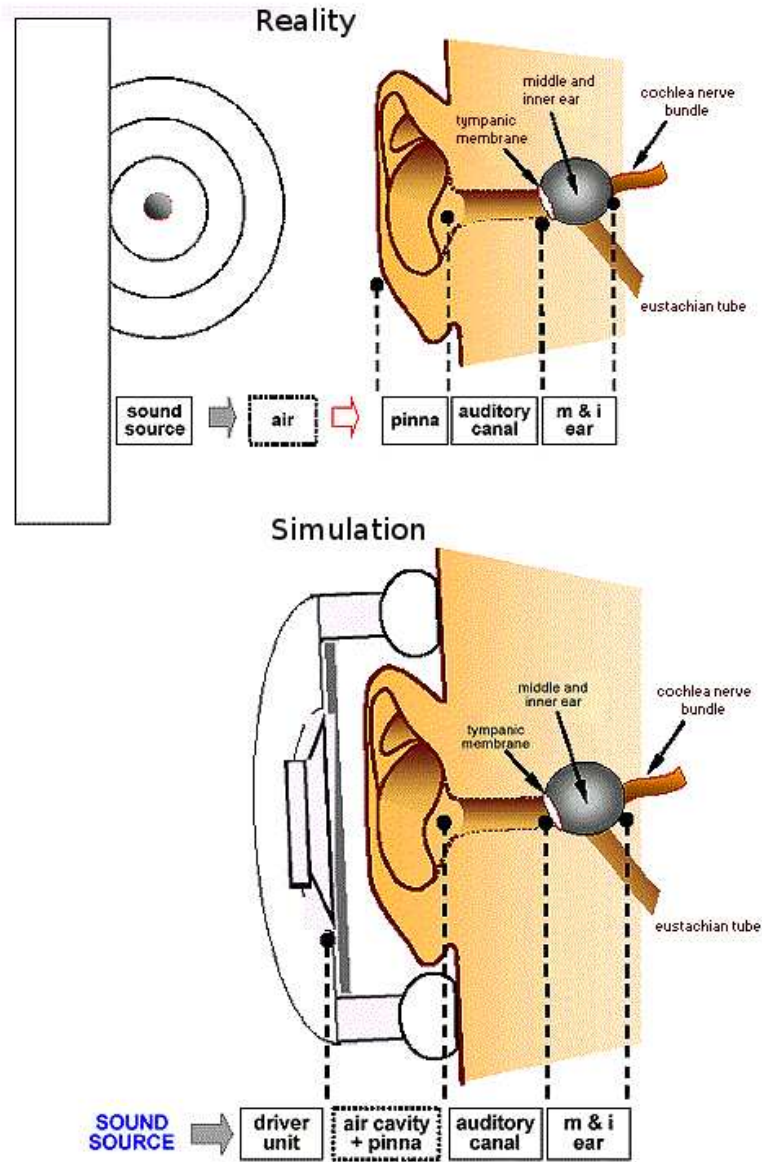


Figure 2.1.: Graphical example of the binaural synthesis concept. Up: Perception of a real sound source. Down: reconstruction of the sound pressure at the ear-drums through binaural synthesis.

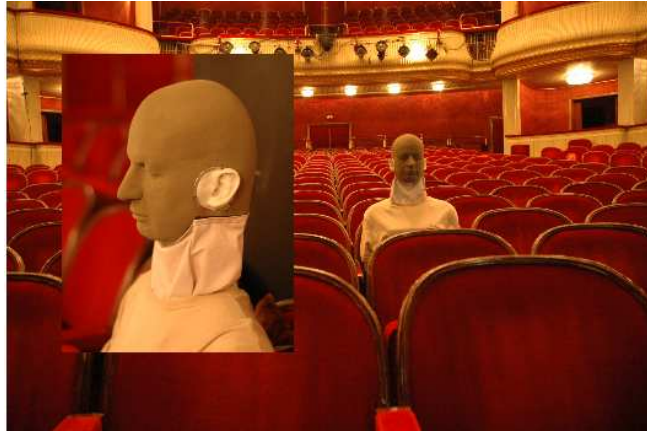


Figure 2.2.: Binaural data set acquisition using the head and torso simulator FABIAN (Lindau 2006).

2.1.1. Encoding binaural information

The process of gathering cues for spatial localization, known as binaural encoding, is based on the synthesis of pertinent temporal (ITD) and spectral characteristics (ILD and spectral cues (SC)). There are several ways to acquire binaural signals, the most straightforward consists of recording an acoustic scene with the aid of a HATS (see fig. 2.2) or a human subject wearing microphones at the entrance of the ear-canals. Another way to create an artificial acoustic scene is to use binaural synthesis. This technique consists in convolving anechoic audio with binaural room transfer functions (BRTFs) or binaural room impulse responses (BRIRs) that describe the acoustic path between the sound source and the listener's ears. Therefore the acquisition of binaural information consists in gathering data sets of HRTFs or their temporal counterpart head-related impulse responses (HRIR).

2.1.2. Decoding binaural information

Reproduction of the encoded sound pressure at the ear drums is not a trivial matter since headphones/earphones are not acoustically transparent and the recording and reproduction points are not identical, thus, headphone calibration is required (McAnally and Russell 2002). Frequency calibration accounts for:

- the frequency response of the acoustic emitter of the headphones,
- the acoustic coupling between headphone and the listener's external ear

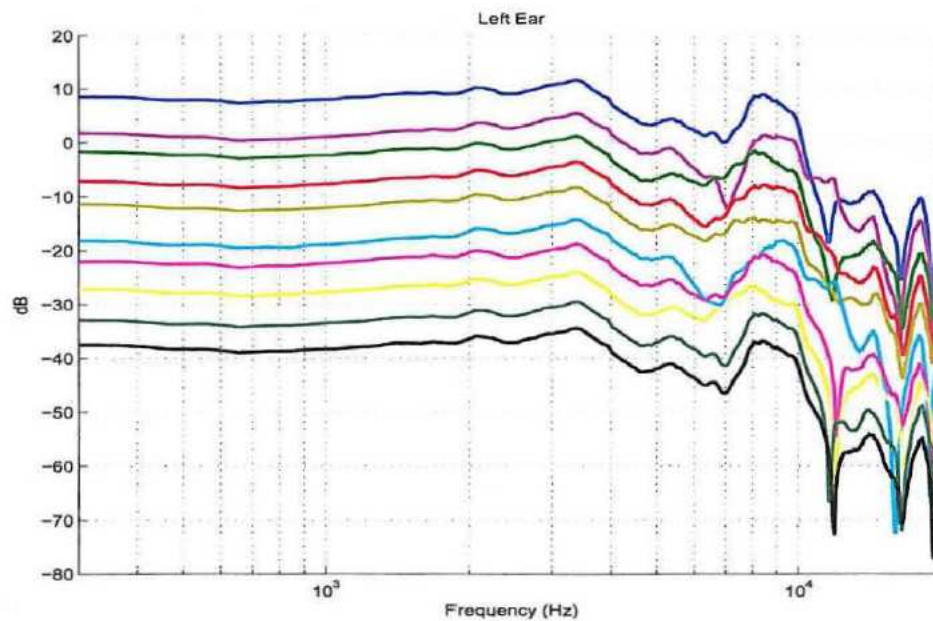


Figure 2.3.: HpTF dependency on headphone positioning and the individual morphology. 10 successive headphone positions (Sennheiser D600) at the left ear of a representative subject are shown. The curves are shifted by 5 dB for legibility purposes. Note the differences for frequencies above 5 kHz. From Nicol (2010).

Since the headphone transfer function (HpTF) depends not only on the individual pinna morphology, but also on the headphone positioning (Møller et al. 1995) a common compromise to achieve calibration is to realize several measurements and compute the average (Nicol 2010). Figure 2.3 shows an example of 10 HpTF measurements at 10 consecutive positions. Note the differences above 5 kHz.

2.1.3. HRTF and localization cues

As already mentioned in Chapter 1 HRTFs contain the cues for spatial localization which are the essence of binaural synthesis.

Let us in this section discuss in more detail further characteristics of these localization cues.

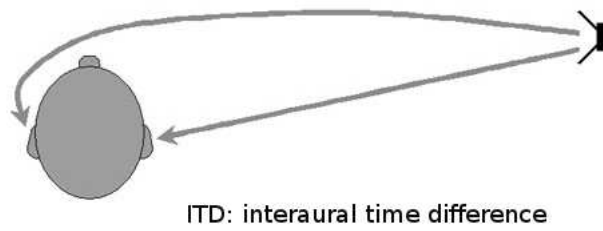


Figure 2.4.: Path length differences causing the interaural time difference

2.1.3.1. Low and high frequency ITD

[Kuhn \(1977\)](#) analyzed the frequency dependency of the ITD based on the diffraction of a plane wave on a spherical head model. In his work the interaural phase difference (IPD) is computed as:

$$IPD = 3ka \cdot \sin(\theta) \quad (2.1)$$

where k is the wave number, a is the head radius and θ the azimuth incidence angle in radians. The ITD is then:

$$ITD = \frac{IPD}{2\pi f} \quad (2.2)$$

At low frequencies assuming $(ka)^2 \ll 1$ the ITD becomes:

$$ITD_{lowF} = \frac{3a}{c} \cdot \sin(\theta) \quad (2.3)$$

where c is the speed of sound.

At high frequencies the diffracted wave can be considered as wave propagating at a speed close to the speed of sound c . Thus, it can be derived from the path difference of the symmetrically disposed ears including the path circumventing the spherical head. The ITD becomes:

$$ITD_{highF} = \frac{a}{c} \cdot (\sin\theta + \theta) \quad (2.4)$$

This equation matches exactly Woodworth's formula⁷ which is therefore identified as a high frequency modeling of the ITD ([Algazi et al. 2001b](#)).

For angles close to the median plane ($\theta \approx \sin(\theta)$) ITD_{lowF} is 150% bigger as the ITD_{highF} . Figure 2.5 shows the $ITD(f)$ of two human subjects where the above mentioned relationship can be seen.

⁷Woodworth's formula will be addressed with more detail in section 2.2

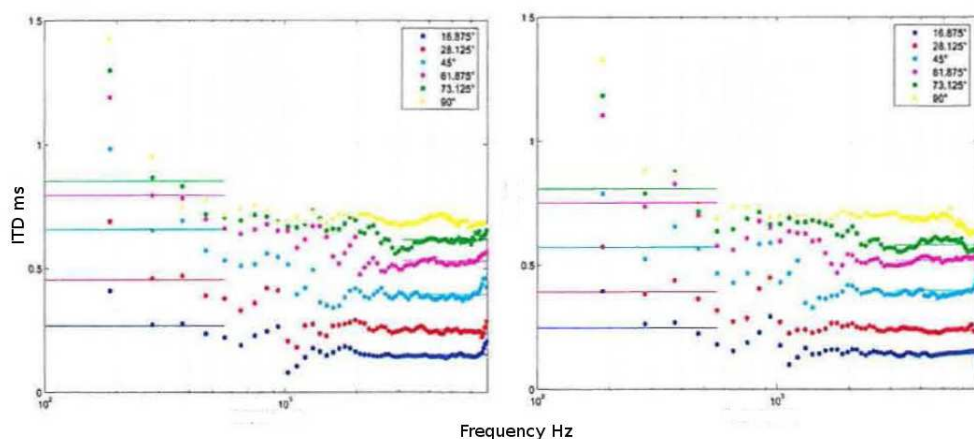


Figure 2.5.: ITD as a function of frequency on increasing azimuths (17° to 90°) computed using HRTFs of selected subjects. 6 sound source locations in the azimuth plane were considered. Note the low and high frequency ITD. From (Nicol 2010).

2.1.3.2. ITD estimation in binaural data sets

In a previous work (Estrella 2010) and under the scope of ITD individualization, several ITD estimation methods were analyzed and compared with each other. In the following a brief description of those methods and the results of the analysis is given.

Maximum of the interaural cross-correlation (MIACC) This method consists of cross correlating the impulse responses of left and right ears with each other and measure the time to it's maximum. According to Mills (1958) the threshold for detection of ITD changes is approx. $10\mu\text{s}$ when the conditions are optimal. For a samplerate of 44100 Hz the time difference between one sample to another is already $22\mu\text{s}$. Therefore, for appropriate accuracy, the HRIRs should be first up-sampled.

This method seems to give erratic ITD values at lateral positions. These occurs most probably, due to the minor SNR of the contralateral IR and the lack of coherence between ipsilateral and contralateral IRs at those angles (Busson et al. 2005).

Cross Correlation with minimum phase impulse responses Nam et al. (2008) showed that for the vast majority of HRIRs, the correlation between an impulse response and its minimum phase representation is over 0.9, thus, finding the times until maximum of this type of cross-correlation for left and right HRIRs and subtracting them from each other gives us the ITD.

The method also presents discontinuities at lateral angles (around $\pm 50^\circ$ to 130°). It also requires a lot of processing time because the extraction of the minimum phase impulse responses and the cross-correlation are both realized with up-sampled IRs. BRIRs of large rooms which already consist on large vectors become problematic in this sense.

Onset detection This method, also known as edge detection, measures the time up to a given threshold in the left and right onsets of the binaural IRs (ie. 10% of the peak in [Minnaar et al. \(2000\)](#)). The ITD equals the difference between the times found. For appropriate accuracy the IRs should be up-sampled.

Visual inspection of the data set helps finding an appropriate threshold. The ITD in BRIRs is reliably detected when using thresholds of -20 to -40 dB of the maximum peak. This estimation method performs quite fast and robust but it depends on the chosen threshold.

Interaural group delay difference at 0Hz, (IGD₀) A HRTF can be decomposed in minimum-phase and excess phase component ⁸. In this approach the ITD is the interaural group delay difference of the excess phase components evaluated at 0Hz.

However as binaural data sets are recorded using real electro acoustical transducers (loudspeakers and microphones) as well as AD converters utilizing DC-blockage, thus, they do not provide any useful information at 0 Hz (DC).

One approach to overcome this problem is to employ extrapolation using as reference data of a frequency range below 1,5 kHz, where according to [Minnaar et al. \(2000\)](#) the group delay should be almost constant.

This method has the disadvantage of being highly dependent on the frequency range chosen and requires a lot of computation time with longer impulse responses.

Phase delay fitting This method was first proposed in [Jot et al. \(1995\)](#), it assumes that the excess phase of a HRTF is a linear function of frequency until 8 to 10 kHz. Since the excess phase component of the HRTF can be replaced with a pure delay, this delay can be calculated by fitting a linear curve on the excess-phase response between 1 kHz and 5 kHz for left and right ears and computing the difference. [Huopaniemi and Smith \(1999\)](#) proposed another frequency range, 500 Hz to 2 kHz, whereas [Minnaar et al. \(2000\)](#) states that the phase can only be linear as a function of frequency for frequencies below 1.5 kHz.

⁸Decomposition of HTRfs is treated in [3.2](#)

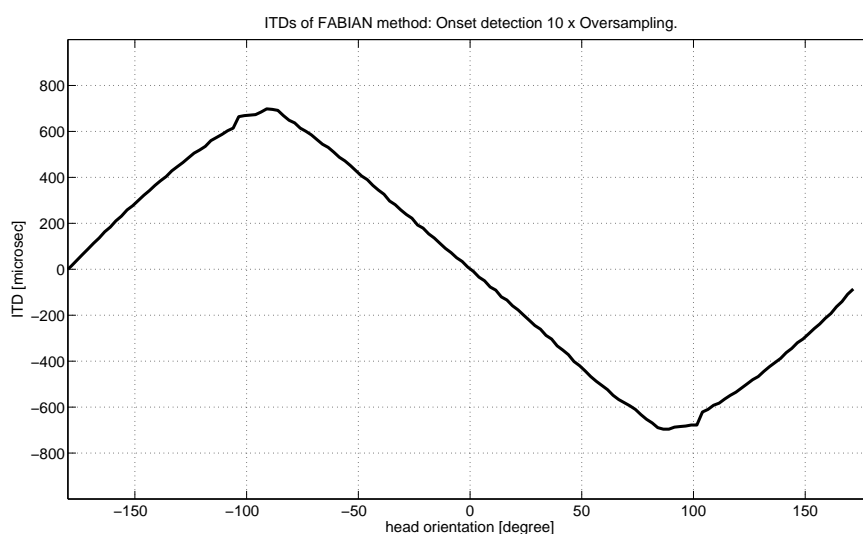


Figure 2.6.: ITD extracted using the onset detection method with 10x up-sampling, threshold -3dB. Data set: FABIAN's HRIRs (elevation: 0° , azimuth: -180° to $+180^\circ$, resolution: 1°)

The frequency dependency of this method means that the ITD obtained varies according to the frequency evaluation range. In the work of [Algazi et al. \(2001a\)](#), it is also mentioned that phase related methods are problematic because of reflections and resonances of torso and pinnae causing unpredictable phase responses on the HRTFs.

Out of the methods analyzed in [Estrella \(2010\)](#) the threshold detection method seems to be the most appropriate since it delivers estimations of ITD which are continuous functions of the angle with most kinds of data sets, the method is computationally fast and delivers values which according to [Busson et al. \(2005\)](#) are most similar to the perceptually correct ones. Note that the high frequency ITD (achieved by the threshold method) has been found to be perceptually satisfying ([Constan and Hartmann 2003](#); [Nicol 2010](#)).

Figure 2.6 shows an ITD estimation example using the onset detection method.

The performance of the threshold method compared to the perceptual ITD can be seen on figure 2.7.

2.1.3.3. ILD as localization cue

The ILD as the other cues for spatial localization is closely related to the physical structure of the human subject or HATS involved in the data set acquisition (see fig. 2.8). It is defined

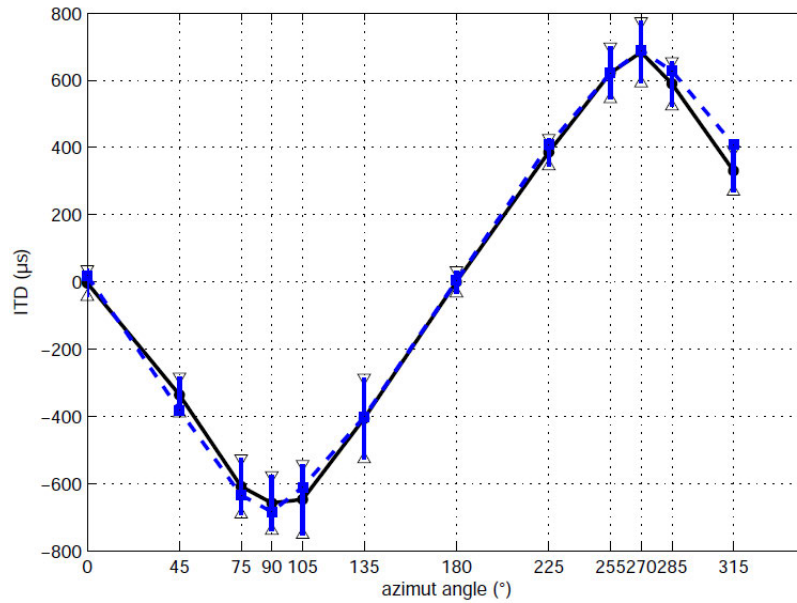


Figure 2.7.: Subjective ITD vs. ITD extracted with the Edge Detection method. Means of subject's answers and standard deviations are plotted with continuous lines. Dotted line represent the ITD estimation method. Only the horizontal plane is considered. From [Busson et al. \(2005\)](#).

as the difference of the spectrum magnitude of the left and right HRTFs. The ILD can be computed as:

$$ILD = 10 \cdot \log \left[\frac{\int_{f_1}^{f_2} |H_L(f)|^2 df}{\int_{f_1}^{f_2} |H_R(f)|^2 df} \right] \quad (2.5)$$

where $|H_{L,R}(f)|$ are the HRTF magnitude spectrum and f_1, f_2 are the integration band. The boundaries that are usually chosen are 1 kHz and 5 kHz since that is the range where the ILD plays a primary role as a localization cue ([Nicol 2010](#)).

2.1.3.4. SC as localization cue

Spectral cues are defined as salient features of the HRTF spectrum magnitude which can be interpreted by the auditory system to localize sounds. It has been demonstrated that SC convey primarily information regarding sound elevation ([Langendijk and Bronkhorst 2002](#); [Møller and Hoffmann 2006](#)).

SC are mainly located within the 4 - 16 kHz band.

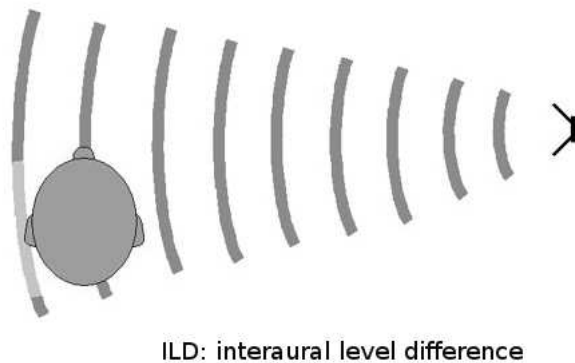


Figure 2.8.: Acoustic shadowing causing the interaural level difference

2.1.4. Dynamic binaural auralization

Rendering of binaural signals that take head movements into account are referred to as dynamic binaural synthesis. Here exchanging BRIRs in real time according to listener's head movements leads to a stable sound source localization. Dynamic binaural systems require a head-tracking system which information is used to adapt the binaural filters modifying the virtual direction of the sound source in order to keep the perception of the sound source at a fixed location in the listener's referential (Nicol 2010; Sandvad 1996).

The advantages of dynamic binaural systems are:

- Reduction of front-back confusion compared to static systems (Wenzel 1999).
- According to Nicol (2010) subjects judge spatial attributes of the sound scenes in dynamic binaural systems as having increased spatial precision, externalization, envelopment and realism as static binaural systems.

Since the binaural filters need to be interactively changed two aspects have to be considered in dynamic binaural systems:

- Latency between head movements and system interaction, which should be a minimum as possible in order to keep the binaural system latency under 60 ms⁹.
- A common practice in order to avoid audible artifacts (discontinuities) caused by the commutation of binaural filters consists in the use of Cross-fading between the current and previous signals.

⁹according to Brungart et al. (2006) latency values lower than 60 ms are likely to be adequate for most virtual audio applications, and delays of less than 30 ms are difficult to detect even in very demanding virtual auditory environments.

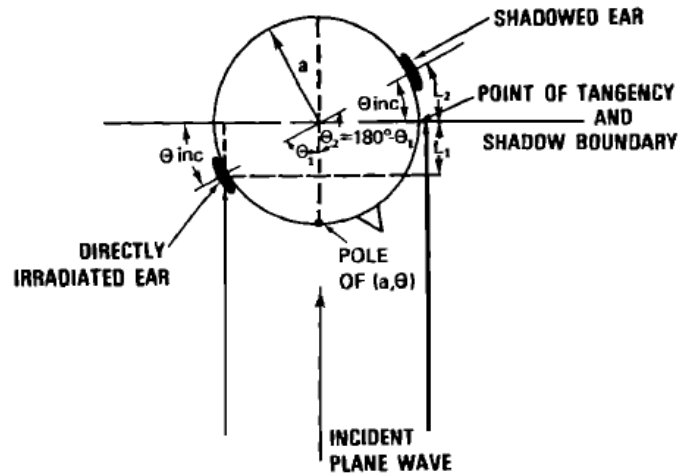


Figure 2.9.: Woodworth and Schlosberg's ITD computation method based on a spherical head model. From (Kuhn 1977).

2.2. Individualization using geometrical models

In section 1.1 the problematic of non-individualized binaural reproduction was already addressed, in this section approaches of binaural individualization are reviewed.

The need for individualized spatial cues has lead many investigators to try to synthesize head related impulse responses (HRTFs) relating it's characteristics to anthropometric parameters. Woodworth et al. (1972) developed a formula for predicting the high frequency ITD based on just one anthropometric parameter, the head radius (see fig. 2.9). This formula (eq. 2.6) takes account of the diffraction of a plane wave around the sphere¹⁰:

$$ITD = \frac{a}{c}(\sin \theta + \theta) \quad (2.6)$$

with:

a = head radius

c = speed of sound

θ = azimuth angle in [Rad] $\in -\frac{\pi}{2} < \theta < \frac{\pi}{2}$

Larcher and Jot (1999) and Savioja et al. (1999) extended formula 2.6 to include the elevation dependency of the ITD and to cover the whole horizontal and frontal planes.

¹⁰The Woodworth-Schlosberg formula (eq. 2.6) contemplates only the horizontal plane.

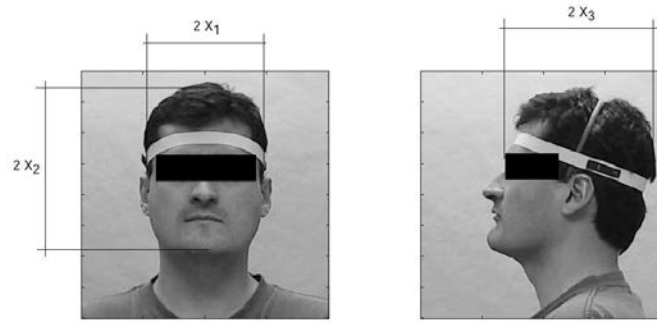


Figure 2.10.: Anthropometric measures used to find the optimal head radius in [Algazi et al. \(2001b\)](#).

The mentioned geometrical models use the the sound source direction¹¹ and the head radius as individualization parameter to synthesize the ITD. In order to apply them, a method for computing an equivalent head radius out of head dimensions is required.

2.3. Anthropometric aided individualization

Algazi developed an empirical formula to provide an optimal head radius for it's use with Woodworth's ITD model ([Algazi et al. 2001b](#)). It's equation is based on three anthropometric measures: head width, head height and head depth (X_1 , X_2 and X_3 respectively on eq. 2.7).

$$a_{opt} = W_1X_1 + W_2X_2 + W_3X_3 + b[cm] \quad (2.7)$$

with:

X_1 = head width/2

X_2 = head height/2

X_3 = head deep/2

By means of linear regression over the above mentioned head-dimensions¹² an empirical formula for predicting the optimal head radius was achieved.

$$a_{opt} = 0.51X_1 + 0.019X_2 + 0.18X_3 + 3.2[cm] \quad (2.8)$$

¹¹In data set based binaural auralization the position of the sound sources is mostly unknown.

¹²HRTF recordings conducted on 25 subjects male and female, caucasian and asian were used in this method. Least squares fitting between the measured ITD and the ITD produced by the Woodworth's formula was applied.

Algazi's anthropometric method for finding an optimal head-radius represent an enhancement on the applicability of the geometric models and at the same time an interesting approach for relating human-head's dimensions to the individualized ITD.

However, the position of the sound sources has to be known in order to apply those geometric individualization models. In binaural data set based auralization the position of the sound sources is rarely known, thus, the above mentioned methods are not suitable for this purpose; nevertheless, the procedure of relating anthropometric head measures to the interaural time difference serve as inspiration for a new individualization approach.

3. Methods

3.1. Chapter's overview

The individualization model presented in the previous chapter assumes the separate processing of time and spectral characteristics of binaural data sets as a valid approach. This assumption will be theoretically founded in section 3.2.

The spectral characteristics in the presented model are processed in a fast convolution engine that uses minimum-phase IRs instead of the original HRIRs. Section 3.3 discusses the method chosen for the extraction of those minimum-phase IRs.

As explained before, the proposed individualization model (figure 1.2) reinserts the individualized ITD in form of a time delay between the left and right ears audio streams.

Interpolation between audio samples is needed in order to provide delays shorter than one sample (22 μ s for 44100 kHz). In this sense, the concepts of fractional delay and band limited interpolation are discussed on section 3.4.

Finally, a resume of this chapter is presented in section 3.5.

3.2. Separate processing of time and spectral characteristics

Head-related transfer functions can be treated as linear time-invariant (LTI) systems. In LTI system theory the complex frequency response of a transfer function can also be expressed in terms of magnitude response and phase response. In the case of HRTFs, the phase can be split in a minimum phase component and an excess-phase component.

$$H(j\omega) = |H(\omega)| \cdot e^{j\Phi_{min}(\omega)} \cdot e^{j\Phi_{excess}(\omega)} \quad (3.1)$$

The frequency dependent excess-phase component can also be decomposed into linear-phase and all-pass components.

$$H(j\omega) = |H(\omega)| \cdot e^{j\Phi_{min}(\omega)} \cdot e^{j\Phi_{lin}(\omega)} \cdot e^{j\Phi_{allpass}(\omega)} \quad (3.2)$$

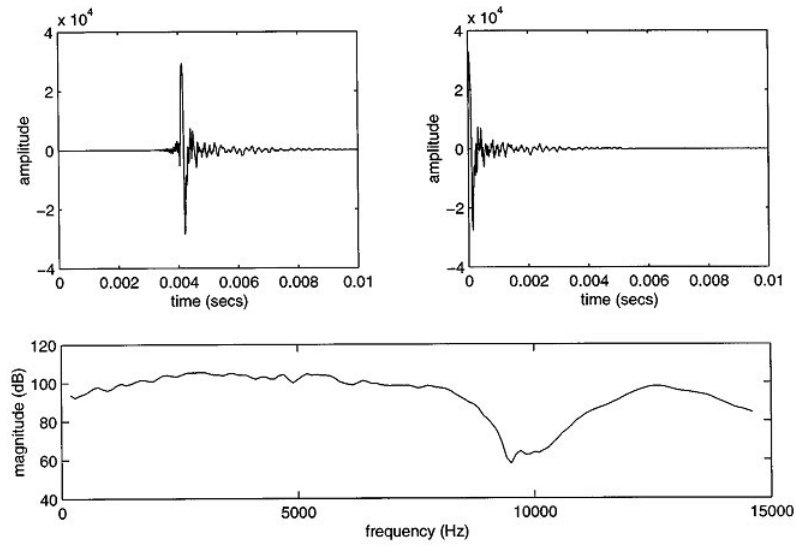


Figure 3.1.: (up) HRTFs with and without phase component. (down) frequency response of both HRTFs. Source (Kulkarni et al. 1999)

Since the sensitivity to phase spectra on humans is low (Preis 1982), the all-pass component can be neglected without disturbing the spatial perception (Minnaar et al. 1999) as has been shown for HRIRs that the contained all-pass component is inaudible for most directions of sound incidence.

$$H(j\omega) = |H(\omega)| \cdot e^{j\phi_{min}(\omega)} \cdot e^{j\phi_{in}(\omega)} \quad (3.3)$$

Moreover, the linear-phase component on equation 3.3 can be replaced by a time delay without audible consequences as long as it adequately approximates the ITD (Kulkarni et al. 1999).

A graphical example of this temporal-spectral separation is shown on Figure 3.1 where two IRs, original, and minimum- phase have the same frequency response.

3.3. Extraction of minimum-phase impulse responses

Two methods for the extraction of minimum phase IRs were also discussed in Estrella (2010):

- Hilbert transformation based method (Oppenheim et al. 1999) also known as Kolmogorov method of spectral factorization. Obtained using Matlab's™ `rceps` function.

Matlab's™ algorithm finds first the real cepstrum of the input signal as:

```
y = real(ifft(log(abs(fft(x)))));
```

The minimum phase impulse response is computed after windowing in the cepstral domain.

```
window = [1;2*ones(n/2-1,1);ones(1-rem(n,2),1);zeros(n/2-1,1)];
```

```
min_phase = real(ifft(exp(fft(window.*y))));
```

- Threshold method, consists in extracting the impulse response starting at the ITD detection spot (using the onset detection as ITD estimation method). For better accuracy the ITD estimation and the decomposition are realized with 10x up-sampled HRIRs.

Two ABX listening tests were conducted in order to assess if perceptual differences between the extracted minimum phase impulse responses and the original HRIRs can be detected. ABX tests allow to assess whether discrimination between two samples is possible (performance better than chance).

In these listening tests the H_0 hypothesis ("no audible difference existing") was the H_0 research hypothesis. As the H_0 cannot be proved directly in inferential statistic tests, instead, one tries to neglect a rather small-effect-size H_1 , indirectly supporting the H_0 if a small effect can be shown to be absent (Murphy and Myers 1999).

The same 10 subjects participated on both tests. On the first listening test, Hilbert minimum phase IRs vs. original HRIRs, each of them had to listen 14 times to each stimulus, resulting in 42 decisions per subject.

Being limited to this reasonable sample size, the existence of at least 75% detection rate per individual on 5% significance level with a test power of 95% could be tested ¹³.

The left and right BRIRs of three rooms: large, medium and small with respectively 1.2, 1.8 and 2 seconds reverberation time, recorded with the HATS FABIAN at 0° azimuth, 0° elevation were used.

The test consisted in the comparison between the original BRIRs and the Hilbert minimum-phase BRIRs convolved with a short piece of drum solo as content. This stimulus was chosen because it contains many transients, which are supposed to ease the detection of time domain alterations.

The hypothesis H_0 would be rejected if at least 27 of the 42 decisions were correct. Figure 3.2 shows the results of this test. It can clearly be seen that all participants could easily

¹³The test characteristics were computed using Burstein's approximation formulas (Burstein 1988).

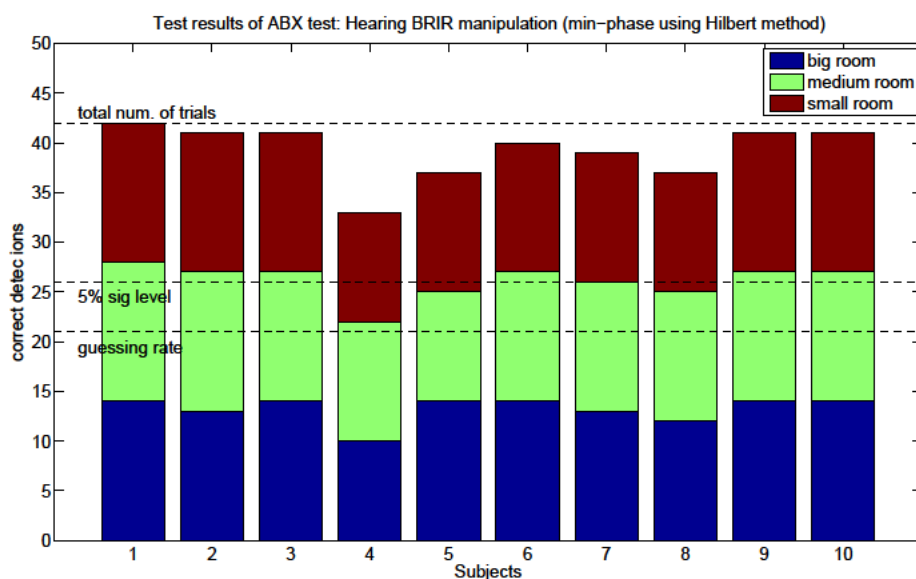


Figure 3.2.: Results of ABX listening test of minimum-phase IRs (Hilbert method) vs. original impulse responses.

recognize the Hilbert minimum phase IRs from the original. For half of the subjects the detection rate was above 97% and never sank below 78.4%.

In the second ABX listening test, minimum phase IRs obtained with the threshold method were compared with the original HRIRs. The hypothesis H_0 would have to be rejected if at least 31 of the 48 decisions were correct. Two contents: male speaker and noise bursts, were convolved with the original and the manipulated HRIRs.

The task consisted of identifying whether the reference corresponded either to the auralization using the original HRIR or the manipulated HRIR on 48 decisions (8 auralization directions x 2 contents x 3 runs). For every decision the direction of sound incidence and the audio content were randomized.

To test the performance of the onset method at large ITD differences, HRIRs of IRCAM's public database at 8 auralization directions $[90^\circ, 0^\circ]$, $[90^\circ, 45^\circ]$, $[90^\circ, -45^\circ]$, $[-90^\circ, 0^\circ]$, $[-90^\circ, 45^\circ]$, $[-90^\circ, -45^\circ]$, $[45^\circ, 45^\circ]$, $[-45^\circ, -45^\circ]$, were selected and manipulated as follows:

- 10x Upsampling.
- ITD detection and extraction (shortening of the impulse response).
- Downsampling to the original samplerate.

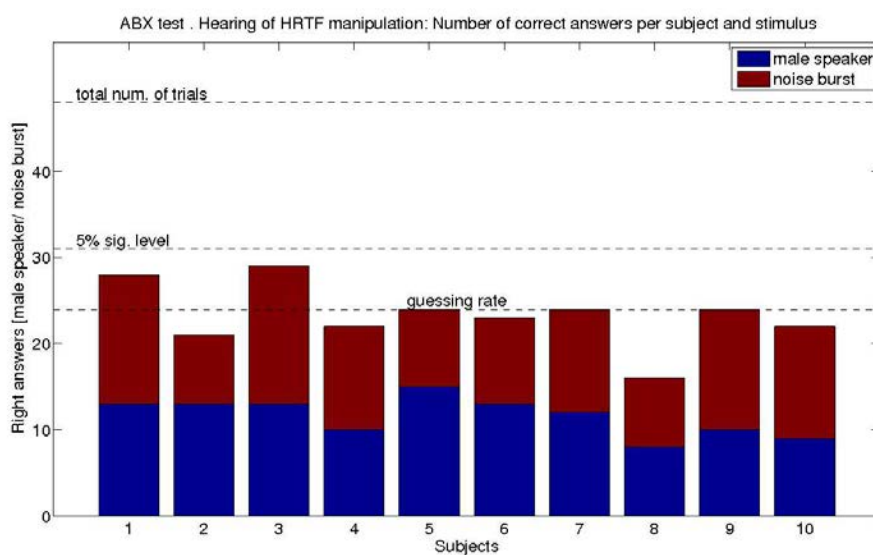


Figure 3.3.: Results of ABX listening test of minimum-phase IRs (extracted with the onset detection method) vs. original impulse responses.

- Convolution with audio content.
- Upsampling with a factor of 10.
- Zero padding on one of the IRs to an equivalent ITD.
- Downsampling to the original samplerate.

Figure 3.3 shows the results of this test. None of the subjects were able to reach the 31 correct decisions. **This approach can practically be considered as not having obvious audible consequences.**

Figure 3.4 shows the extraction of the minimum-phase impulse response with the threshold method on a selected contralateral HRIR.

3.4. Fractional delay filters

For accurate individualization of the ITD, the implementation of fractional delays (FD) should be considered.¹⁴

¹⁴Välimäki and Laakso (2000) defines fractional delay as a delay that is non integer multiple of the sample interval (assuming uniform sampling).

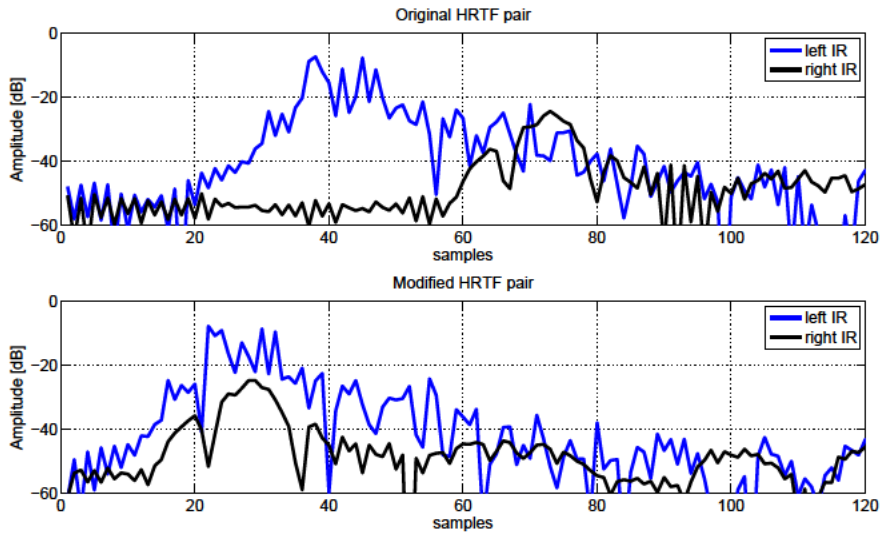


Figure 3.4.: Extraction of quasi minimum-phase impulse responses with the onset detection method. Note that the envelope has slightly changed due to manipulation. It was these kind of differences that were assessed for audibility in the second listening test of section 3.3.

In Välimäki et al. (1996) the concepts of fractional delay are explained. Let us summarize from that paper the most relevant aspects for our work. First, let us consider the case of delaying a band-limited continuous-time signal $x_c(t)$ by an amount t_D . A continuous-time delay can be defined as an operator L_c which yields its output as:

$$y_c = L_c\{x_c(t)\} = x_c(t - t_D) \quad (3.4)$$

Converting eq. 3.4 into discrete time by sampling at time instants $t = nT$ ($n \in \mathbb{N}$ and T is the sampling period) we obtain:

$$y(n) = L\{x(n)\} = x(n - D) \quad (3.5)$$

where D is a positive real number that can be split into integer and fractional part as:

$$D = \text{Int}(D) + d \quad (3.6)$$

However we can see that eq. 3.5 is only meaningful for integer values of D . But for non integer values of D the output would lie between two samples, thus, appropriate values on the sampling grid must be found.

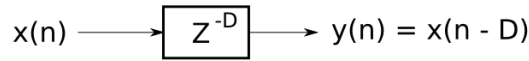


Figure 3.5.: Basic discrete delay system.

The solution can be obtained by first reconstructing the continuous band-limited signal and then resampling it after shifting, thus, the problem can be solved treating the delay as a resampling process (Välimäki and Laakso 2000).

Let us now view the delay system of eq. 3.5 (fig 3.5) in the Z plane.

$$H_{id}(z) = \frac{Y(z)}{X(z)} = \frac{z^{-D}X(z)}{X(z)} = z^{-D} \quad (3.7)$$

Where $X(z)$ and $Y(z)$ are the Z transform of $x(n)$ and $y(n)$ respectively. H_{id} stands for *ideal* response. Formulating eq. 3.7 in the frequency domain ($z = e^{j\omega}$) we have:

$$H_{id}(e^{j\omega}) = e^{-j\omega D} \quad (3.8)$$

where $w = 2\pi f$ is the normalized angular frequency.

The complex function expressed in terms of magnitude and phase has the form:

$$|H_{id}(e^{j\omega})| = 1 \quad (3.9)$$

$$\arg\{H_{id}(e^{j\omega})\} = \Theta_{id}(\omega) = -D\omega \quad (3.10)$$

Since *group delay* and *phase delay* are both measures of the delay in a the system and when the phase is linear both yield identical results:

$$\tau_{p,id}(\omega) = D \quad (3.11)$$

$$\tau_{g,id}(\omega) = D \quad (3.12)$$

Now for a band-limited signal, the implementation of a constant delay can be viewed as the approximation of an ideal linear phase all-pass filter with unity gain and a constant group delay of value D . The corresponding impulse response is obtained via inverse discrete-time Fourier transform:

$$h(n) = \frac{1}{2\pi} \int_{-\pi}^{\pi} H(e^{j\omega}) e^{j\omega n} d\omega \quad (3.13)$$

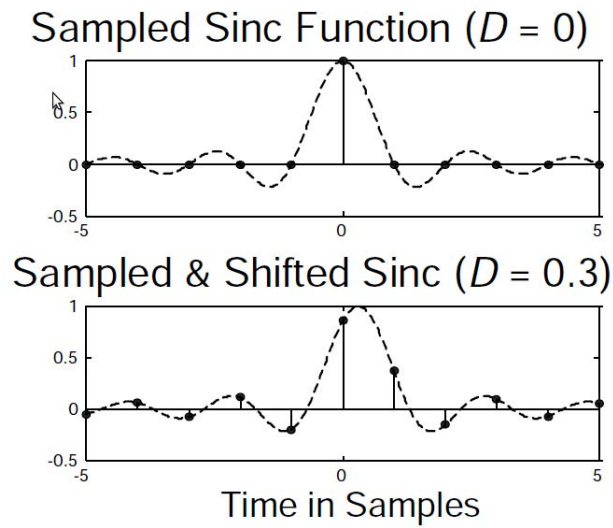


Figure 3.6.: Ideal fractional delay approximation. Up: delay D is integer, sampling occurs at zero crossings. Down: delay D is non integer, sampling occurs between zero crossings. Infinite length impulse response is required in the ideal case. From [Välimäki and Laakso \(2000\)](#).

Integrating eq. 3.8 into 3.13 the solution for the ideal impulse response is obtained:

$$h_{id}(n) = \frac{\sin[\pi(n-D)]}{\pi(n-D)} = \text{sinc}(n-D) \quad (3.14)$$

which corresponds to the shifted *sinc* function as the one in figure 3.6.

Equation A.5 gives solution for the original problem of finding the value between samples. Note that in the ideal case it should be spread over all the discrete time signal values weighted by the appropriate values of the *sinc* function.

Summarizing, the ideal solution for fractional delaying is *sinc* interpolation. However, since the shifted *sinc* impulse response cannot be infinite, the length of the filter plays an important role in the accuracy of the interpolation. This aspect will be discussed in Chapter 4 from the point of view of the practical implementation.

3.5. Chapter's resume

In this chapter the methods to implement in the ITD individualization model were reviewed. As stated in chapter 1 the proposed individualization method requires separate processing

of temporal and spectral cues. This requirement was assessed under the system theory point of view.

It was also explained that the binaural data set must be prepared for its use with our application. The ITD as a function of head position $ITD(\theta, \phi)$, where θ and ϕ are azimuth and elevation, has to be estimated and collected. For this purpose the *onset detection* method has been found to be most suitable. Minimum phase impulse responses will be extracted using the onset method for detecting their starting point, as this method does not introduce artifacts in the auralization process.

For optimal performance the variable delay line (VDL) that reinserts the customized ITD should be capable of delivering fractional delays. It has been demonstrated that the ideal interpolation method is re-sampling using band-limited *sinc* interpolation, as this method represents an enhancement of the sampling theorem.

4. Implementation

This chapter explains implementation related topics. Section 4.1 describes the steps that are followed to prepare the data set to work with our application.

4.1. Data set pre-processing

As explained in the previous chapter special preparation of the binaural data-sets is required to work with our application. The ITD of the data set to process has to be collected and minimum phase IRs extracted from it.

These tasks were accomplished using a Matlab™ script consisting in the following steps:

- Read HRIRs (stored as a *.wav* files),
- Up-sample to a factor of 10 (ie. 44100 Hz to 441000 Hz),
- Find indexes of the given threshold (ie. -35 dB from peak) in left and right onsets,
- Write matrix of detected indexes for left and right IRs,
- Compute ITD as the difference of left and right (up-sampled) onset indexes.
- Convert ITD in μs
- Find the maximal time of flight
- Find the new length of the IRs as: $new_length = size_IR - max_time_of_flight$
- Extract minimum phase IR of length new_length starting at the onset indexes.
- Down sample minimum phase IRs in a factor of 10.
- Store minimum phase IRs
- Store ITD matrix in a machine readable format (*.txt*).
- Generate a description of the modified dataset (start and end angles, angular resolution in degree for azimuth and elevation) in a machine readable format (*.xml*)

Code from the above mentioned steps can be found in Appendix B.

4.2. Functional requirements

Guidelines for the development of the ITD individualizer (ITD-I) are taken from the functional requirements of the complete system. These can be listed as follows:

- Real-time audio. Since dynamic binaural synthesis consists in essence of interactive processes, the application must be capable of operating in real-time at low latencies.
- Low signal to noise ratio (SNR).
- Bandwidth covering the frequency range of human hearing.
- Compatibility with the *fwonder* project. Modular integration with the real-time convolution engine should be granted.
- Fractional delays. According to [Mills \(1958\)](#) the least noticeable difference (lsd) of ITD changes can be as low as $10 \mu\text{s}$. Since at a samplerate of 44100 Hz the period is already $22.7 \mu\text{s}$, the ITD-I has to be capable of delivering finer resolutions, that means fractional delays.
- Support of configuration scripts. To facilitate the start of the software with customized parameters, easy to configure parameter-scripting should be possible.
- Command-line configuration. Another technique to provide pre-configuration is to use command-line instructions and arguments.
- Real-time control. ITD individualization should be possible not only with pre-configured settings, but also in real time using a graphical user interface (GUI), run-time hotkeys and OSC messages.

4.3. Software components

This section reviews the main software components employed in the application. It should be first stated that the application was written in C++ since several efficient libraries of methods are available in this programming language.

4.3.1. Low latency high priority audio thread: The JACK Audio application programming interface (API)

One of the requirements listed in the previous section is compatibility with the *fwonder* project. *fwonder* was written for Linux and uses JACK as real time low latency audio system. JACK's most important functional characteristics are:

- Low latency audio recording and reproduction,
- Sending audio between applications,
- Sharing an audio interface (soundcard),
- JACK guaranties sample accurate synchrony between applications,
- Open source software, thus free available and modifiable.

Usage

In its most typical use, applications working with JACK do not start a new thread to execute their audio processes, instead they provide a callback function to the JACK server (thus, acting like a plugin or client) which takes care of calling that function and keeping synchrony between all clients.

From the developer point of view this are the steps required to run a JACK client using the C++ API:

1. Include JACK's header file

Necessary for the compiler in order to include JACK's functions, type definitions and enumerators in the application.

```
#include <jack/jack.h>
```

2. Opening a new client in the server

jack_client_open has to be called to create a new client, requisites are:

- a unique name for reference purposes,
- starting options like server name to register the client to, or session identification, etc,
- a status pointer used by JACK to return client information.


```
jack_client_t * jack_client_open (  const char*  client_name ,
                                   jack_options_t  options ,
                                   jack_status_t *  status  )
```

3. Register the client's ports

The pointers (memory addresses) to the buffers that will be used for collecting input and output should be registered in the JACK server. The parameters to register are:

- the client whose port(s) we are registering
- the name of the port
- the type of port, audio or midi (ie. *JACK_DEFAULT_AUDIO_TYPE*)
- the flags indicating if the port is input, output, physical, monitor or terminal

```
jack_port_t *  jack_port_register (  jack_client_t *  client ,
                                   const char*  port_name,
                                   const char*  port_type ,
                                   unsigned long  flags ,
                                   unsigned long  buffer_size  )
```

4. Register a process callback-function

Here we tell JACK what audio processing routine it should call. The parameters to use are:

- the name of the client
- the name of the callback-function
- a pointer for passing our own arguments (if any) to the function

```
int  jack_set_process_callback (  jack_client_t *  client ,
                                JackProcessCallback  process_callback ,
                                void*  arg  )
```

The callback-function should have the form:

```
int process (jack_nframes_t nframes, void *arg)
{
    –operations–
}
```

and it is here where the audio process has to be executed. For us, real-time fractional delay using *sinc* interpolation. Further in this chapter the algorithm developed to dynamically reinsert the individualized ITD as a VDL will be explained in detail.

5. Activate the JACK client

By calling this method the server starts processing audio. The only parameter required is the name of the client that was previously registered.

```
int jack_activate ( jack_client_t * client )
```

4.3.2. Delay-lines based on sample rate conversion (SRC)

In section 3.4 we saw that the optimal approach to achieve fractional delays in discrete-time systems is to reconstruct the audio signal and resample it after shifting the sampling filter (*sinc* function), thus, using Nyquist sampling theorem. The implementation of SRC to achieve delay lines will be explained in this section with the aid of an example from [Wefers \(2007\)](#).

Let us consider a time discrete signal $s(n)$ with a sample period T , let us also define $f(n)$ as a function that for every sample in $s(n)$ assigns the point in time of its reproduction. For a non delayed $s(n)$, we have $f(n) = nT$.

Let us now consider two delays of $s(n)$:

$$s_1(n) = s(n - n_1) \Rightarrow f_1(n) = (n + n_1)T$$

$$s_2(n) = s(n - n_2) \Rightarrow f_2(n) = (n + n_2)T$$

Now, during the time interval $[n_aT, n_bT]$ ($n_a, n_b \in \mathbb{N}$, $n_a \leq n_b$) the delay changes from n_1 to n_2 samples. If k_a^1 denotes the sample of $s_1(n)$ that is reproduced at the beginning of the interval $[n_aT, n_bT]$, then:

$$f(k_a^1) = (k_a^1 + n_1)T \stackrel{!}{=} n_aT \Rightarrow k_a^1 = n_a - n_1$$

And for both signals at the beginning and ending of the interval it would be:

$$\begin{aligned} k_a^1 &= n_a - n_1 & k_b^1 &= n_b - n_1 \\ k_a^2 &= n_a - n_2 & k_b^2 &= n_b - n_2 \end{aligned}$$

At the beginning of the interval $[n_a T, n_b T]$ (at the time $n_a T$) the delay has not changed yet, hence, the sample to reproduce is k_a^1 . On the other hand at the end of the interval (at the time $n_b T$) the new delay n_2 should be reached, meaning that the sample k_b^2 should be reproduced at this point. Since $n_1 \neq n_2$ the only way to meet this condition is to stretch or squeeze the signal by altering its sampling rate by a factor of $r \in \mathbb{R}^+$.

Let us consider the modified function $\tilde{s}(n)$ as a variant of $s(n)$ with a sampling period rT . If $\tilde{s}(n)$ is delayed by a time \tilde{t} and $\tilde{f}(n)$ corresponds to $f(n)$ we have:

$$\tilde{s}(n) = s(n) \text{ and } \tilde{f} = (n - \tilde{t})rT$$

Now the idea is to stretch or squeeze $\tilde{s}(n)$ and adjust \tilde{t} to meet the conditions:

$$\begin{aligned} i) \quad & \tilde{f}(k_a^1) = n_a T \\ ii) \quad & \tilde{f}(k_b^2) = n_b T \end{aligned}$$

From $i)$ \tilde{t} can be isolated:

$$\begin{aligned} \tilde{f}(k_a^1) &= (k_a^1 - \tilde{t})rT \stackrel{!}{=} n_a T \Rightarrow k_a^1 - \tilde{t} = \frac{n_a}{r} \\ \Rightarrow \tilde{t} &= k_a^1 - \frac{n_a}{r} \end{aligned}$$

applying this in $ii)$ the sampling rate conversion ratio r can be obtained:

$$\begin{aligned} \tilde{f}(k_b^2) &= (k_b^2 - k_a^1 - \frac{n_a}{r}) \stackrel{!}{=} n_a T \\ r(k_b^2 - k_a^1) &= n_b - n_a \Rightarrow r = \frac{n_b - n_a}{k_b^2 - k_a^1} \end{aligned}$$

Inserting the definitions of k_a^1 and k_b^2 we obtain:

$$r = \frac{n_b - n_a}{n_b - n_2 - n_a + n_1} = \frac{n_b - n_a}{n_b - n_a + n_1 - n_2}$$

If the interval of sample rate conversion is kept constant $N = n_b - n_a$, and we express the delay as $\Delta = n_2 - n_1$ the samplerate conversion ratio is:

$$r = \frac{N}{N - \Delta} \tag{4.1}$$

where $N, \Delta \in \mathbb{N}$

A decrease in delay is indicated by $\Delta < 0 \Rightarrow r > 1$ on the other hand an increase in delay means $\Delta > 0 \Rightarrow r < 1$.

Note that $N = \Delta$ should be avoided to prevent division by zero. As well as $|\Delta| \leq N$ in order to keep $r \in \mathbb{R}^+$.

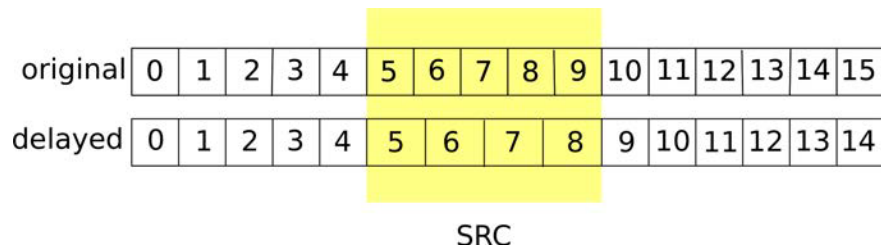


Figure 4.1.: Time stretching for achieving one sample delay ($22\mu s$). Note the use of sample rate conversion at the stretching region.

Example

Let us suppose the simple case of delaying one sample of an audio stream as presented on figure 4.1. Here, the sample rate conversion ratio to apply in the yellow region would be:

$$r = \frac{N}{N - \Delta} = \frac{\text{input samples}}{\text{output samples}} \quad (4.2)$$

In our example, the amount of input samples in the SRC is $N = 5$, and the amount of output samples is $N - \Delta = 4$, so, the conversion ratio equals:

$$r = \frac{5}{4} = 1.25$$

Now, if we want to delay the signal on 2 more samples we can either use the SRC with $r = 1.25$ two times, or change the conversion ratio to:

$$r = \frac{5}{3} = 1.66667$$

and use it once. As long as no further delay is needed the SRC should be driven with a conversion ratio $r=1$.

4.3.3. The libamplerate API

Since the realization of a SRC software is very complex and out of the scope of this work, an open source library, [Libamplerate SRC API](#) was used. This specific library was chosen because:

- SRC on audio streams is possible,
- *Libamplerate* implements *sinc* interpolation (among other interpolators) in three bandwidth variants: 97% , 90% and 80% ,
- The API has been maintained and actively developed since 2002,

- SNR reaches 97dB for all its *sinc* interpolators.

The `libsamplerate` API has three operation modes: simple, for static audio files; full and callback mode for audio streams. We used the full API since it allows more operation parameters to be specified.

Usage

To use `libsamplerate`'s functions it is required to include its header file:

```
#include <samplerate.h>
```

The SRC needs to be initialized using following method:

```
SRC_STATE* src_new(int converter_type, int channels, int *error)
```

This function returns a pointer to a SRC object and requires as parameters the converter type, the number of audio channels and an error pointer that the converter will fill if errors are encountered. The library can work with five converters:

- **SRC_SINC_BEST_QUALITY** A *sinc* SRC with a bandwidth of 97%
(ie. $(44100 \div 2) \cdot 0.97 = 21388.5\text{Hz}$)
- **SRC_SINC_MEDIUM_QUALITY** A *sinc* SRC with a bandwidth of 90%
- **SRC_SINC_FASTEST** A *sinc* SRC with a bandwidth of 80%
- **SRC_ZERO_ORDER_HOLD** A very fast converter with poor quality
- **SRC_LINEAR** A very fast converter based on linear interpolation

Once a SRC handle is created the function `src_process` needs to be called every time there is a process to execute.

```
int src_process (SRC_STATE *state, SRC_DATA *data)
```

This function uses the data of an object of type `SRC_DATA` containing following information:

data_in	A pointer to the input data samples
input_frames	The number of frames of data pointed to by data_in
data_out	A pointer to the output data samples
output_frames	Maximum number of frames pointer to by data_out
src_ratio	Equal to $\text{output_sample_rate} / \text{input_sample_rate}$
end_of_input	Equal to 0 if more input data is available and 1 otherwise

The functions returns the number of output samples generated as well as the number of input samples used in the conversion (using members of the *SRC_DATA struct*). This information should be used to manage the audio buffers in order to assure glitch free coherence of the audio streams. In our application it is extremely important to guarantee a steady size of output samples to fill the output buffer. Remaining samples are reinserted in the audio stream with the aid of a ring buffer.

Every time there is a new delay to reach, a properly computed new conversion ratio should be set using:

```
int src_set_ratio (SRC_STATE *state, double new_ratio) ;
```

4.3.4. OSC control: The liblo API

Another functional requirement of the application is to receive open sound control (OSC) messages to update the individualized ITD according to the user's head position and to customize the ITD scaling factor in real time.

OSC was developed at the Center for New Music and Audio Technology (CNMAT) of the University of California at Berkeley. One of its goals was to create a protocol among multimedia devices that is optimized for modern networking technology (Wright 2005). Another goal was to create a protocol "open" to the generality of messages. Unlike the limited messaging possibilities of MIDI, OSC can handle strings, numbers as 32-bit floating point, 32-bit integer or 64-bit double precision among other formats, it also handles boolean values, binary "blobs" and more (Liblo OSC API).

OSC is also called "open" because no standard messaging parameters are established, thus, allowing the implementer to decide the nature and organization characteristics.

In our project OSC was implemented using the [Liblo OSC API](#), an open source (GNU LGPL) very easy to use and well documented API.

Usage

To use Liblo in C++ there are basically 5 steps to follow:

1. Include the header files:

```
#include "lo/lo.h"
```

2. Create a OSC server

Defining a port and a function that will be called in the event of an error being raised:

```
lo_server_thread lo_server_thread_new (const char* port, lo_err_handler err_h)
```

3. Register methods for parsing OSC messages

The functions to be called on the occurrence of a given kind of message are passed to the OSC server thread using the *lo_server_thread_add_method* function.

```
lo_method lo_server_thread_add_method( lo_server_thread st,
                                       const char* path,
                                       const char* typespec,
                                       lo_method_handler h,
                                       void* user_data )
```

The parameters are:

- st The server thread the method is to be added to.
- path The OSC path to register the method to. If NULL is passed the method will match all paths.
- typespec The type specification the method accepts. Incoming messages with similar types (e.g. ones with numerical types in the same position) will be coerced to the *typespec* given here.
- h The method handler callback function that will be called if a matching message is received.
- user_data A value that will be passed to the callback function h.

4. Start the OSC server

Once configured the OSC server can start receiving messages. The function *lo_server_thread_start* performs this task using the registered thread as parameter.

```
int lo_server_thread_start ( lo_server_thread st )
```

5. Stop the OSC server

If the server is not required anymore or if the application is being closed, the thread should be stopped and the reserved memory released. The Liblo API provides two functions for these purposes, both take as argument the OSC object thread:

```
int lo_server_thread_stop ( lo_server_thread st )
```

and

```
void lo_server_thread_free ( lo_server_thread st )
```

4.3.5. XML script parsing: The libxml++-2.x API

Another requirement of our software is to implement parsing of start-up configuration scripts. The extensible markup language (XML) is for that purpose very appropriate since, for the amount of parameters we are using, it can as well be easily generated an/or edited per hand or automatically using a computer.

A XML file contains *root nodes*, *child nodes* and their *elements* (attributes, parameters). The procedure to parse the contents of the parameters is to compare strings iteratively to first find a given node and then assign its content to a receptacle (ie. a member of a class-object). The **Libxml++** API, also an open source library ([GNU LGPL](#)) contains methods to facilitate the navigation in XML files (*Document* :: *get_root_node()*, *Node* :: *get_children()* among other functions).

The ITD-I parses three XML files at start-up:

- Application configuration file. Containing relevant information regarding OSC port, JACK-client name to adopt, SRC method to apply, etc. Table 4.1 presents a detail of the nodes and their configuration attributes.
- Modified BRIR information file. Containing information about the angular ranges and resolution of the data set.

- fwonder configuration file. Some operation parameters from fwonder are read in the ITD-I to assure coherence between both softwares. These are: The path to the data set in use, and the JACK name, which is used to automatically connect to fwonder's output ports to the ITD-I input ports.

The first configuration file is explained in the next section, while the other two are referred in the appendix.

Application configuration file

The ITD-I needs configuration at start-up, the parameters required are reviewed in this section in detail.

```

<?xml version="1.0"?>
<!DOCTYPE stretcher_config PUBLIC "" "stretcher_config.dtd">

< individualizer_config >
  <jack name="ITD_stretcher"/>
  <path fwonder_config_file="/home/jorgose/configs/fwonder_config.xml"/>
  <config OSC_listen_port="58800" source_number="1" SRC_modus="2" processing="4"
    scale="1" user_tragus="0" data_tragus="148"/>
</ individualizer_config >
```

Root node	Child node	Attribute
jack	name	Client name to be used when working with JACK
path	fwonder_config_file	The path of the configuration file that fwonder uses.
config	OSC_listening_port	Number of the OSC port to "listen" to.
	source_number	Specify the source number that fwonder uses.
	SRC_modus	One of the 5 SRC modus explained in section 4.3.2.
	processing	Smooths the squeezing and stretching of new delays to a given amount of processing chunks.
	scale	ITD scaling factor.
	user_tragus	Intertragus distance of the user.0 means use scaling factor instead of anthropometric formula. Section 5.3 refers this use in detail
	data_tragus	Intertragus distance of the dataset.

Table 4.1.: Starting parameters of the configuration file of the ITD individualizer.

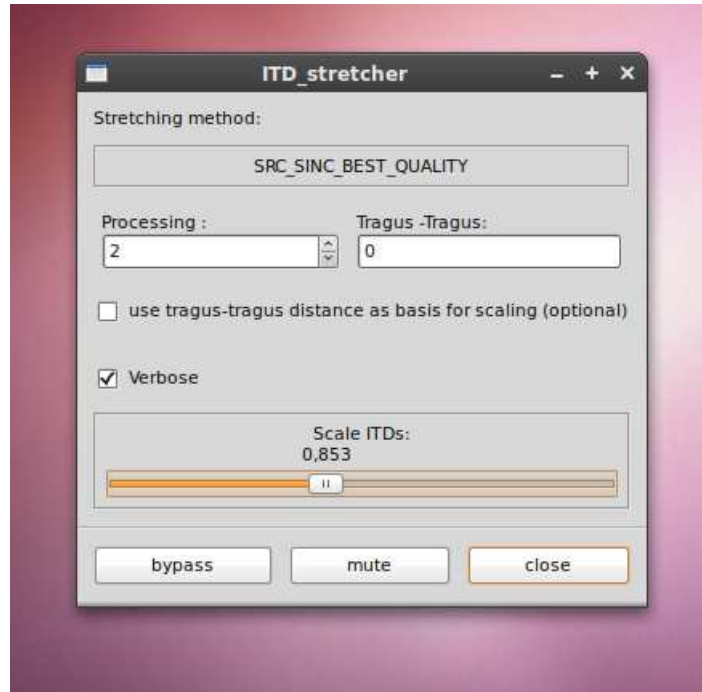


Figure 4.2.: Graphical user interface of the ITD individualizer developed using GTK+2.2

4.3.6. GUI control: The GTK+2.0 Project

In order to provide the end-user with intuitive manipulation of the ITD, a graphical user interface was developed using the GTK+ project. The GTK+ toolkit was chosen because it is an open source project (licensed under the [GNU LGPL](#)), it also has a very accurately documented API, and a big supporting community of developers.

Figure 4.2 shows the developed GUI. The GUI offers:

- A space for displaying text messages.
- Smoothing the change of one delay to another by increasing the amount of processing chunks to achieve a new delay.
- ITD scaling factors using anthropometry (see section 5.3).
- Enable verbose output with messages about the actual ITD or other events (mute, bypass, scaling factor)

- A horizontal fader scales the ITD in real time with the factor displayed on top of it. The fader ranges from 0.000 (no ITD) to 2.000 (twice as much as the ITD in the data set).
- Bypass, here the output ports reflects input ports exactly.
- Mute, output ports buffers are filled with 0.

4.4. Flowchart of the audio process

Figure 4.3 shows the flowchart of the callback function that JACK processes in its audio thread. This function is executed in real-time for every incoming audio buffer and it is here where the ITD is individualized and reinserted in the audio path. In the following the numbered blocks of the flowchart are explained:

- 1 When the function is called the pointers to the input and output buffers have to be defined, these memory addresses are changed by JACK every time the function is called.
The approach used in our application is to only affect one ear's audio stream with positive and negative delays to synthesize the individualized ITD. Since positive delays imply more output as input samples, ring buffers initialized with zeros are necessary to provide the system some headroom. Therefore the incoming left and right ears' samples are both inserted into ring buffers.
- 2 Every new delay (ITD) can only be updated after the previous delay has been reached. The user can decide how many processing blocks does a delaying task takes to complete. This is controlled with the *smoothing_size* parameter. In case of no delay update the SRC keeps processing conversions using the already computed SRC ratio.
- 3 A counter for the delay-smoothing feature is incremented every processing block.
- 4 When the amount of processing blocks has completed a flag is set scheduling the computation of a new delay.
- 5 The SRC converter is called using the parameters explained in section 4.3.3. In case of a new SRC ratio the *change_ratio* flag is passed to the SRC to make use of the *src_set_ratio()* method.
- 6 The *change_ratio* flag should only be set at every new delay computation, therefore it has to be cleared after its use.

4. Implementation

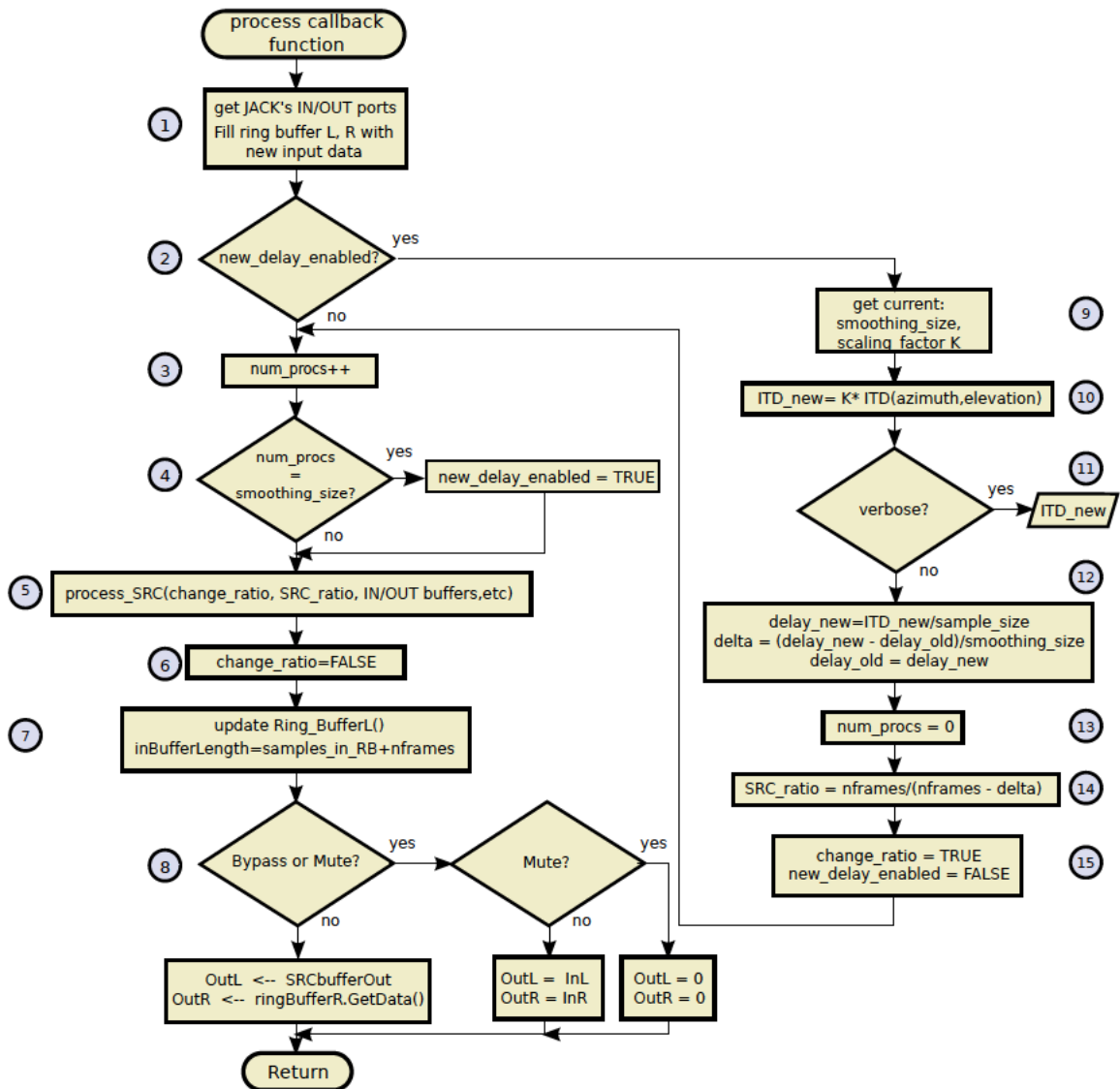


Figure 4.3.: Flowchart of the callback function that manages time stretching.

- 7 Since we provide the SRC with more samples as needed (in order to be capable of big delay changes), the unused samples have to be reinserted into the ring buffer. We have to take account of the amount of samples in the ring buffer to indicate the SRC how many samples it may use as input.
- 8 Now that the SRC has delivered the processed samples, they have to be written in the output buffers JACK gave us. At this point the algorithm checks if mute or bypass flags are set. In that case we write zeros or copy the input samples in those buffers respectively. Note that even if the ITD-I is muted or bypassed the output stream containing the individualized ITD is still computed, thus, returning to normal operation is fast and easy.
- 9 In case of delay update the algorithm gathers the current scaling factor and *smoothing_size* to compute the parameters of the new delay.
- 10 As explained before the individualized ITD consists in the ITD of the data set scaled with a frequency independent factor. The ITD of the data set collected in the preprocessing stage (see section 4.1) was already parsed from a *.txt* file and gathered in a two dimensional array at start-up.
- 11 If the verbose flag is set the new ITD is displayed in console.
- 12 To compute the SRC ratio the new ITD must be expressed in samples according to the sample rate in use. The new delay (since last computing) is divided by amount of processing blocks stated in *smoothing_size*.
- 13 The counter of those processing blocks is re-started.
- 14 The SRC ratio is computed using equation 4.1.
- 15 The flag for a new SRC ratio is set and the flag of for computing a new delay is cleared.

Figure 4.4 shows a schematic description of our JACK client *process* callback function.

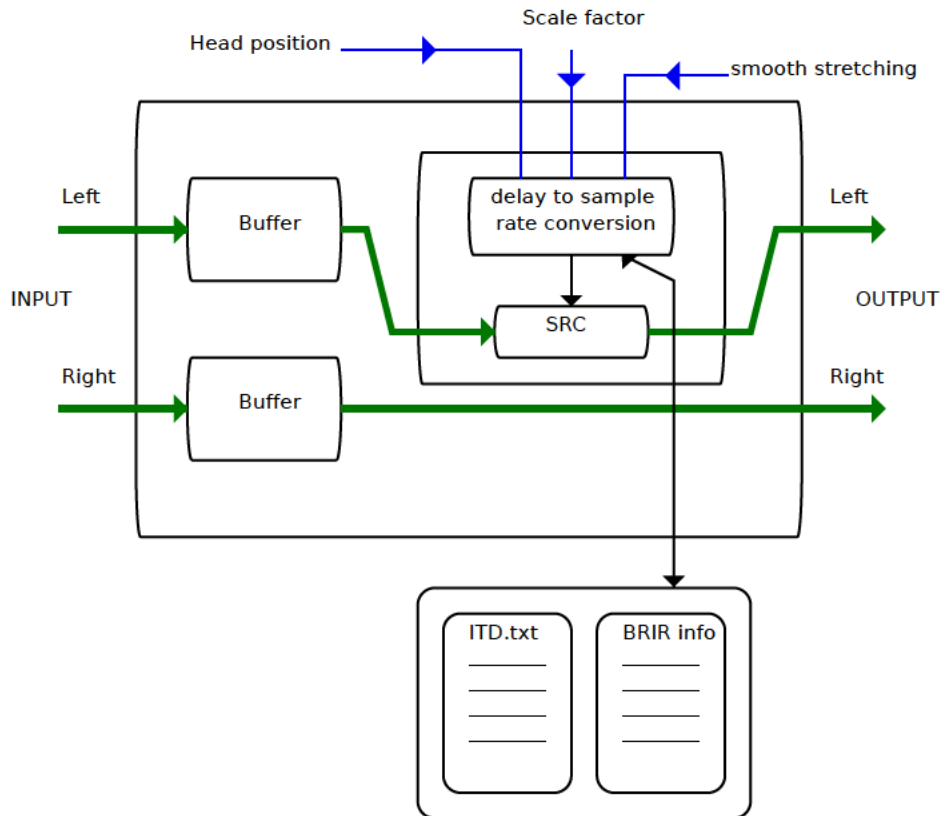


Figure 4.4.: Schematic description of the processing callback-function of the ITD individualizer

5. Anthropometry-based ITD individualization

The next step after designing a software capable of real-time ITD scaling is to provide a method that, based on anthropometric measures, scales the ITD originally contained in the BRIRs (i.e. that of the HATS FABIAN) that accomplishes the individualization task. The intrinsic individual characteristics of the ITD were covered in chapters 1 and 2. It was explained that the morphology of the subject plays a decisive role in the constitution of interaural delays.

Therefore, this chapter deals with the development of an empiric formula to find the individual ITD scaling factor based on anthropometric measures.

5.1. Listening test: ITD Individualization by scaling

The ITD individualization software was used in a listening test where the ITD of a foreign data set should be individually scaled by different subjects. Similar to the work of [Algazi et al. \(2001b\)](#) for finding an optimal head radius (see sec. 2.3), relevant anthropometric measures: head height, head width and head depth according to the norm [DIN33402-2E \(2005\)](#) of body dimensions as well as a new measure the **intertragus distance** were considered for finding an individual ITD scaling factor by means of multiple regression.

The last measure, the intertragus distance, represents the distance between the *incisura anterior* of left and right ears marking the tragus upper end. This measure was included because of its proximity to the ear-canal and the ease of its determination. Figure 5.1 shows the above mentioned anthropometric measures.

5.1.1. Test setup

A binaural data set of BRIRs was recorded in a damped room (Volume=155 m^3 , RT = 0.47 s) with 1° azimuthal resolution, covering 180° (+-90°) of the frontal plane not including elevation. The HATS FABIAN (Fast and Automatic Binaural Impulse response Acquisition) wearing the acoustically transparent headphones to be deployed in the simulation was used for the recording. ITDs were extracted using the onset detection method explained in

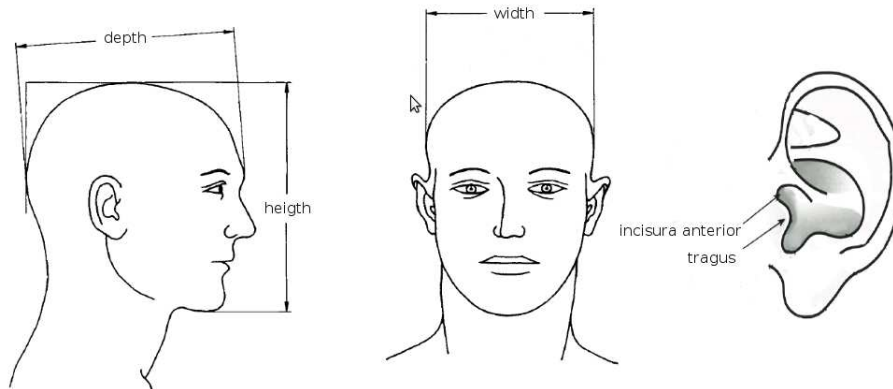


Figure 5.1.: Relevant anthropometric measures defining the individual ITD

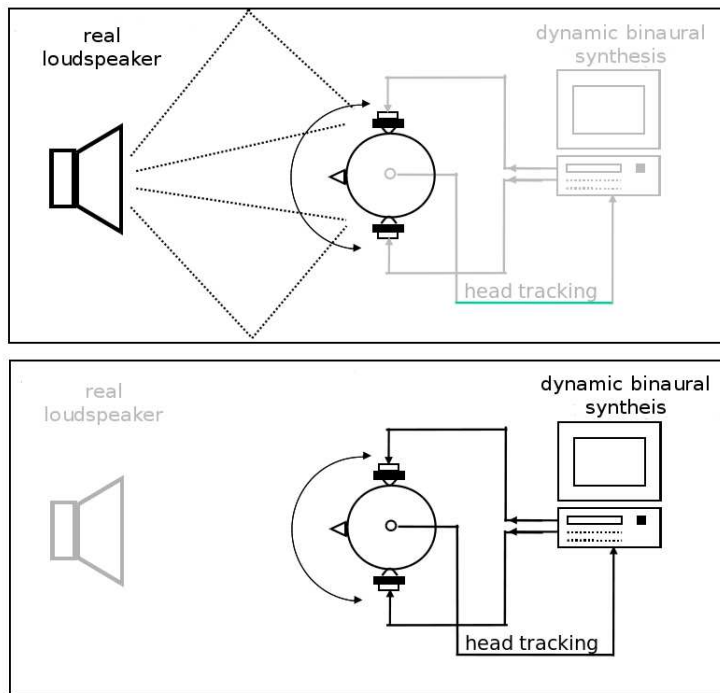


Figure 5.2.: Listening test setup. Up: While using the reference speaker. Down: while using binaural system.

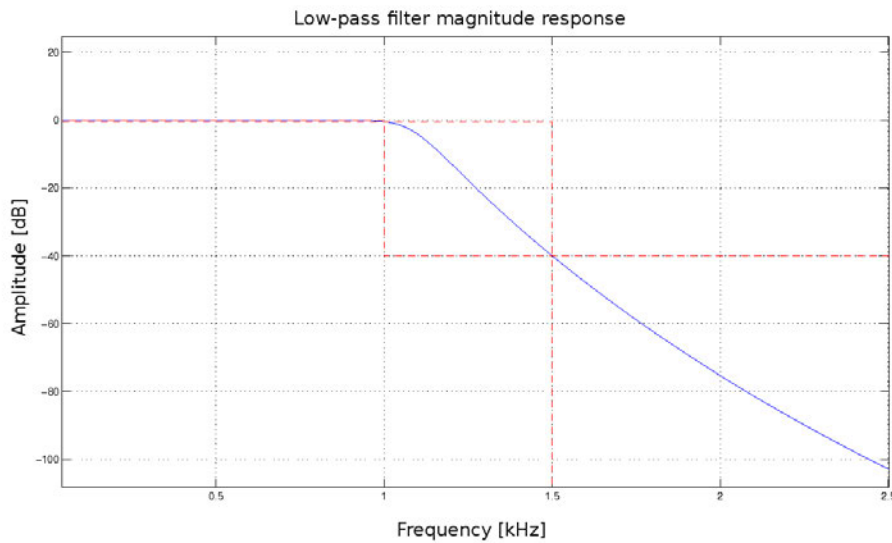


Figure 5.3.: Low-pass filter applied to the noise-burst stimulus to minimize the lateralization influence of ILD.

sec. 2.1.3.2, setting the detection threshold as -35 dB. A minimum phase BRIR dataset was stored for its use in the simulation. A Genelec 1030A loudspeaker was positioned frontally at a distance of 2 m from HATS/listener.

The listening test consisted in adjusting the presented ITD by finding the point where the virtual sound source appears stable and matches the position of the real loudspeaker. To accomplish this task the listener could switch between real loudspeaker and simulation. All subjects were told to rotate the head in order to produce bigger ITD values that facilitate the calibration task. Figure 5.2 shows a schema of the setup.

5.1.1.1. Stimulus

Low-pass filtered white noise bursts were used as stimulus to minimize the lateralization influence of ILD which start to be a dominant localization cue at about 1.5 kHz. The filter characteristics chosen were:

Filter type: Butterworth
 Attenuation: 40.0 dB
 F_{pass} : 1.0 kHz
 F_{stop} : 1.5 kHz

The magnitude response of the filter can be seen in fig. 5.3.



Figure 5.4.: Graphical user interface of the listening test audio application.

5.1.1.2. Listening-test's software

To execute the test an audio application was written in C++. The software implemented following features:

- Triggering either loudspeaker or binaural simulation.
- Randomized starting value of the scaling factor between 0.0 (no ITD) and 2.0 (2x original ITD) at each trial.
- Storing decisions (calibrated scaling factors) of each trial in a computer readable format.
- Test status sent over the network to a monitoring remote computer.
- GUI.

Figure 5.4 shows the GUI of the application, while figure 5.5 show the protocol of the test for a selected user including randomized ITD start values and the generated scaling factors as comma separated values (CSV).

5.1.1.3. Interface

A computer keyboard with marked operational keys (see fig. 5.6) was used as interface in the listening test. The keys marked with plus and minus allowed the listener to scale the ITD in $\pm 1\%$ steps of the original ITD. Although the ITD-I has no constraints in scaling resolution, this step size was chosen to keep the task of test simple.

5. Anthropometry-based ITD individualization

```

# *** CALIBRATION TEST ***

# PROJECT:      ITD_scaling_Test
# USING FILE:   ./audios/test/list.txt
# USER:        vp1
# scale on gui, scale on stretcher, offset, repeat, audio file,VRef, ARef
1.810074,1.010074,-0.800000,11,0,0,0
0.330000,1.030000,0.700000,12,0,0,0
0.250000,0.850000,0.600000,5,0,0,0
1.420000,1.020000,-0.400000,10,0,0,0
1.650000,1.050000,-0.600000,15,0,0,0
1.530000,0.930000,-0.600000,7,0,0,0
0.700000,1.000000,0.300000,6,0,0,0
1.290000,0.990000,-0.300000,2,0,0,0
1.810000,1.010000,-0.800000,9,0,0,0
0.380000,0.980000,0.600000,0,0,0,0
  
```

Figure 5.5.: Protocol of a selected user’s listening test written in CSV format. Red arrow points to the column of the generated ITD scaling factors

Listener’s could switch between real loudspeaker and simulation at any time. Once the appropriate scaling factor was found, the “apply” key allowed the user to store the value and start a new trial.

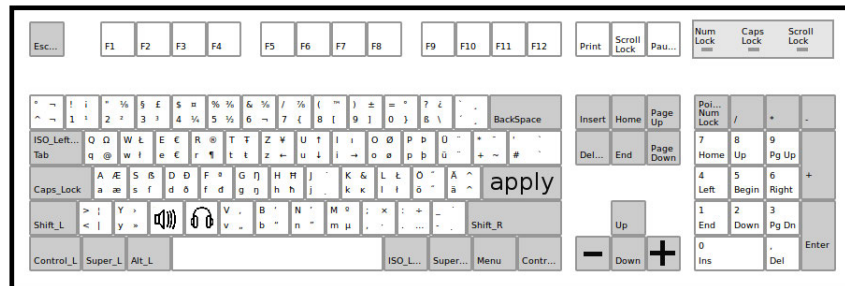


Figure 5.6.: User interface for the listening test

5.1.2. Listening test procedure

Before the test began, the 4 head measurements mentioned in sec 5.1 were conducted on the 11 participants that took part in the test. Information about age, experience in listening tests, music studies and known hearing loss were gathered with following results:

Average age: 28 years.

Experience in: listening tests All.

Music studies: All but 1, 8 years average.

Hearing loss acc. to own testimony: None.

The participants were seated in the same place as FABIAN for the recording of the binaural dataset. Special care was taken on keeping the same height of the subject's ear-canal.

All participants were instructed about the use of the interface and were introduced with the effects of non-individual ITD with the aid of examples (ie. exaggerating the ITD size to provoke instability of the virtual sound source at head movements). The listeners could practice the individualization task until they felt confident with the procedure.

Each subject had to scale 10 times its own perceptually correct ITD, always starting with randomized values. The test took 25 to 30 minutes.

5.2. Statistical analysis

Despite training and the obvious effect of the non-individual ITD, the calibration task was for some subjects difficult to realize. By means of residual analysis and outlier tests 2 of the 11 subjects were excluded from the final analysis.

Figure 5.7 shows the individual distributions from the 10 ITD scaling factors of each of the 9 subjects as box-plots. Note the big confidence intervals. The individual medians covered -5% to $+7\%$.

Table 5.1 presents a resume of the multiple regression analysis on different configurations.

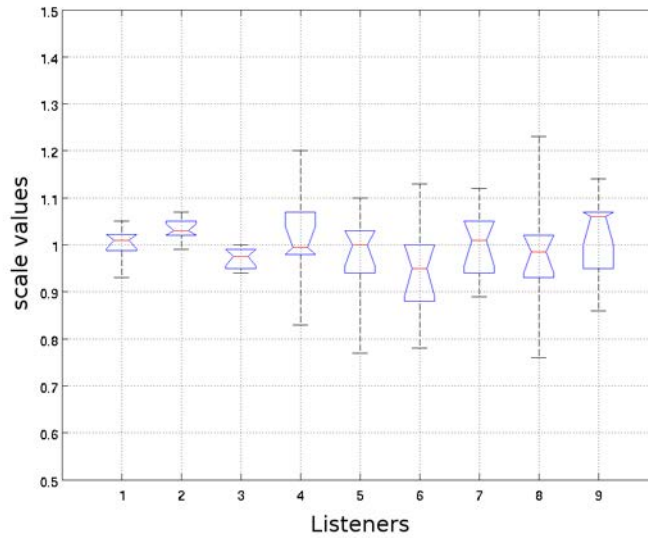


Figure 5.7.: Distribution of the individually generated ITD scaling factors from 9 subjects. Note the big dispersions.

Explained variance (adj. R^2) indicated that a single predictor the intertragus distance is best suited for predicting the individual ITD scaling factors reaching 66.0 % of explained variance.

Figure 5.8 shows that the individual ITD scaling factors increase linearly in proportion to the head diameter described by the intertragus distance. Figure 5.8 also shows all nine mean individual scaling values together with their 95% confidence intervals (CI).

The linear regression model is also shown with its 95% CIs.

5.3. Anthropometry-based ITD individualization formula

The regression formula for predicting the individual scaling factor based on the intertragus distance is:

$$S = 0.00304 \cdot d_i + 0.5792 \quad (5.1)$$

with the intertragus distance d_i , specified in millimeters. Note that this model is valid only for binaural datasets acquired with the HATS FABIAN. The model could possibly be generalized to arbitrary binaural data by scaling the predicted values by a certain ratio of the

Model	R	R square	Adjusted R square	Std. Error of the estimate
Intertragus distance	0.838	0.703	0.660	0.0155637
Intertragus distance, Depth-DIN	0.839	0.704	0.606	0.167640
Intertragus distance, Depth-DIN, Height-DIN	0.840	0.705	0.529	0.183355

Table 5.1.: Summary of correlation factors using three models: Intertragus distance; Intertragus distance and Depth-DIN; Intertragus distance, Depth-DIN and Height-DIN. Adjusted R square of the intertragus distance is bigger as in the other models.

intertragus distances of the foreign and FABIAN's artificial heads. Though this aspect has not been investigated yet.

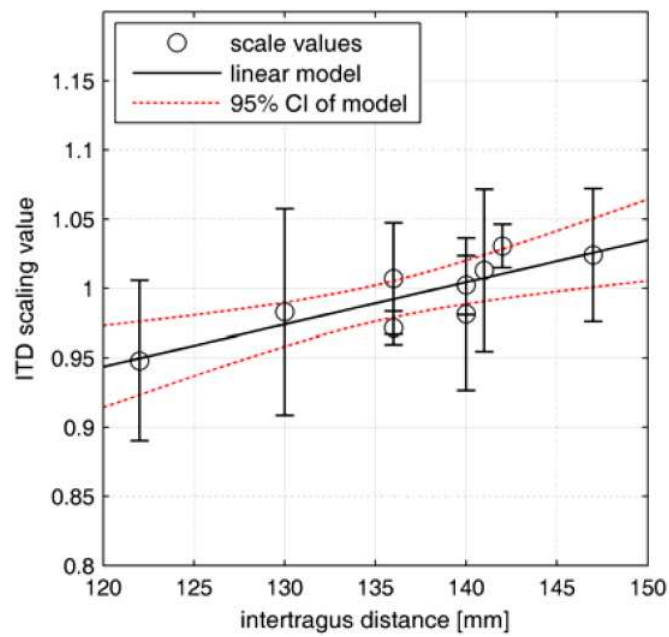


Figure 5.8.: Modeling of listening test results: The linear regression model over the intertragus distance is shown with hyperbolic 95% CIs.

6. Summary and conclusions

This work addressed the problematic in the binaural synthesis reproduction using non-individual auralization cues. A method to accomplish individualization of one of this cues, the ITD, highly responsible for the localization in the horizontal plane, was developed as well as a software application that implements it in real-time.

The method consists in decomposing binaural datasets into their temporal and spectral characteristics. The spectral cues (ILD and SC) of the data set permeate the audio information using the classical convolution approach, while the ITD is reinserted as a fractional delay (using time stretching) between left and right ear signals. In order to achieve individualization the ITD is reinserted scaled with a frequency independent factor. The issue of instability of virtual sound sources can be, under the premise of proper calibration, completely resolved.

To find the individual ITD scaling factor a formula based on one anthropometric measure the *intertragus distance* was developed based on data gathered in an empirical listening test.

Outlook and future work

The individualization methods explained in this work are the first of their kind in the development of binaural synthesis techniques; they are therefore attained to further investigation. These are a list of tasks that should be taken into account in a future work:

- In order to assess the validity of the proposed formula (eq. 5.1) a new validation listening test should be carried out.
- Assess adoption of regression formula (eq. 5.1) to HRIRs from other HATS, i.e. by correction factors based on intertragus-distance-ratios or ITD-ratios.
- The spatial resolution of binaural data sets as well as the use of interpolation of the extracted ITD values should be reviewed since our methodology allows separate processing of time and spectra, both working on different resolutions.

6. Summary and conclusions

- In this work a frequency and angle independent factor was assumed to be sufficient. This aspect could also be reviewed.

Bibliography

Algazi et al. 1997

ALGAZI, V. R. ; DIVENYI, P.L. ; MARTINEZ, V.A. ; DUDA, R. O.: "Subject Dependent Transfer Functions in Spatial Hearing". In: *IEEE, Proc. of the 40th Midwest Symposium, Sacramento, CA, Aug. 3-6 1997* Bd. 2, 1997, S. 877–880

Algazi et al. 2001a

ALGAZI, V. R. ; DUDA, R. O. ; THOMPSON, D. M. ; AVENDANO, C.: "The CIPIC HRTF Database". In: *IEEE, Proc. of the Workshop on Applications of Signal Processing to Audio and Acoustics, Mohonk Mountain House, New Paltz, NY, Oct.21-24 2001*, 2001, S. 99–102

Algazi et al. 2001b

ALGAZI, V. R. ; AVENDANO, C. ; DUDA, R. O.: "Estimation of a Spherical-Head Model from Anthropometry". In: *J. Audio Eng. Soc* 49 (2001), Nr. 6, S. 472–479

Begault et al. 2000

BEGAULT, D. R. ; WENZEL, E. M. ; ANDERSON, M. R.: "Direct Comparison of the Impact of Head Tracking, Reverberation, and Individualized Head-Related Transfer Functions on the Spatial Perception of a Virtual Speech Source". In: *in Proc.2000 of the 108th AES Convention Paris. Preprint 5134*, 2000, S. 904–916

Brungart et al. 2006

BRUNGART, D. ; KORDIK, A. ; SIMPSON, B.: "Effects of Headtracker Latency in Virtual Audio Displays". In: *J. Audio Eng. Soc* 54 (2006), Nr. 1/2, S. 32–44

Burstein 1988

BURSTEIN, H.: "Approximation Formulas for Error Risk and Sample Size in ABX Testing". In: *J. Audio Eng. Soc* 36 (1988), Nr. 11, S. 879–883

Busson et al. 2005

BUSSON, S. ; KATZ, B. ; NICOL, R.: "Subjective Investigations of the Interaural Time Difference in the Horizontal Plane.". In: *Proc. of the 118th Convention of the Audio Eng. Soc., Barcelona Spain. Preprint 6324*, 2005

Constan and Hartmann 2003

CONSTAN, Z. A. ; HARTMANN, W. H.: "On the Detection of Dispersion in the Head-Related Transfer Function". In: *J. Ac. Soc. Am.* 114 (2003), Nr. 2, S. 998–1008

DIN33402-2E 2005

DIN33402-2E: "Körpermaße des Menschen". Deutsches, Institut für Normung e.V., 2005

Duda et al. 1999

DUDA, R. O. ; AVENDANO, C. ; ALGAZI, V. R.: "An Adaptable Ellipsoidal Head Model For The Interaural Time Difference". In: *Proc. of the IEEE International Conference on Acoustics, Speech and Signal Processing ICASSP 1999*, 1999, S. 965–968

Estrella 2010

ESTRELLA, J. R.: "On the Extraction of Interaural Time Differences from Binaural Room Impulse Responses". Studienarbeit at the Technische Universität Berlin, 2010

GNU LGPL

GNU LGPL: "The current version of the GNU Lesser General Public License". <http://www.gnu.org/licenses/lgpl.html>

Huopaniemi and Smith 1999

HUOPANIEMI, J. ; SMITH, J. O.: "Spectral and Time-Domain Preprocessing and the Choice of Modeling Error Criteria for Binaural Digital Filters". In: *Proc. of the 16th International Conference on Spatial Sound Reproduction of the Audio Eng Soc., Rovaniemi, Finland*, 1999, S. 301–312

Jot et al. 1995

JOT, J. M. ; LARCHER, V. ; WARUSFEL, O.: "Digital Signal Processing Issues in the Context of Binaural and Transaural Stereophony". In: *Proc. of the 98th Convention of the Audio Eng. Soc.* Paris, France, Februar 1995

Kuhn 1977

KUHN, George F.: "Model for the interaural time differences in the azimuthal plane". In: *J. Ac. Soc. Am.* 62 (1977), July, Nr. 1, S. 139–147

Kulkarni et al. 1999

KULKARNI, A. ; ISABELLE, S. K. ; COLBURN, H. S.: "Sensitivity of Human Subjects to Head-Related Transfer-Function Phase spectra". In: *J. Ac. Soc. Am.* 105 (1999), Nr. 5, S. 2821–2840

Langendijk and Bronkhorst 2002

LANGENDIJK, E. H. ; BRONKHORST, A. W.: "Contribution of spectral cues to human sound localization". In: *J. Ac. Soc. Am.* 112 (2002), Nr. 4, S. 1583–1596

Larcher and Jot 1999

LARCHER, V. ; JOT, J. M.: "Techniques D'Interpolation de filtres Audio-numériques : Application à la Reproduction Spatiale des sons sur Écouteurs". In: *Congrès Français D'Acoustique, Marseille, France, 1999*

Liblo OSC API

LIBLO OSC API: *Homepage of the liblo light-weight OSC implementation.*
<http://liblo.sourceforge.net/>,

Libsamplerate SRC API

LIBSAMPLERATE SRC API: "*The Secret Rabbit Code: Homepage of the libsamplerate application interface*". <http://www.mega-nerd.com/SRC/index.html>

Libxml++

LIBXML++: "*Libxml++2.x: Homepage of the libxml2 application interface*".
<http://libxmlplusplus.sourceforge.net/>

Lindau 2006

LINDAU, A.: "*Ein Instrument zur softwaregestützten Messung binauraler Raumimpulsantworten in mehreren Freiheitsgraden.*". Magister Arbeit, Technische Universität Berlin, 2006

Lindau 2009

LINDAU, A.: "The Perception of System Latency in Dynamic Binaural Synthesis". In: *Proc. of the NAG/DAGA 2009 International Conference of Acoustics, Rotterdam Netherlands, 2009*

McAnally and Russell 2002

MCANALLY, K. I. ; RUSSELL, M.: "Variability in the Headphone-to-Ear-Canal Transfer Function". In: *J. Audio Eng. Soc.* 50 (2002), Nr. 4, S. 263–266

Mills 1958

MILLS, A. W.: "On the Minimum Audible Angle". In: *J. Ac. Soc. Am.* 30 (1958), April, Nr. 4, S. 237–246

Minnaar et al. 1999

MINNAAR, P. ; PLOGSTIES, J. ; CHRISTENSEN, F. ; MØOLLER, H. ; OLESEN, S. K.: "The Audibility of All-Pass Components in Binaural Synthesis". In: *Proc. of the 106th Audio Eng. Soc. Convention, Munich, Germany, 1999* (4911)

Minnaar et al. 2000

MINNAAR, P. ; PLOGSTIES, J. ; OLESEN, S. K. ; CHRISTENSEN, F. ; MØLLER, H.:
“The Interaural Time Difference in Binaural Synthesis”. In: *Proc. of the 108th Audio
Eng. Soc. Convention, Paris, France, 2000* (Preprint 5133)

Møller and Hoffmann 2006

MØLLER, H. ; HOFFMANN, P. F.: “Audibility of Spectral Differences in Head-Related
Transfer Functions”. In: *Proc. of the 120th. Convention of the Audio Eng. Soc. Paris,
France, 2006*

Møller et al. 1995

MØLLER, H. ; SØRENSEN, F. M. ; JENSEN, B.C. ; HAMMERSHØI, D.: “Transfer
Characteristics of Headphones Measured on Human Ears”. In: *J. Audio Eng. Soc* 43
issue 4 (1995), April, S. 203–217

Murphy and Myers 1999

MURPHY, K. R. ; MYORS, B.: “Testing the Hypothesis That Treatments Have Negligi-
ble Effects: Minimum-Effect Tests in the General Linear Model”. In: *J. Appl. Psychol.*
84 (1999), Nr. 2, S. 234–248

Nam et al. 2008

NAM, J. ; ABEL, J. S. ; III, J. O. S.: “A Method for Estimating Interaural Time
Difference for Binaural Synthesis”. In: *Proc of the 125th Audio Eng. Soc Convention,
San Francisco Ca., U.S.A., 2008*

Nicol 2010

NICOL, R.: “*Binaural Technology*”. New York, U.S.A. : Monograph of the Audio
Eng. Soc., 2010

Oppenheim et al. 1999

In: OPPENHEIM, A. V. ; SCHAFER, R. W. ; BUCK, J. R.: *Discrete-Time Signal Pro-
cessing (2nd Edition) (Prentice-Hall Signal Processing Series). 2.* Prentice Hall, 1999.
– ISBN 0137549202, S. 775–802

Preis 1982

PREIS, D.: “Phase Distortion and Phase Equalization in Audio Signal Processing. A
Tutorial Review”. In: *J. Audio Eng. Soc* 30 (1982), S. 774–794

Sandvad 1996

SANDVAD, J.: “Dynamic Aspects of Auditory Virtual Environments”. In: *Proc. of the
120th. Convention of the Audio Eng. Soc. Copenhagen, Denmark, 1996*

Savioja et al. 1999

SAVIOJA, L. ; HUOPANIEMI, J. ; LOKKI, T. ; VNNEN, R.: "Creating Interactive Virtual Acoustic Environments". In: *J. Audio Eng. Soc.* 47 (1999), S. 675–705

Strutt 1907

STRUTT, J. W.: "On our Perception of Sound Direction". In: *Philos* 13 (1907), S. 214–232

Välimäki and Laakso 2000

VÄLIMÄKI, V. ; LAAKSO, T.: "Principles of Fractional Delay Filters". In: *Proc. of the IEEE International Conference on Acoustics, Speech and Signal Processing*. Istanbul, Turkey, June 2000

Välimäki et al. 1996

VÄLIMÄKI, V. ; LAAKSO, T. ; KARJALAIEN, M. ; LAINE, U. K.: "Splitting the Unit Delay - Tools for Fractional Delay Filter Design". In: *IEEE Signal Processing Magazine* 14 (1996), Nr. 1, S. 30–60

Wefers 2007

WEFERS, F.: "*Optimizing Segmented Real-time Convolution*". Diploma thesis at the RWTH Aachen University, September 2007

Weinzierl 2004

WEINZIERL, S.: "*Kommunikationstechnik 1*". Skript zur Vorlesung, Berlin: Technische Universität Berlin, 2004

Wenzel 1999

WENZEL, E.: "Effect of Increasing System Latency on Localization of Virtual Sounds". In: *Proc. of the 16th. International Conference: Spatial Sound Reproduction of the Audio Eng. Soc. Rovaniemi, Finland, 1999*

Wenzel et al. 1988

WENZEL, E. ; WIGHTMAN, F. ; KISTLER, D. ; FOSTER, S.: "Acoustic Origins of Individual Differences in Sound Localization Behavior". In: *J. Ac. Soc. Am.* 84 (1988), Nr. S1, S. S79–S79

Wenzel et al. 1993

WENZEL, E. M. ; ARRUDA, M. ; KISTLER, D. J. ; WIGHTMAN, F. L.: "Localization using nonindividualized head-related transfer functions". In: *J. Ac. Soc. Am.* 94 (1993), Nr. 1, S. 111–123

Woodworth et al. 1972

WOODWORTH, R. S. ; SCHLOSBERG, H. ; KLING, J. W. ; RIGGS, L. A.: "*Woodworth*

and Schlosberg's Experimental Psychology". 3d ed. by J. W. Kling and Lorrin A. Riggs and seventeen contributors. Methuen, London, 1972. – xv, 1279 p. S. – ISBN 0416674607

Wright 2005

WRIGHT, M.: "Open Sound Control: An Enabling Technology for Musical Networking". In: *Organised Sound* 10, Issue 3 (2005), December, S. 192–200

Zwicker and Fastl 1999

ZWICKER, E. ; FASTL, H.: "*Psychoacoustics Facts and Models. 2. Ed*". Heidelberg, Germany : Springer Information Sciences, 1999

A. Zusammenfassung in deutscher Sprache

Der Prämisse folgend, dass die relevanten Merkmale (Cues) für die Lokalisation innerhalb einer akustischen Umgebung im Schalldruck am Trommelfell vorhanden sind, ermöglicht die binaurale Synthese die Wiedergabe von virtuellen akustischen Umgebungen (VAE - virtual acoustic environments).

In der sog. Duplextheorie des Hörens bestimmte Lord Rayleigh ([Strutt 1907](#)) interaurale Pegel- (ILD, interaural level difference) und Laufzeitunterschiede (ITD, interaural time difference) als maßgebliche Cues für das räumliche Hören. Er stellte fest, dass der Einfall von Schall mit kleinen Wellenlängen im Vergleich zu den Kopfabmessungen eine Schallabschattung bewirkt, wodurch interaurale Pegeldifferenzen entstehen. Außerdem hat der Ohrabstand zur Folge, dass interaurale Laufzeitdifferenzen entstehen.

Aufgrund ihrer Natur haben beide Cues für jedes Subjekt individuelle Ausprägungen. Da es aber unpraktikabel ist individuelle Messungen durchzuführen, werden für die datenbasierte Binauralsynthese sogenannte binaurale Raumimpulsantworten (BRIR, binaural room impulse response) verwendet (siehe [Abb. A.1](#)). Diese enthalten die binauralen Cues des bei der Messung verwendeten (Kunst-)kopfes, welche, je nach Hörer, unterschiedliche Auswirkungen haben können.

In dieser Diplomarbeit werden sowohl ein Verfahren für die Individualisierung einer dieser Cues – der ITD – sowie eine Software für die praktische Umsetzung vorgestellt. Darüber hinaus wird eine auf anthropometrischen Maßen basierende empirische Formel hergeleitet, welche dazu dient, die praktische Anwendung der Individualisierung zu vereinfachen.

A.1. Motivation

A.1.1. Die Verwendung von nicht-individuellen Lokalisationscues

Die Verwendung von nicht-individuellen binauralen Cues wurde von verschiedenen Autoren ausführlich untersucht ([Algazi et al. 1997](#); [Wenzel et al. 1988, 1993](#)). Dies kann anhand von zwei Aspekten verdeutlicht werden:

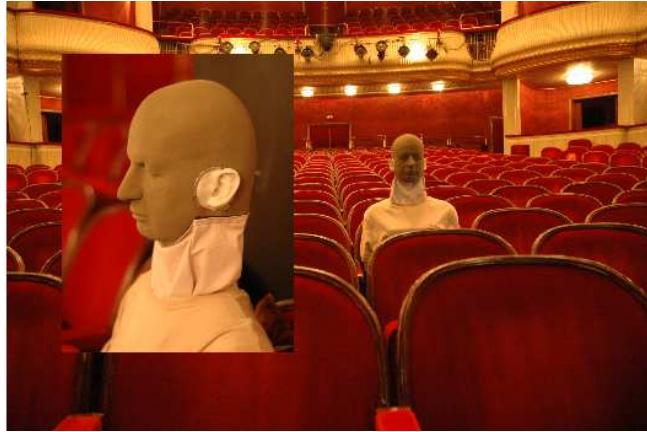


Abbildung A.1.: Data-Set Akquise unter Verwendung des HATS (Head and torso simulator) FABIAN (Lindau 2006).

- **die Veränderung der Klangfarbe**, welche aufgrund der Verwendung von nicht-individuellen ILDs bzw. unterschiedlichen spektralen Charakteristika zustande kommt. Da diese Unterschiede meistens nur im direktem Vergleich zur realen Schallquelle wahrnehmbar sind (Møller and Hoffmann 2006), werden sie im allgemeinen als nicht gravierend eingestuft.
- **Lokalisationsfehler** aufgrund von nicht-individuellen ITDs sind dagegen störender, weil sie zu konstanten Quellverschiebungen und/oder zur Instabilität der virtuellen Schallquellen führen. Algazi et al. (2001b) erwähnt dass bei dynamischen binauralen Systemen (d.h. der Kopfbewegung folgende Systeme), wenn der Kopfradius des Hörers kleiner ist als der Kopfradius des zugrundeliegenden Data-Sets, die virtuellen Schallquellen wahrgenommen werden, als würden sie in die Gegenrichtung wandern. Hat der Hörer einen größeren Kopfradius als der bei der Akquisition verwendeten Kopf, werden die virtuellen Schallquellen wahrgenommen, als würden sie mit der Kopfbewegungsrichtung mitwandern.

A.1.2. Das vorgeschlagene Individualisierungsmodell

Um die im vorigen Abschnitt erwähnten Lokalisationsfehler zu vermeiden, wird ein Individualisierungsmodell entwickelt, basierend auf der frequenzunabhängigen Manipulation der ITD in Echtzeit. Das Modell kann in drei Schritten beschrieben werden:

- Die im binauralen Data-Set enthaltene $ITD(\theta, \phi)$ soll in einen maschinenlesbaren Format erfasst werden.

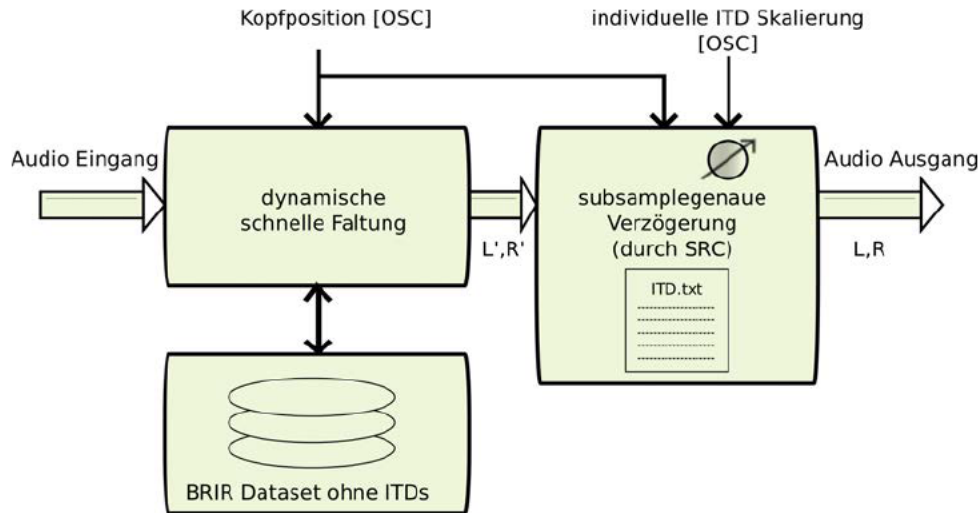


Abbildung A.2.: Vereinfachtes Schema des ITD- Individualisierungsmodells

- Die Faltungseinheit verwendet ein Data-Set aus minimalphasigen anstelle der originalen Impulsantworten.
- Die fehlenden Laufzeitunterschiede zwischen linkem und rechtem Kanal werden durch eine subsample-genaue Zeitverzögerung ersetzt, welche die im ersten Schritt erfasste ITD mit einem Individualisierungsfaktor skaliert.

Abbildung A.2 zeigt eine schematische Darstellung des Modells.

A.1.3. Vorteile des Modells

Die Implementierung des Modells hat folgende Vorteile:

- Unter der Voraussetzung einer entsprechend skalierten ITD kann die Instabilität von virtuellen Schallquellen beseitigt werden.
- Reduktion der CPU Last aufgrund der Faltung von verkürzten Impulsantworten.
- Artefaktfreie Überblendung von Impulsantworten. Dynamische binaurale Systeme verwenden je nach Kopf- bzw. Quellposition unterschiedliche Impulsantworten. In unserem Modell enthalten diese während des Faltungsprozesses keine Laufzeit. Dadurch werden bei der Überblendung Kammfilter-, Repetitions- und Omissionsartefakte vermieden.

- Verbesserte Verteilung der CPU Last bei Multiprozessorsystemen. Virtuelle Schallquellen werden in unserem Modell getrennt behandelt, dies ermöglicht die Parallelisierung der Audioprozesse in Multiprozessorsystemen.
- Erhaltung des natürlichen Hörvorgangs. Das Individualisierungsmodell aus Abb. A.2 sieht die Verwendung von *Time-Stretching* für das Wiedereinfügen der ITD vor. Dieser Prozess ähnelt dem physikalischen Vorgang des Dopplereffekts beim natürlichen Hören, wenngleich die Auswirkungen nicht direkt wahrnehmbar sind (siehe Anhang F).
- Reduktion der Winkelauflösung bei der Data-Set Akquisition. Die getrennte Behandlung von spektralen und zeitlichen binauralen Cues ermöglicht die Verwendung von unterschiedlichen Auflösungen, z.B. eine grobe Auflösung bei der Data-Set Akquisition und beim Faltungsprozess während die temporalen Cues, unter Verwendung von Interpolation, mit feinerer Auflösung repräsentiert werden können.

A.2. Stand der Forschung

A.2.1. Individualisierung mit Hilfe von anthropometrischen Maßen

Verschiedene Autoren beschäftigten sich mit der Individualisierung von HRTFs auf der Basis von anthropometrischen Maßen. Bereits im Jahr 1962 wurde von Woodworth und Schlosberg eine Formel für die Ermittlung von ITDs, auf Grundlage des Kopfradius entwickelt ¹⁵. Diese Formel (Gl. A.1) nimmt als Modell einen sphärischen Kopf mit symmetrisch zur Medianebene angebrachten Ohren an (Woodworth et al. 1972):

$$ITD = \frac{a}{c}(\sin \theta + \theta) \quad (\text{A.1})$$

mit:

a = Kopfradius

c = Schallgeschwindigkeit

θ = Azimut in [Rad] $\in -\frac{\pi}{2} < \theta < \frac{\pi}{2}$

Duda et al. (1999) entwickelte ein ellipsoides Kopfmodell, das eine bessere Approximation der ITD für Elevationseinflüsse liefert. Allerdings sind dafür mehrere anthropometrische Messungen notwendig und das Verfahren ist rechenintensiv. Daher eignet sich diese Methode nicht für die Echtzeitimplementierung.

¹⁵Für den Fall synthetischer Schallfelder oder wenn die Position der Quelle bekannt ist.

[Algazi et al. \(2001b\)](#) untersuchten den Fehler, der unter Verwendung eines nicht angepassten Kopfradius in der Woodworth-Schlosberg Formel entsteht, und entwickelten eine empirische Formel für die Ermittlung eines optimalen Kopfradius basierend auf Messungen von Kopfbreite, Kopftiefe und Kopfhöhe von 25 Versuchspersonen. Sie kamen zu dem Schluss, dass nur zwei Messungen notwendig sind, um einen optimalen Kopfradius zu berechnen, und zwar der Kopfbreite und der Kopftiefe.

Larcher und Jot erweiterten die Woodworth-Schlosberg Formel, um die Abhängigkeit der ITD zur Elevation miteinzubeziehen ([Larcher and Jot 1999](#)). Ebenso wurde in [Savioja et al. \(1999\)](#) eine Vereinfachung von Larcher's Formel dargestellt.

Im Fall empirischer Data-Sets mit unbekannter Position von Schallquellen sind die o.g. Methoden nicht anwendbar. Nichtsdestotrotz stellt die Verwendung von anthropometrischen Maßen einen inspirierenden Individualisierungsansatz dar.

A.3. Methoden

Das vorgeschlagene Individualisierungsmodell setzt voraus, dass sowohl spektrale als auch temporale Cues für die räumliche Wahrnehmung in getrennten Prozessen behandelt werden können. Diese Annahme soll in diesem Abschnitt mit Hilfe der Systemtheorie begründet werden.

In der LTI (linear time invariant) Systemtheorie kann der komplexe Frequenzgang einer kopfbezogenen Transferfunktion als Betrag und Phase dargestellt werden:

$$H(j\omega) = |H(\omega)| \cdot e^{j\varphi_{min}(\omega)} \cdot e^{j\varphi_{excess}(\omega)} \quad (\text{A.2})$$

Der frequenzabhängige Exzessphasenanteil kann wiederum durch ihre linearphasigen und allpass-haltigen Komponenten ausgedrückt werden.

$$H(j\omega) = |H(\omega)| \cdot e^{j\varphi_{min}(\omega)} \cdot e^{j\varphi_{lin}(\omega)} \cdot e^{j\varphi_{allpass}(\omega)} \quad (\text{A.3})$$

Aufgrund der geringen Hörbarkeit von Phasenspektren für Menschen ([Preis 1982](#)) und dadurch, dass für die meisten Schalleinfallswinkel die Allpass-Komponenten unhörbar sind, können diese vernachlässigt werden, ohne die räumliche Wahrnehmung zu beeinträchtigen ([Minnaar et al. 1999](#)):

$$H(j\omega) = |H(\omega)| \cdot e^{j\varphi_{min}(\omega)} \cdot e^{j\varphi_{lin}(\omega)} \quad (\text{A.4})$$

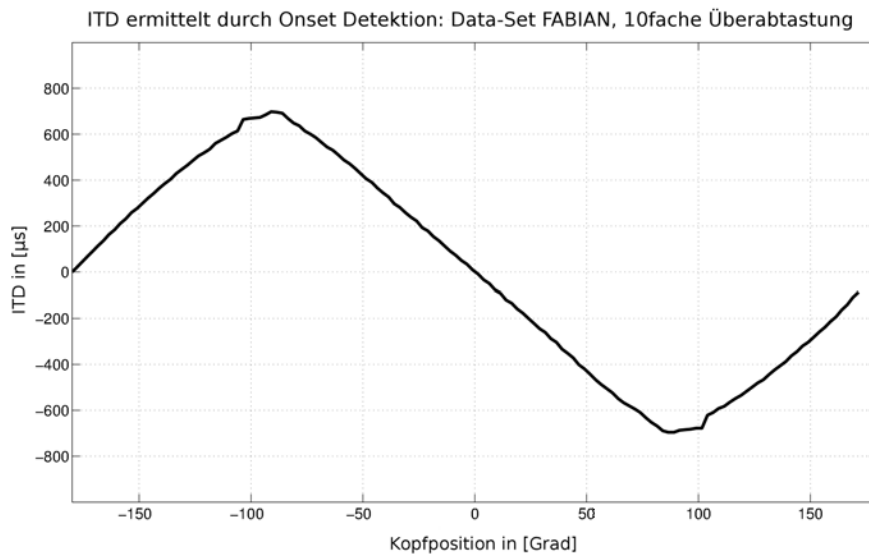


Abbildung A.3.: ITD ermittelt durch Onset Detektion bei 10facher Überabtastung, Detektionsschwelle -3dB . Data Set: FABIAN's HRIRs (Elevation: 0° , Azimut: -180° bis $+180^\circ$, Auflösung: 1°)

Weiterhin kann die linearphasige Komponente in Gl. A.4 durch eine Zeitverzögerung ersetzt werden, solange diese einer geeigneten Näherung der ITD entspricht (Kulkarni et al. 1999).

A.3.1. Ermittlung der ITD aus binauralen Datensätzen mittels Onset Detektion

Die Anwendung unseres Modells erfordert Methoden für die Ermittlung der $ITD(\theta, \phi)$ und die Extraktion von minimalphasigen Impulsantworten aus empirischen Data-Sets. In einer Vorarbeit (Estrella 2010) wurden verschiedene Verfahren ausführlich quantitativ und perceptiv ausgewertet. An dieser Stelle werden diese Methoden nur erwähnt.

Als geeignete Verfahren zur Ermittlung der ITD wurde die Onset Detektion ermittelt. Diese Methode bestimmt für jede Impulsantwort eines BRIR-Paars die Sampleposition, bei der ein Schwellwert bezüglich des BRIR-Spitzenwerts erreicht wird. Aus der Differenz der beiden Samplepositionen kann der interaurale Laufzeitunterschied bestimmt werden. Es ist anzumerken, dass, um die Genauigkeit der Detektion zu verbessern, diese im überabgetasteten Bereich erfolgen sollte. Abbildung A.3 zeigt ein Beispiel dieses Verfahrens.

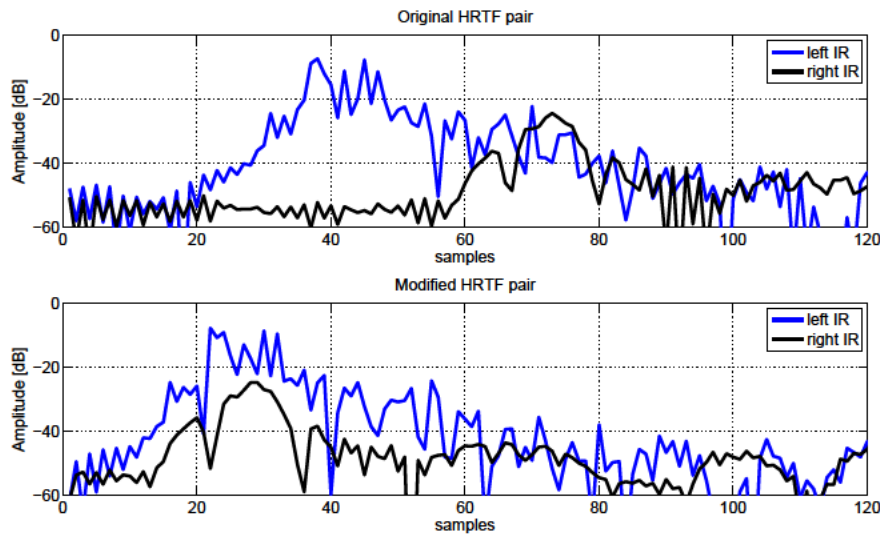


Abbildung A.4.: Extraktion von minimalphasigen Impulsantworten mit der Methode der Onset Detektion. Die leichte Veränderung der Einhüllenden beruht auf dem Manipulationsvorgang an den BRIRs.

A.3.2. Extraktion von minimalphasigen Impulsantworten nach der Onset Detektion Methode

Als geeignete Methode für die Extraktion von minimalphasigen Impulsantworten wurde in [Estrella \(2010\)](#) die Trennung der originalen BRIRs in einen minimalphasigen Anteil und einen Verzögerungsanteil am Abtastwert des jeweiligen Onsetkriteriums bestimmt. Abbildung A.4 zeigt ein Beispiel dieses Verfahrens.

A.3.3. Fraktionale Zeitverzögerung

Um die ITD akkurat zu skalieren, sollte das System in der Lage sein, fraktionale Zeitverzögerungen (d.h. zwischen Abtastwerten) zu liefern:

$$nT < \tau < (n + 1)T$$

wobei: τ = Zeitverzögerung

T = Abtastintervall mit $n \in \mathbb{N}$

Als optimaler Ansatz für die Umsetzung der fraktionalen Verzögerung in zeitdiskreten Systemen wird in [Välimäki et al. \(1996\)](#) die Anwendung der *sinc* Interpolation vorgeschlagen.

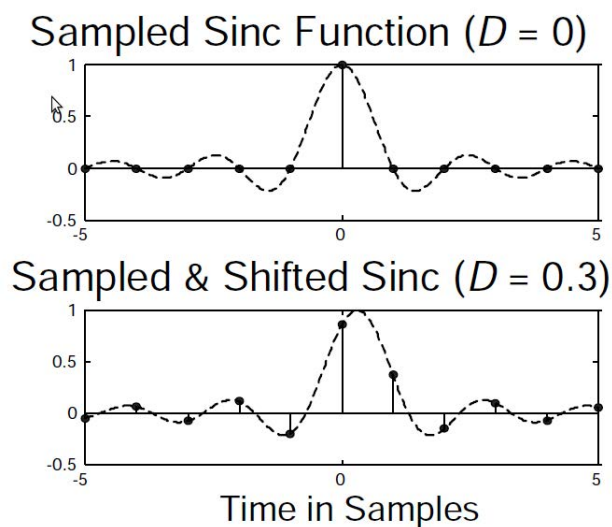


Abbildung A.5.: Ideale fraktionale Zeitverzögerung. Oben: Delay D ist ganzzahlig, die Abtastung erfolgt an den Null-Übergängen. Unten: Delay D ist nicht ganzzahlig, Abtastung erfolgt zwischen den Null-Übergängen. Dem Idealfall entspricht eine Impulsantwort unendlicher Länge. Aus [Välimäki and Laakso \(2000\)](#).

Die Methode, bekannt als bandbegrenzte Interpolation, basiert auf der Verwendung von versetzten *sinc* FIR Filtern (siehe Gl. A.5) die das zu verzögernde Signal erneut abtasten. Diese Methode kann als eine Erweiterung des Nyquist Theorems interpretiert werden:

$$h_{id}(n) = \frac{\sin[\pi(n-D)]}{\pi(n-D)} = \text{sinc}(n-D) \quad (\text{A.5})$$

Abbildung A.5 zeigt ein Anwendungsbeispiel.

Da die *sinc* Filter keine unendliche Länge haben können, spielt die Auswahl der Anzahl von Filterkoeffizienten eine große Rolle in Bezug auf die Bandbreite des Audiosignals.

A.4. Implementierung

In diesem Abschnitt werden praktische Aspekte der Implementation der Individualisierungsmethode als Softwareanwendung behandelt.

A.4.1. Data-Set Vorbereitung

Wie in Sektion A.3 bereits erklärt wurde, sind die Ermittlung der $ITD(\theta, \phi)$ und die Extraktion von minimalphasigen Impulsantworten Voraussetzungen für die Verwendung der Individualisierungssoftware. Diese Aufgaben werden mit Hilfe von einem Matlab™ Skript durchgeführt (siehe Anhang B). Folgende Anforderungen dienen als Richtlinie für das Design der ITD-Individualisierungssoftware:

- Individualisierung in Echtzeit. Dynamische binaurale Systeme erfordern dass alle Prozesse in Echtzeit mit niedrigen Latenzen durchgeführt werden.
- Hoher SNR (signal to noise ratio).
- Die Bandbreite soll mindestens das menschliche Hörvermögen abdecken.
- Kompatibilität mit der Echtzeitfaltungseinheit FWONDER.
- Sample-fraktionale Zeitverzögerung.
- Konfigurationsskripte.
- Konfiguration über die Kommandozeile.
- Echtzeitsteuerung über OSC, GUI (graphical user interface) und Tastatur.

A.4.2. Softwarekomponenten

Die Softwareanforderungen aus dem vorigen Abschnitt wurden unter Verwendung folgender Komponenten erfüllt:

Jack Audio Server Sowohl die Kompatibilität mit der Echtzeitfaltungseinheit FWONDER als auch die Durchführung der Audioprozesse mit niedriger Latenz werden bei der Wahl von *Jack Audio* als Basis für die Echtzeitanwendung erfüllt. Dabei arbeitet der Audioprozess der ITD Individualisierungssoftware ähnlich wie ein *Plugin* wobei die synchronisierte Durchführung und Wiedergabe *Jack* überlassen wird.

Time-Stretching Die in C++ geschriebene Softwareanwendung verwendet Quellcode-offene Bibliotheken für die Durchführung bestimmter Aufgaben. Für das Wiedereinfügen der ITD mittels Time-Stretching wird die [Libsamplerate SRC API](#) (API, application programming interface) verwendet. Diese Programmierschnittstelle bietet Sinc- Interpolation mit einer Bandbreite von bis zu 97% , einem SNR von 97 dB und der Abarbeitung von Audiostreams.

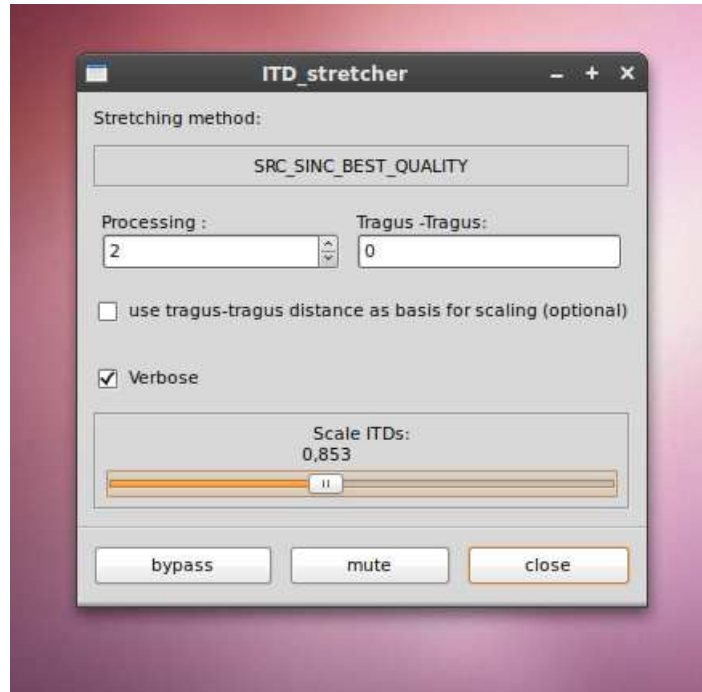


Abbildung A.6.: Grafische Benutzeroberfläche der ITD-I Software unter GTK+2.2

OSC Steuerung Der Headtracker am Kopfhörer sendet die aktuelle Kopfposition als OSC (open sound control) Nachrichten. Diese werden in FWONDER zur Aktualisierung der Filter und in der Software zur Aktualisierung der ITD verwendet. Die OSC-Nachrichtenübertragung wird mit Hilfe der [Liblo OSC API](#) durchgeführt.

XML Skripte Um die Kompatibilität mit FWONDER zu gewährleisten soll die ITD-I (ITD Individualisierungssoftware) in der Lage sein, FWONDER's Startparameter lesen und interpretieren zu können. Des Weiteren soll die Software sowohl die bei der Vorbereitung des Data-Sets erstellte Konfigurationsbeschreibung, sowie auch die im XML-Format vorliegenden ITD-I Konfigurationsskripte lesen können. Die Syntaxanalyse erfolgt hierbei mit Hilfe der [Libxml++](#) Library.

Grafische Benutzeroberfläche Um eine gängige Verwendung der ITD-I Software zu ermöglichen, wurde ein GUI (graphical user interface) mit dem *GTK+ Toolkit* entwickelt. Dieses Werkzeug ist unter der [GNU LGPL](#) Lizenz verfügbar und wird bei den meist verwendeten Betriebssystemen unterstützt. Abb. A.6 zeigt einen Screenshot in der Linux-Gnome Version.

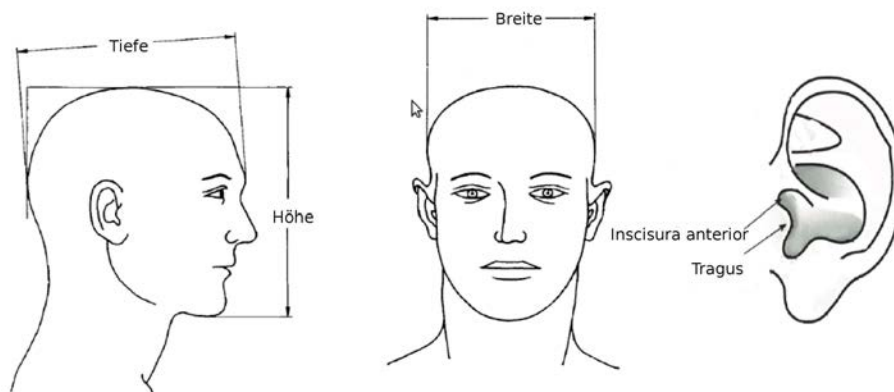


Abbildung A.7.: Relevante anthropometrische Maße für die Bestimmung der individuellen ITD

A.5. Anthropometrie-basierte ITD Individualisierung

Die korrekte Skalierung der fremden ITDs wird nur gewährleistet wenn diese in Verbindung mit den physikalischen Charakteristika des Höreranatomie gebracht werden kann. Dabei ist es wichtig anzumerken dass nicht alle Hörer in der Lage sind ihre eigene ITD akkurat zu skalieren. Deshalb wird eine Methode benötigt um diesen Schritt zu vereinfachen und die Anwendung der ITD-I Software in der Praxis zu erleichtern.

A.5.1. Hörversuch zur Ermittlung des individuellen ITD-Skalierungsfaktors

Vergleichbar zu Algazi's Ansatz (Algazi et al. 2001b) wurde ein Hörversuch entwickelt, um anthropometrische Maße der Versuchsteilnehmer mit einem Skalierungsfaktor in Verbindung zu bringen. Hierfür wurde an allen Teilnehmern Messungen von relevanten Kopfdimensionen durchgeführt. Es wurden die Kopfbreite, Kopftiefe und Kopfhöhe nach Norm DIN33402-2E (2005), sowie ergänzend die Kopfbreite anhand der Intertragusdistanz gemessen. Dieses Maß wurde aufgrund der, nahe dem Ohrkanal gelegenen und gut reproduzierbaren Position der incisura anterior gewählt. Abb. A.7 zeigt diese Maße.

A.5.1.1. Hörversuchsaufbau

In einem akustisch bedämpften Raum (Volumen = 155 m^3 , RT = 0.47 s) wurde ein binaurales Data-Set mit Hilfe des FABIAN HATS aufgenommen. Die Messung deckte einen

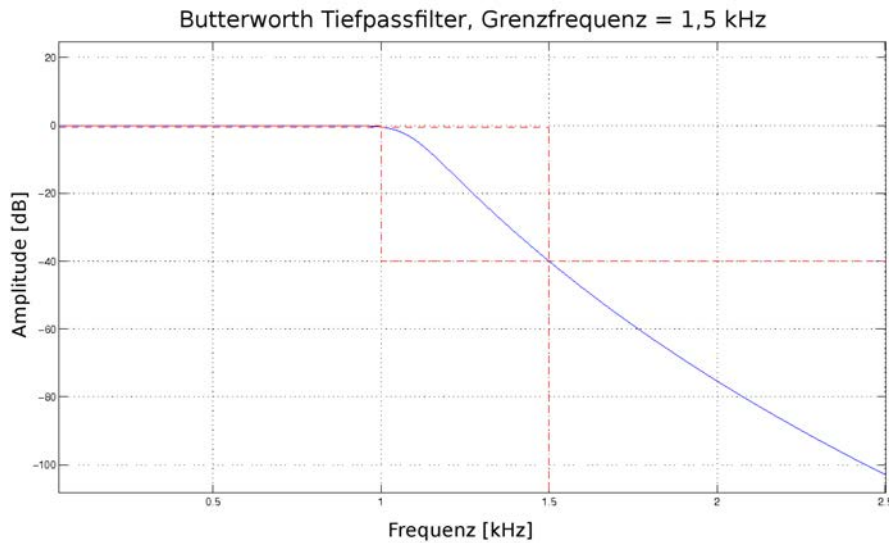


Abbildung A.8.: Frequenzgang des Tiefpassfilters zur Minimierung des Lokalisationseinfluss der ILD.

Bereich von 180° in der Horizontalebene ab und wurde mit 1° Auflösung durchgeführt (ohne Berücksichtigung der Elevation, bzw. Elevation = 0°).

Ein Lautsprecher Genelec 1030A, plaziert in 2 m Entfernung vom HATS bzw. vom Hörer, diente als Schallquelle für die Aufnahme und als Referenz während des Versuchs. Um den Einfluss der ILD auf die Lokalisation zu minimieren wurde als Stimulus eine Folge von Tiefpass-gefilterten ($\omega_g = 1.5kHz$) Impulsen weißen Rauschens verwendet, welche mit dem aufgenommenen Data-Set auralisiert wurden (siehe Abb. A.8). Für die Durchführung des Hörversuchs wurde eine Software mit folgenden Eigenschaften in C++ entwickelt:

- Umschaltung zwischen Referenzlautsprecher und Simulation mit Hilfe einer Steuerungsschnittstelle.
- Erzeugung von zufallsgenerierten Skalierungsfaktoren für jede neue ITD-Kalibrierung. Die Werte konnten zwischen dem 0.00- und 2.00-fachen der ITD vom Data-Set variiert werden.
- Speicherung des Hörversuchsergebnis in einem maschinell lesbaren Format (CSV).
- Überwachung des Hörversuchs über einen Computer im Netzwerk.

Als Steuerungsschnittstelle wurde eine Computertastatur verwendet, dessen für den Hörversuch relevanten Tasten entsprechend gekennzeichnet waren. Die Versuchsteilnehmer konnten damit den ITD-Skalierungsfaktor vergrößern bzw. verkleinern, zwischen Simulation und Referenz beliebig umschalten und den gewünschten Stand der Kalibrierung speichern.

A.5.1.2. Versuchsdurchführung

11 Versuchspersonen nahmen an diesem Hörversuch teil. Von allen wurden vor Beginn des Hörversuchs die o.g. anthropometrischen Maße erfasst. Zunächst wurde jede Versuchsperson auf die Instabilität von Schallquellen bei unpassender ITD aufmerksam gemacht. Um sicher zu stellen dass die Versuchsaufgabe und die Bedienung der Schnittstelle verstanden wurden, wurde den Versuchsteilnehmern für das Training beliebig viel Zeit zur Verfügung gestellt.

Die Aufgabe im Hörversuch bestand darin, mit Hilfe der Schnittstelle die virtuelle Schallquelle stabil und in Übereinstimmung mit der realen Schallquelle (aus dem Referenzlautsprecher) zu bringen. Es wurden von jedem Teilnehmer 10 Einstellungen von Kalibrierungsfaktoren erfasst.

A.5.2. Statistische Auswertung

Trotz des Trainings, und der deutlichen Hörbarkeit des Artefakts instabiler Schallquellen, war manchen Versuchspersonen der funktionale Zusammenhang und die Manipulation durch die variable ITD nicht klar zu machen. Durch Outliertests, Residualanalysen und nach Selbstaussagen wurden 2 der Versuchspersonen ausgeschlossen, so dass nur noch 9 der 11 Ergebnisse in die weitere Auswertung einfließen.

Abbildung A.9 zeigt die individuellen Verteilungen der je 9 hergestellten Skalierungsfaktoren als Boxplots. Die Medianwerte des Skalierungsfaktors der ITD überdecken einen Bereich von -5% bis +7%, individuell teils starke Streuungen sind erkennbar. Die korrigierte Varianzaufklärung zeigt, dass lediglich ein Prädiktor, die Intertragusdistanz, zur Vorhersage ausreicht. Sie zeigt eine Korrelation von $r = 0.84$ auf einem Signifikanz-Niveau von 0.5%, und erreicht damit auch einen höheren Zusammenhang als die noch bei Algazi verwendete Kopfbreite nach DIN. Das endgültige Vorhersagemodell wurde daher aus einer linearen Regression über die Intertragusdistanz bestimmt. Abbildung A.10 zeigt alle Messwerte, deren Standardabweichungen, die Regressionsgerade sowie die hyperbolischen 95% Konfidenzintervalle. Man erkennt den mit zunehmenden Abstand erwartungsgemäß ansteigenden Skalierungsfaktor.

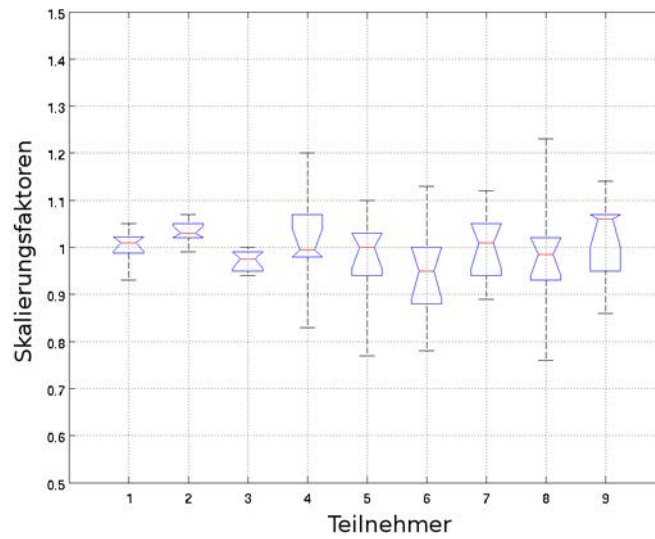


Abbildung A.9.: Verteilung der individuell hergestellten ITD-Skalierungsfaktoren von 9 Versuchsteilnehmern. Die großen Streuungen sind anzumerken.

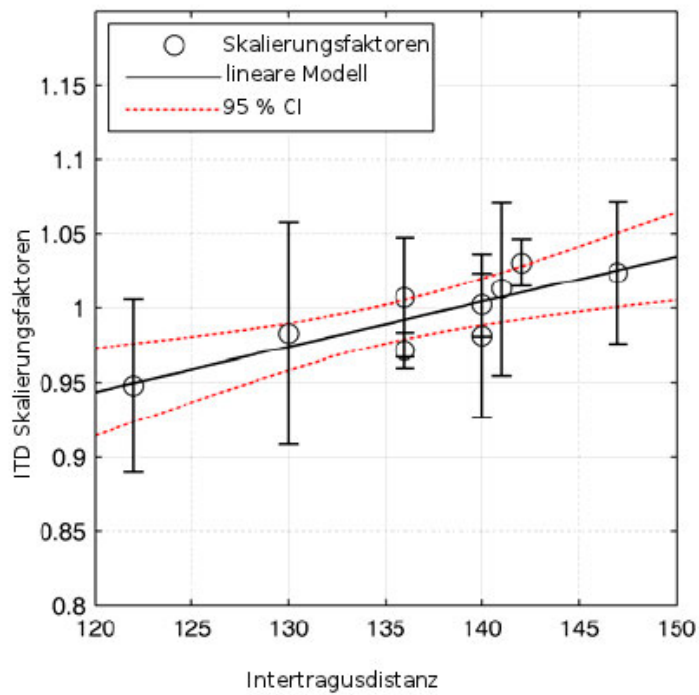


Abbildung A.10.: Modellierung der Hörversuchsergebnisse: lineares Regressionsmodell dargestellt über die Intertragusdistanz mit 95% Konfidenzintervallen.

A.5.3. Anthropometrie-basierte ITD-Individualisierungsformel

Die durch Regression ermittelte Formel für die individualisierte Skalierung der fremden ITD lautet:

$$S = 0.00304 \cdot d_i + 0.5792 \quad (\text{A.6})$$

mit d_i = Intertragusdistanz in mm.

An dieser Stelle ist wichtig anzumerken, dass dieses Modell nur für binaurale Data-Sets, die mit dem HATS FABIAN aufgenommen sind, gültig ist. Das Modell könnte zu arbiträren Data-Sets verallgemeinert werden, wenn der mit Gl. A.6 ermittelte Skalierungsfaktor mit dem Verhältnis der Intertragusdistanzen FABIAN / anderer Kunstkopf angepasst bzw. skaliert wäre. Dieser Aspekt wurde jedoch noch nicht untersucht.

A.6. Zusammenfassung und Diskussion

Diese Diplomarbeit befasste sich mit der Problematik der nicht-individuellen ITD in der Binauralsynthese. Es wurden sowohl eine Methode für die Individualisierung der ITD in Echtzeit sowie eine Softwareanwendung für die Anwendung behandelt bzw. entwickelt.

Bei der Methode handelt es sich um die Trennung der Impulsantworten in ihre spektralen und temporale Merkmale. Hierfür werden minimalphasige Impulsantworten aus einem binauralen Data-Set extrahiert und für die Wiedergabe der spektralen Cues mittels Echtzeitfaltung verwendet.

Ebenso wird die ITD als temporales Lokalisationsmerkmal aus dem Data-Set ermittelt und als Zeitverzögerung zwischen linkem und rechtem Ohrsignalen wieder eingefügt. Dies geschieht durch lineare Skalierung der ermittelten ITD mit einem individualisierten Faktor.

Eine empirische Formel für die Feststellung des geeigneten Skalierungsfaktors auf Basis von anthropometrischen Maßen wurde durch Regression ermittelt. Zu diesem Zweck wurde ein Hörversuch durchgeführt, durch den anthropometrische Maße mit den ITD-Skalierungsfaktoren von 9 Versuchspersonen in Verbindung gebracht werden konnten.

Ausblick

Die Methode zur Echtzeit ITD Individualisierung, die in dieser Arbeit dargestellt wurde, stellt eine Fortentwicklung in der binauralen Technologie dar, für sie bleiben jedoch noch

offene Fragen und Entwicklungsmöglichkeiten. Einige Aspekte die noch zu untersuchen sind wären:

- Gl. A.6 soll in einem Hörversuch und mit möglichst vielen Probanden perzeptiv untersucht und überprüft werden.
- Die Verwendung von Gl. A.6 mit Datensätzen anderer HATS als FABIAN über einen Korrekturfaktor der Intertragusdistanzen sollte ebenso in einem Hörversuch erforscht werden.
- Die minimale Auflösung für die Akquise eines Data-Sets im Fall der getrennten Behandlung von spektralen und temporalen Lokalisationscues, sowie Auswirkungen der Interpolation der erfassten ITDs ist noch nicht bekannt.
- Die Annahme, dass die frequenzunabhängige Skalierung der ITD ausreichend ist, könnte weiter erforscht werden.

B. Matlab code for data set preparation

The preparation of data sets was included as an option in the script for data set post-processing (after data acquisition) written by Alexander Lindau. This script was originally developed for shortening the HRIRs for its optimal use with *fwonder*, the real-time fast convolution software of the Audio Communication Institute of the TU Berlin.

```
%
% Mode=='B': extract & eliminate ITDs incl. time of flights
%
if Mode=='B'
    % 1st Run: find maximum time of flight in dataset
    for o= turntable_start:turntable_steps:turntable_stop
        post_position . rot=o;
        for p= elev_start:elev_steps:elev_stop
            post_position . elev=p;
            for q= azim_start:azim_steps:azim_stop
                post_position . azim=q;
                for r=1:postpro_setup . sources
                    post_position . source=r;

                    % Reconstruct formatted filename:
                    % [P/N_elev]_[P/N_azim]_[P/N_rot].wav
                    %... for instance: P00_N180_P00.wav
                    [elev,azim,rot]= postprocess_reconstruct_filename ;

                    % Create source filename
                    filename= strcat ( src_dir , '\source',num2str(post_position.source), '\',
                        elev,azim,rot, '.wav' )

                    % Read len values of IR
                    IR=wavread(filename);

                    % Use short IR for finding on set
                    IR_c=IR(1:len,:);

                    % Find maximum time of flight w. edge detect method
                    max_IR=max(abs(IR_c));
```



```

max_IR= repmat(max_IR,size(IR_c,1),1);
IR_c=IR_c./max_IR;
if size(IR_c,1) < size(IR_c,2)
    IR_c=IR_c';
end

%Upsampling
IR_up(:,1)=interp(IR_c(:,1),US);
IR_up(:,2)=interp(IR_c(:,2),US);

% for all channels, find position where onset threshold
value is reached
go_on=1; k=0; del(1)=1;
for i=1:size(IR_up,2)
    MAX=max(abs(IR_up(:,i)));
    % for full length of channel, find peak position
    while go_on
        k=k+1;
        if abs(IR_up(k,i)) > MAX*(onset_threshold)
            del(i)=k;
            go_on=0;
        end
    end
    go_on=1;k=0;
end

% Build matrix with onset indexes
onsetIndex_L(r,((p+abs(elev_start))/elev_steps)+1,((q+abs(
    azim_start))/azim_steps)+1)=del(1);
onsetIndex_R(r,((p+abs(elev_start))/elev_steps)+1,((q+abs(
    azim_start))/azim_steps)+1)=del(2);

% Calc & save ITD in [samples] per source from peak
positions
% ITD= tLeft - tRight
ITD_SAMPLES(r,((p+abs(elev_start))/elev_steps)+1,((q+abs(
    azim_start))/azim_steps)+1)=(del(1)-del(2))/US;

% save ITDs in microsec
ITD(r,((p+abs(elev_start))/elev_steps)+1,((q+abs(azim_start)
)/azim_steps)+1)=(del(1)-del(2))*1000000/(postpro_setup.
fs*US);

```

B. Matlab code for data set preparation

```
% check maximum time of flight...
if max(del)>max_time_of_flight(post_position.source)
    max_time_of_flight(post_position.source)=max(del);
end
% currently processed filename
clc
disp(['Time of Flight correction: Run 1/2 ',num2str(count),' of ',
     blanks(1),num2str(measurement_OverallNumber)]);
count=count+1;

    end
end
end
end

max_time_of_flight
pause(2)

count=1;

% Calculate length of dataset wo. times of flight
tmpIR = interp(IR(:,1),US);
[sizeIR,b] = size(tmpIR);
new_len=sizeIR-max_time_of_flight(post_position.source)+1;
%
% 2nd Run: remove time of flight & shorten dataset to new minimum duration
%
for o=turntable_start:turntable_steps:turntable_stop
    post_position.rot=o;
    for p=elev_start:elev_steps:elev_stop
        post_position.elev=p;
        for q=azim_start:azim_steps:azim_stop
            post_position.azim=q;
            for r=1:postpro_setup.sources
                post_position.source=r;

                % Reconstruct formatted filename:
                % [P/N_elev]_[P/N_azim]_[P/N_rot].wav
                %...for instance: P00_N180_P00.wav
                [elev,azim,rot]=postprocess_reconstruct_filename;

                % Create source filename
```

B. Matlab code for data set preparation

```
filename=strcat(src_dir, '\source', num2str(post_position.
    source), '\', elev, azim, rot, '.wav')

IR=wavread(filename);

% Eliminate ITD but insert a common peak secure margin
% Upsample the hole BRIR and find the size of it
left_up=interp(IR(:,1),US);
right_up =interp(IR(:,2),US);
IR_all_up=[left_up,right_up];

% look for onset indexes
indexL = onsetIndex_L(r,((p+abs(elev_start))/elev_steps)
    +1,((q+abs(azim_start))/azim_steps)+1);
indexR = onsetIndex_R(r,((p+abs(elev_start))/elev_steps)
    +1,((q+abs(azim_start))/azim_steps)+1);

if USE_PSM ==1
% create new dest. vector
IR_e=zeros(new_len+PSM,2);

% store the amount of samples
IR_e(:,1) = IR_all_up(indexL-PSM : new_len + indexL -1,1);
IR_e(:,2) = IR_all_up(indexR-PSM : new_len + indexR -1,2);
else
% create new dest. vector
IR_e=zeros(new_len,2);

% store the amount of samples
IR_e(:,1) = IR_all_up(indexL: new_len + indexL -1,1);
IR_e(:,2) = IR_all_up(indexR: new_len + indexR -1,2);
end

% Downsample
IR_down(:,1) = downsample(IR_e(:,1),US);
IR_down(:,2) = downsample(IR_e(:,2),US);

% Rewrite to file in specified format
filename=strcat(dest_dir, '\source', num2str(post_position.
    source), '\', elev, azim, rot, '.wav')
wavwrite(IR_down,postpro_setup.fs,postpro_setup.bits,
    filename);
```

B. Matlab code for data set preparation

```
        % currently processed filename
        clc
        disp(['Time of Flight correction: Run 2/2 ', num2str(count), ' of ',
            blanks(1), num2str(measurement_OverallNumber)]);
        count=count+1;

    end
end
end
end

% Write txt file with ITDs in microseconds and separated by a ';'
filename=strcat(dest_dir, '\source', num2str(post_position.source), '\', 'ITDs.
    txt');
dlmwrite(filename, ITD, ';');
disp(['ITDs saved in ', filename]);

% Save a copy of the ITDs in Matlab format
tmpITD(:, :) = ITD(1, :, :);
savefile = 'D:\Archivo\SVN_Lindau\4 dataset manipulation\testITD.mat';
save(savefile, 'tmpITD')

% Write xml file with data set information
BRIRinfo=[];
BRIRinfo.Elevation.ATTRIBUTE.elev_start = postpro_setup.elev_start;
BRIRinfo.Elevation.ATTRIBUTE.elev_end = postpro_setup.elev_stop;
BRIRinfo.Elevation.ATTRIBUTE.elev_resolution = postpro_setup.
    elev_stepwidth;
BRIRinfo.Elevation.ATTRIBUTE.elev_interpolation = 0; % not used at this
    time
BRIRinfo.Azimuth.ATTRIBUTE.azim_start = azim_start;
BRIRinfo.Azimuth.ATTRIBUTE.azim_end = azim_stop;
BRIRinfo.Azimuth.ATTRIBUTE.azim_resolution = postpro_setup.azim_stepwidth;
BRIRinfo.Azimuth.ATTRIBUTE.azim_interpolation = 0; % not in use for now
xml_write(strcat(dest_dir, '\source', num2str(post_position.source), '\', '
    BRIRinfo.xml'), BRIRinfo);
disp(['Info file in xml generated: ']);
type(strcat(dest_dir, '\source', num2str(post_position.source), '\', 'BRIRinfo.
    xml'))

% Return value
ToF = [];
end % end of Mode "B"
```

C. ITD-Individualizer usage

The following instructions can be found in the README.txt file, accompanying the source files.

C.1. Installation

C.1.1. Dependencies

make sure you have the development files of the following components installed on your system:

liblo

libsamplerate

libxml++-2.6

gtk+-2.0

jackd1 or **jackd2**

fwonder

tracker

C.1.2. Compilation

For compiling just open a console, go to the stretcher's main directory and type: **scons**

C.2. Usage

Starting the program:

For starting the software you have to first configure the ITD stretcher, there are some basic settings that you must define before starting. Some of them may not be changed (without re-starting), and other may be modified in real time (here marked as *rt*).

The settings may be loaded from an XML configuration file, or via command line options. There is a template with the name "stretcher_config.xml" on the main directory. See `./doc/ITD_stretcher_params.ods` for details on the options.

The syntax for starting is:

`./stretcher -[option1] [parameter1] -[option2] [parameter2]...`

The basic settings are:

Jack client name: the jack server doesn't allow clients to have identical names. You have to remember this if you are starting more instances of the program. Any number of numbers or letters are allowed but no blank spaces.

Path to fwonder configuration file: the stretcher needs to know the auralisation engine's setup (ranges of operation, etc).

OSC listening port: for receiving the current head's position and some control parameters.

Source number: a number according to the audio source you are currently using. (1,2,etc). The stretchers looks in the corresponding source folder for other configuration files that are stored in there (BRIRinfo.mxl and ITD.txt).

IMPORTANT NOTE: In order to maintain compatibility with fwonder, every source in a BRIR dataset should have a "source1" folder, therefore, sources will always have to be named "source1". An incrementation over source is not needed, so please use that folder name for all your sources and start a separate fwonder engine for each of them.

SRC modus: this is a number from 0 to 5, there are 5 stretching (interpolation) algorithms available. 0 has the best quality but requires lots of processing power. 5 is the cheapest, but the quality suffers. Recommended is one of the first three because they use SINC interpolation.

For more info see: http://www.mega-nerd.com/SRC/api_misc.html#Converters

processing: a number from 1 to 10. Here you define how fast stretching from one delay to another takes place. The number you give is the number of audio chunks the stretching lasts. (Example: suppose you are using a 512 buffer. processing = 3 means the stretching to the new delay takes 3*512 samples) (*rt*)

scale factor: a number (32-bit float) between 0 and 2. This is the ITD scaling factor you want to use. (*rt*)

user_tragus: the intertragus distance in millimeters (float).(*rt*)

data_tragus: the intertragus distance of the artificial head in millimeters (float).

IMPORTANT: if you set “user_tragus = 0” the empirical formula will be skipped and “scale factor” will directly scale the ITD submitted with the data. Otherwise “scale factor” gets overridden at startup by the scale factor according to the formula:

$$\text{scale factor} = 0.4495 \cdot \text{user_tragus}/\text{data_tragus} + 0.5792$$

Configuration priority

- When starting the program, command-line arguments may override the settings in the submitted config-file if the parameters are added AFTER the config-file path.
- If neither a config-file path nor command-line arguments are submitted, then the program gives you an error message and stops.

Here are a couple of examples, Example 1:

```
./stretcher -c config.mxl -o 56789
```

Here the OSC port will be configured to 56789 no matter what the xml file says.

Example 2:

```
./stretcher -o 56789 -c config.xml -u 130
```

In this case the OSC port will be set to whatever the config file says (56789 gets overridden) and finally the intertragus distance will be 130 (overriding the config file).

Additional starting options that do not require a parameter are:

- -v –verbose for displaying more information (like the actual scaled ITD).
- -g –gui for starting the Graphical User Interface (GUI).
- -h –help if you want to see the options.

Controlling the ITD-stretcher

You can modify parameters in real time on the ITD stretcher via, OSC messages, hotkeys or a GUI. (or using the Multistretcher. See the documentation of that program for further details.)

Via OSC:

/itd_individualizer/quit for closing the application.
/itd_individualizer/mute 1 for muting the application; 0 to unmute.
/itd_individualizer/bypass 1 for bypass on; 0 for bypass off.
/itd_individualizer/scale for changing the ITD scaling factor (float) between 0-2.

NOTE: The following messages are not supposed to be send by the user but for the head-

/WONDER/tracker/move/pan azimuth angle in degrees (float).
tracker. */WONDER/tracker/move/tilt* elevation angle in degrees (float).
/WONDER/tracker/move azimuth, elevation and rotation (not implemented).

Via hotkeys:

If you don't start the GUI the control loop looks for key events. The following keys can be

q(Q) for closing the program.
m(M) for muting/unmuting.
used: **x(X)** for bypass on/off.
v(V) for verbose on/off.
+ for increasing the scaling factor on 0.01
- for decreasing the scaling factor on 0.01

all of them give visual feedback on the console.

Via GUI:

with the GUI you can change:

- The 'processing' interval a number from 1 to 10. Here you define how fast the stretching takes place.
- The intertragus distance in millimeters. You have to activate the corresponding checkbox before you can write on it. The text field takes floats between 0 and 300 mm. After you hit "enter" the scaling factor changes according to the regression formula.
- Verbose, check box for verbose on/off
- ITD-scaling factor using a horizontal fader. Goes from 0-2.
- Mute, toggle button for muting/unmuting
- Bypass, toggle button for bypassing or not.
- Close, for closing the application.

D. A software for controlling multiple ITD individualizers

In the case of multiple source-rendering it is sometimes required to control all instances of the ITD individualizer at the same time with the same parameters to preserve coherence. An application example could be the rendering of a string quartet in a performance, where every instrument would be processed separately.

Since the ITD individualizer can be controlled via OSC protocol, a standalone application was developed to provide this feature.

The parameters that can be controlled are:

- ITD scaling factor (controllable through a slider)
- Mute: output of all ITD-Is is set to 0.
- Bypass: input of ITD-I is reflected at its output.

Figure [D.1](#) shows GUI of the application as well as an example of a configuration file for the communication with 8 clients.

Once again the *liblo* API was used for OSC interfacing.

Multistretcher usage

The following instructions are provided in a README.txt file provided together with the source files.

Installation

Dependencies

Make sure you have following components installed on your system:

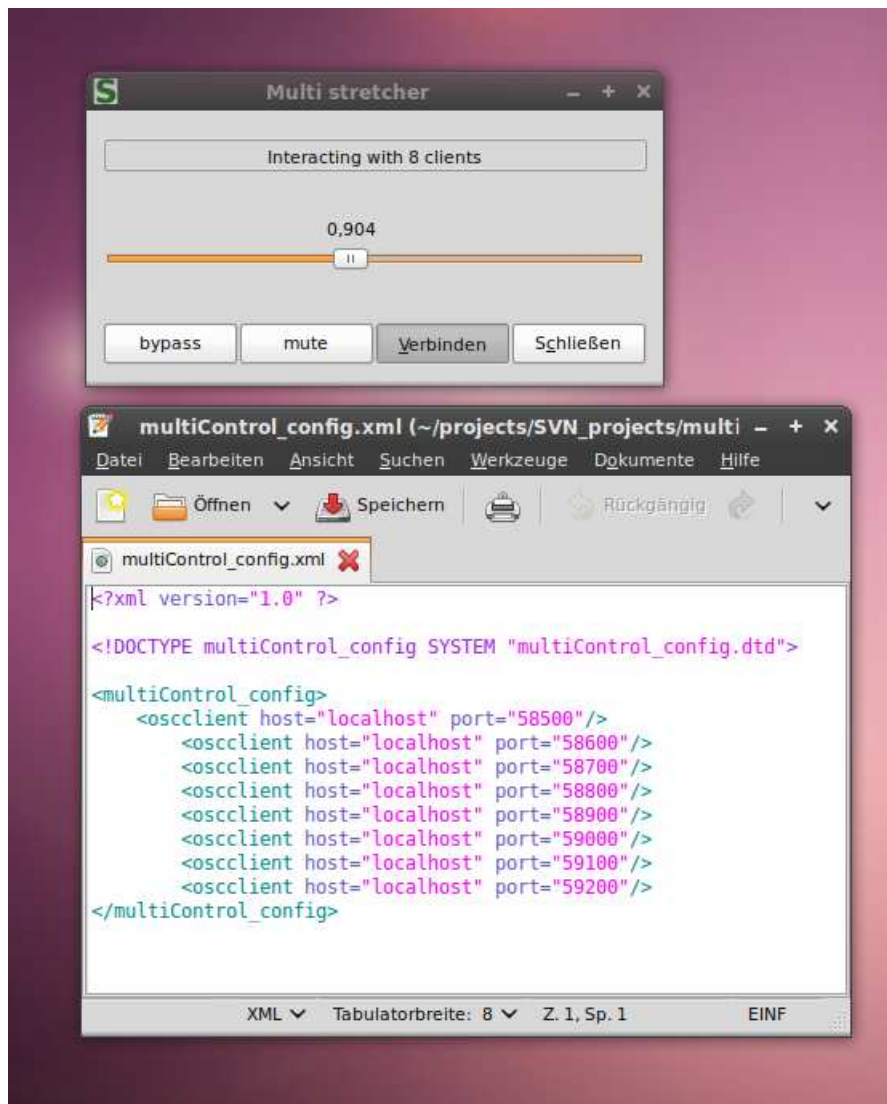


Figure D.1.: (up) Application developed to control multiple instances of the ITD individualizer. (down) Configuration file for determining the OSC ports to transmit commands to.

liblo-dev

libxml++-2.6

gtk+-2.0

gmodule-2.0

libglade-2.0

Compilation

For compiling just open a console, go to the program's main directory and type: **scons**

Starting the software

The Multi-Stretcher is simply an OSC interface for controlling multiple "Stretcher" instances (For more info on the ITD-Stretcher see the Stretcher's README.txt file). Before you start it you must first define the OSC clients (the OSC listening ports of the Stretchers) that should receive the messages. For this purpose edit or use as a template the "multiControl_config.xml" file located in the "config" folder according to the following specifications:

host IP address of the receiving OSC client (Stretcher). You can type "localhost" (or 127.0.0.1) if the Stretcher is running on the same computer.

port The OSC listening port that the Stretcher is using.

Note: you should do this for EVERY Stretcher you want to communicate to.

Now that you have a proper configuration file you can start the Multi Stretcher by calling:

./multi

if you are using the configuration file from the "config" folder, or:

./multi -c /pathToConfigFile/multiControl_config.xml

if you use a configuration file from another location.

Usage

On the GUI you will find a legend with the number of instances receiving the OSC messages. This number should be equal to the number of OSC clients specified in the configuration file.

A horizontal fader (goes from 0.0 to 2.0) for controlling the ITD scaling factor on all Stretchers.

And four buttons:

bypass Bypass all stretchers

mute Mute all stretchers

connect This toggle button should be pressed (default) for sending the messages.

close Close the application. Please note that pressing this button does not close any of the ITD-Stretchers.

Comments, bug reports and requests can be send at *jorgose@gmx.de*

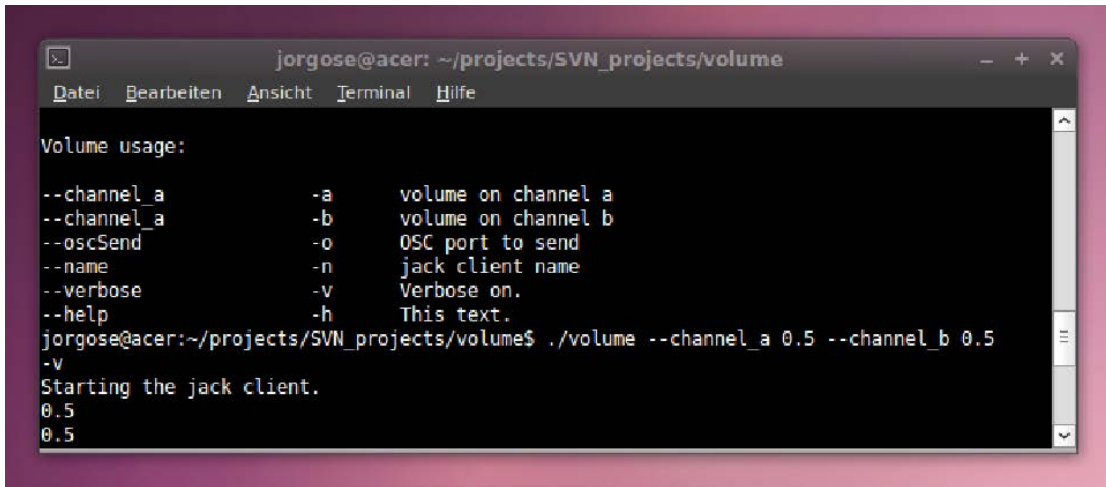
E. A software to control the volume of audio streams

For the listening test realized it was necessary to have a start-script that runs all software applications with the required parameters. One of these parameters was the output volume.

In normal use the user of the auralization system can adjust the volume at the headphones according to his/her needs, so *fWonder* and the ITD-S do not require this feature; however, for the listening test described in this work the user should not change the volume of the reproduction, since this would alter the calibrated same-loudness impression between real loudspeaker and simulation, thus, introducing an unwanted discrimination criteria. Therefore it was decided to pre-set it.

For this task a simple audio application that controls the volume of audio streams (by means of multiplication with a factor) was developed. The software is a *Jack-client* that can be configured at start-up with the desired volume and can optionally send OSC messages with the value of the faders.

Figures [E.1](#) and [E.2](#) show the help file of the application and the GUI of the software respectively.



```
jorgose@acer: ~/projects/SVN_projects/volume
Datei Bearbeiten Ansicht Terminal Hilfe

Volume usage:
--channel_a      -a      volume on channel a
--channel_b      -b      volume on channel b
--oscSend        -o      OSC port to send
--name           -n      jack client name
--verbose        -v      Verbose on.
--help          -h      This text.
jorgose@acer:~/projects/SVN_projects/volume$ ./volume --channel_a 0.5 --channel_b 0.5
-v
Starting the jack client.
0.5
0.5
```

Figure E.1.: Help file of an application to pre-set volume on audio streams.

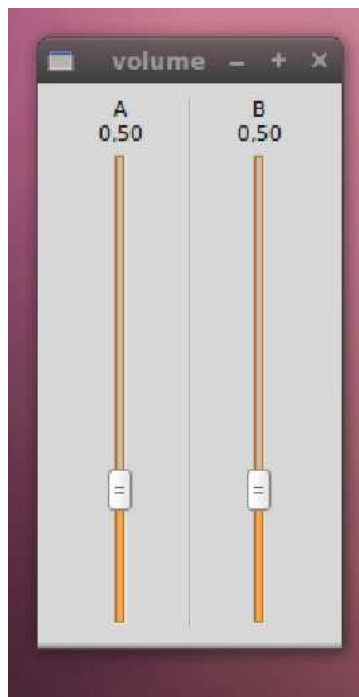


Figure E.2.: GUI of the software application to pre-set and modify volume.

F. Audibility of Doppler effect due to head rotation

As mentioned in Chapter 1 the time stretching implementation that reconstructs the ITD in the proposed individualization model resembles the Doppler effect mechanism as it occurs in natural hearing. This section reviews quantitatively the Doppler effect's audibility due to head rotation.

Assuming an spherical head with ears located at opposite positions of the sphere's surface, head rotation with an angular speed of $\omega = 1000^\circ$ per second (Lindau 2009) and a head radius of $r = 0.085$ m, the linear speed v of one ear can be computed as:

$$v = r \cdot \omega \quad (\text{F.1})$$

thus:

$$v = \frac{1000^\circ}{360^\circ} \cdot 2 \cdot \pi \cdot r = 1.484 \text{ m/s} \quad (\text{F.2})$$

The frequency modulation due to Doppler effect (Weinzierl 2004) can be computed as:

$$F_m = F_0 \cdot \frac{1}{1 + \frac{v}{c}} \quad (\text{F.3})$$

where c is the speed of sound = 343 m/s at 20° C.

Considering a frequency of 1 kHz we have a frequency modulation of:

$$\Delta F = |1 \text{ kHz} - F_m| = 4.3 \text{ Hz} \quad (\text{F.4})$$

According to Zwicker and Fastl (1999) the just noticeable difference in frequency modulation for 1 kHz is about 6 Hz, meaning that even for extremely fast head movements the Doppler effect still falls below the audibility threshold.