Piotr KRUCZKOWSKI, **Tomasz MĄKA**
FACULTY OF COMPUTER SCIENCE AND INFORMATION TECHNOLOGY
WEST POMERANIAN UNIVERSITY OF TECHNOLOGY, SZCZECIN
52 Zolnierska St., 71-210 Szczecin, Poland

LibLaura: A Library for Binaural Sound Source Localization

Abstract

In this work the software library for binaural sound localization is presented. The main purpose of the library are the applications for localization tasks in audio systems based on two microphones. The implemented mechanisms include binaural single sound source localization, ITD (Interaural Time Difference) and ILD (Interaural Level Difference) cues and support real-time analysis. LibLaura is written in C++ language and is easily extensible to support new features, time delay estimators and user-defined callbacks.

Keywords: software library, binaural localization, time delay estimation.

1. Introduction

Sound localization using two microphones has become popular in many robot audition systems. Therefore, in recent years many software and hardware solutions have been proposed. The localization is one of the cues for attention-based processing and affective computing. It is also important for effective implementation of voice-based interaction between human and robot. The estimation of robot's attention to audio events using sound localization scheme plays important role in natural navigation and interaction processes. The obstacles for robust sound localization occurs in the properties of acoustical environment. The interference and diffraction of sound waves can deteriorate the localization capabilities.

Human can locate the sound source in the horizontal and vertical planes with two ears only. This is due to the properties of the human hearing system where the differences of time, intensity and spectral content of sound waves reaching both ears are used in the brain to determine the localization of sound source.

A common approach to find out the approximate localization of sound source in horizontal plane is based on the estimation of the time differences of arrival between microphones. The localization scheme using time delay estimation techniques is based on the assumption that sound wave propagates along a single path to the microphone.

2. Binaural localization

The most of cues for sound localization depends on comparison of signals coming to both ears. The basic binaural localization is based on TDOA (Time Delay of Arrival) technique. Such approach is used to determine the angle in horizontal plane of sound source captured by two microphones. The illustration of planar sound wave propagation from point source to two ears is presented in Fig. 1.

According to this figure, the angle can be calculated using the following relation [8]:

$$\Delta t \cdot c = d \cdot \sin(\alpha), \tag{1}$$

where: Δt is the time delay of approaching sounds between both microphones, c is the speed of sound (c $\cong 342\ \frac{\mathrm{m}}{\mathrm{s}}$ in dry air) and d denotes the distance between microphones. Then, the angle of sound source can be determined as:

$$\alpha = \operatorname{arc}\sin\left(\frac{\Delta t \cdot c}{d}\right). \tag{2}$$

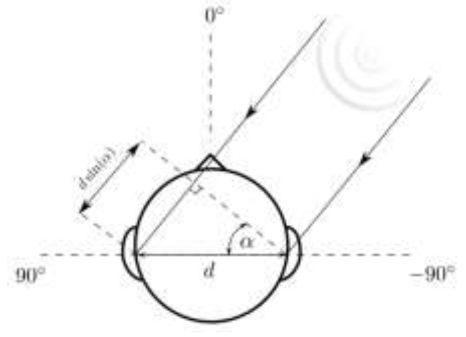


Fig. 1. Planar sound wave propagation from point source to the ears

In order to calculate the time difference between two signals, a delay estimators are used. The commonly used technique is based on the search for extrema in cross-correlation function calculated for both signals. The most popular estimators include the following correlation functions:

1. Cross-Correlation

$$g(k) = \sum_{n=0}^{N-1-k} x_L(n) \cdot x_R(n-k), \tag{3}$$

   where: x_L, x_R are signals acquired from the microphones.

2. Generalized Cross Correlation with Phase Transform (GCC-PHAT) [7]

$$g(k) = F^{-1}\left(\frac{X_L \cdot X_R^*}{|X_L \cdot X_R^*|}\right), \tag{4}$$

   where: X_L,X_R are frequency domain representations of x_L and x_R respectively, $F^{-1}()$ denotes inverse Fourier transform and X^* is the complex conjugate.

The position of maximum value of $g(k)$ indicates the time delay between x_L and x_R signals:

$$\Delta t = \arg\max\ [g(k)]. \tag{5}$$

In case of human head the time delay between signals arriving to the two ears the delay can change between 0 and 690 μs [8]. Due to the front-back ambiguity such technique can determine the azimuthal angle from -90° to 90° only (Fig. 1). This problem can be overcome using pinna amplification effect [6].

3. Software implementation

In order to use the basic mechanisms of binaural localization in real conditions we have designed and developed a library in C++ object-oriented language. The library is licensed under the GNU Public License v2 and can be obtained from GIT repository[1]. The proposed library (called LibLaura) can be compiled on platforms which have the POSIX compliant C++ compiler.

[1] http://github.com/Kruczkowski/LibLaura

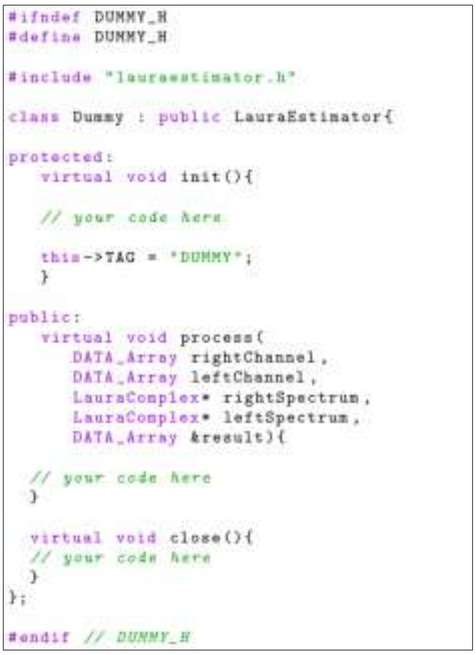
The library processes internally the input signals by a frame-by-frame scheme and provides their representations in time and frequency domains. It uses dedicated library for signal conversion to frequency domain. Additionally, all input/output operations are handled by external libraries. The project relies on the following libraries: 1) FFTW [5] – efficient Fast Fourier Transform calculation (`http://www.fftw.org`), 2) RtAudio [9] – real-time audio input/output access and 3) libsndfile [4] – reading and writing files containing sampled sound.

The FFTW library is used for amplitude spectrum and GCC-PHAT calculation. Thanks to the RtAudio and libsndfile libraries the two type of sound sources are supported: real-time reading audio data from the device and from the audio file for offline processing. In order to perform real-time analysis of audio stream, at the initialization stage, the properties of the acquisition have to be defined and then the method `capture()` starts the process of calculating the localization data. Processing in the offline mode requires to call the `play()` method for calculating data and wait to finish the processing by checking the status using `isStreamRunning()` method.

The details of the library usage are described in two simple examples available in '`examples/`' directory. The calculated data can be obtained using user-defined callback where the types of returned cues are defined by the using flags. In addition to binaural cues, user can process the available stream using its time and frequency representations for both channels.

The functionality of the LibLaura can be easily extended by adding new features for analyzed audio stream using new code in the LauraCallback class. Also, new time delay estimators can be implemented in the library using '`dummy.h`' template in the '`modules/`' directory. The example code skeleton is depicted in Fig. 2 - the `init()` and `close()` methods are called only once while the `process()` method is called for every frame. New estimator has to be included into the project and initialized using `setEstimator()` method.

The availability of time and frequency representations of the audio data in the callbacks and estimators allows for easy implementation of a new mechanisms for binaural analysis and processing.



```
#ifndef DUMMY_H
#define DUMMY_H

#include "lauraestimator.h"

class Dummy : public LauraEstimator{

protected:
    virtual void init(){

    // your code here

    this->TAG = "DUMMY";
    }

public:
    virtual void process(
        DATA_Array rightChannel,
        DATA_Array leftChannel,
        LauraComplex* rightSpectrum,
        LauraComplex* leftSpectrum,
        DATA_Array &result){

   // your code here
   }

   virtual void close(){
   // your code here
   }
};

#endif // DUMMY_H
```

Fig. 2. Time-delay estimator template

At the current state the library has the following features:

- Determination of azimuthal angle in horizontal plane of single sound source.
- Cross-correlation and GCC-PHAT time delay estimators are supported.
- New time delay estimators can be added easily to the library core.
- ITD (Interaural Time Difference) and ILD (Interaural Level Difference) cues [2] calculation.
- User-defined callback in the processing chain.
- Easy to use with simple and ready to use examples.
- Sound sources are supported using audio file or audio device in real-time.

4. Experiments

The experiments have been performed on the Lenovo Z580 machine with 4 cores Intel processor i7-3520M. The data were recorded using Digidesign Focusrite Mbox audio interface and we used two Behringer C-2 matched condenser microphones in our setup.

In the experiments a sample recording of single noise source was used and its time-frequency structure is presented in Fig. 2. The position of sound source was changed in horizontal plane with azimuthal angle from 90° to -90° and then again to 90°.

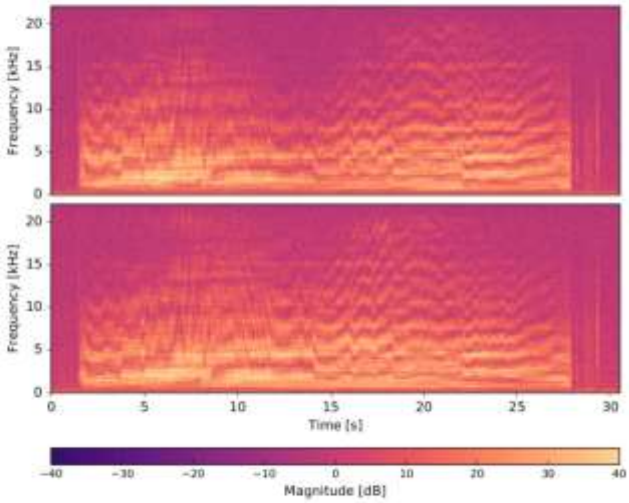


Fig. 3. Spectrograms of the both channels of test signal

At the first stage we have determined the accuracy of azimuthal angle for the test signal. The comparison of actual angle and estimated angle with LibLaura is depicted in Fig. 3. The highest inaccuracies were observed while the sound source changed the direction of rotation.

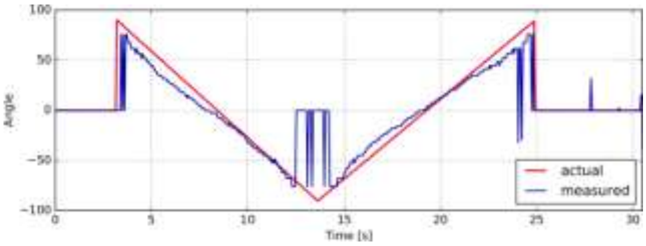


Fig. 4. Actual and measured angle trajectories

In the second experiment we have compared the accuracy of azimuthal angle estimation using LibLaura and SoundLocalizer [3]. We have generated short sound at different angles at the distance from our setup equal to one meter. The angles have changed from 0° to 90° with step equal to 5° (the number of measurements was equal to N = 19). For the actual and measured angles, a set of absolute errors have been calculated:

$$\epsilon = \left\{ \left| \alpha_{\text{actual}}^{(i)} - \alpha_{\text{measured}}^{(i)} \right| : i = 1, \dots, N \right\} \tag{6}$$

and minimum, maximum and mean values were computed and shown in Tab. 1. The slightly better accuracy was obtained in case of LibLaura.

Tab. 1. Accuracy of angle estimation

	ϵ_{min}	ϵ_{max}	$\overline{\epsilon}$
LibLaura	0	5.7	1.4675
SoundLocalizer	0.1	5.7	1.4844

Because the audio stream is processed on a frame-by-frame basis, we performed an analysis how the frame size influences the angle estimation accuracy. For this purpose, we have changed the frame size from 10 ms to 50 ms and compute MSE (Mean Squared Error) using the test signal depicted in Fig. 3. The results are shown in Fig. 5 where the highest accuracy was obtained for frame size equal to 20 ms.

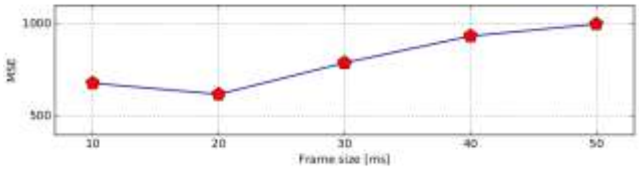


Fig. 5. Mean squared errors for different frame sizes

Finally, we measured the computation time and memory usage for our library and SoundLocalizer. The data is presented in Tab. 2. Since SoundLocalizer is written in Java language where the key element of the performance is the Java virtual machine (JVM), the used resources are much higher than in our C++ implementation. In case of the BSS Locate toolbox [1], the number of resources is much higher due to required MATLAB environment (about 14 times slower and needs 15 times more memory than LibLaura).

Tab. 2. Computational resources usage

	Computation time µs	Memory usage MB
LibLaura	380	27
SoundLocalizer	800	70

5. Conclusions

In this work an open source software library dedicated to determine the azimuth angle of sound source in horizontal plane is presented. The library supports two basic time-delay estimators, can compute binaural cues, capture signals from audio devices or audio files and is easily extensible in terms of new functionality. The user can implement own callbacks called per frame of audio data and add custom time-delay estimators. Despite the fact that it supports only two microphones, its size, speed of calculations and high degree of customization makes it suitable to be used in the hardware platforms with limited resources.

6. References

[1] Blandin C., Vincent E. and Ozerov A.: BSS locate – a toolbox for source localization in stereo convolutive audio mixtures. Matlab toolbox, available at http://bass-db.gforge.inria.fr/bss locate/, 2016.

[2] Blauert J.: Spatial Hearing – Revised Edition: The Psychophysics of Human Sound Localization. The MIT Press, 1996.

[3] Calmes L.: Biologically Inspired Binaural Sound Source Localization and Tracking for Mobile Robots. PhD thesis, RWTH Aachen University, Luxemburg, December 2009.

[4] de Castro Lopo E.: libsndfile – a C library for reading and writing sound files containing sampled audio data. software library, version 1.0.26, available at http://github.com/erikd/libsndfile/, November 2015.

[5] Frigo M.: A Fast Fourier Transform compiler. In ACM SIGPLAN Conference on Programming Language Design and Implementation – PLDI'1999, pages 642–655, Atlanta, Georgia, USA, May 1–4 1999.

[6] Kim U. H., Nakadai K. and Okuno H. G.: Improved sound source localization and front-back disambiguation for humanoid robots with two ears. In 26th International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems – IEA/AIE'2013, pages 282–291, Amsterdam, The Netherlands, 2013.

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

[8] Moore B. C. J.: An Introduction to the Psychology of Hearing. BRILL, 6th edition, 2013.

[9] Scavone G. P.: RtAudio – a set of C++ classes that provide a common API (application programming interface) for realtime audio input/output. software library, version 4.1.2, available at http://www.music.mcgill.ca/~gary/rtaudio/, February 2016.

Received: 05.09.2016 *Paper reviewed* *Accepted: 02.11.2016*

Piotr KRUCZKOWSKI, eng.

He received the BSc degree in computer science from the West Pomeranian University of Technology, Szczecin. His scientific interests include sound processing and software engineering.

e-mail: kruczkowski@gmail.com

Tomasz MĄKA, PhD, eng.

He received the MSc and PhD degrees in computer science from the Szczecin University of Technology in 2000 and 2005, respectively. His research focuses on the auditory scene analysis and audio signal processing techniques.

e-mail: tmaka@wi.zut.edu.pl