REALTIME BINAURAL SPEECH ENHANCEMENT DEMO ON RASPBERRY PI

Masoumeh Azarpour, Jan Siska, and Gerald Enzner

Institute of Communication Acoustics and Adaptive Systems Laboratory
Ruhr-Universität Bochum, Germany
{masoumeh.azarpour, jan.siska, gerald.enzner}@rub.de

ABSTRACT

We demonstrate the feasibility of the realtime implementation of advanced binaural noise reduction algorithms in a single-chip computer called Raspberry Pi. The implementation of the considered algorithms is realized in Simulink, a graphical programming add-on to the integrated development environment Matlab. Using a complementary support package for Simulink, the Raspberry Pi is connected/hosted. The implemented binaural noise reduction algorithm comprises two stages. First, the noise power spectral density (PSD) is estimated by one of the speech blocking-based noise PSD estimators previously proposed. The adaptively estimated noise PSD in each frame is then employed in a cue-preserving MMSE-based noise-reduction spectral gain function. The objective of this demonstration is to present the capabilities of the proposed algorithms with the application for hearing aids in a challenging noisy environment, i.e., congress babble noise in the show-and-tell area. The proposed solution suppresses the noise without having prior information on noise statistics, target speaker location and voice activity detection (VAD). Moreover, we would like to exhibit the powerful solution entirely executed with low-cost hardware.

Index Terms— Binaural noise reduction, binaural cue preservation, Raspberry Pi, Simulink

1. BINAURAL NOISE REDUCTION ALGORITHM

In this contribution, we present a real-time demonstration of binaural noise reduction (NR) algorithms. Figure 1 illustrates the schematic block diagram of the system. The noisy microphone signals $y_i(k)$, with $i \in \{r, l\}$, captured at sampling time index k, can be expressed as $y_i(k) = s(k) * h_i(k) + n_i(k) = x_i(k) + n_i(k)$, where $s(k)$, $h_i(k)$, and $n_i(k)$ are the target speech, binaural room impulse responses, and the ambient background noises, respectively. In the first stage, the noise power is estimated employing the target speech cancellation techniques. Three different noise PSD estimators have been implemented for this contribution [1–3]. The estimated noise PSD is then employed in a cue-preserving MMSE filter [4] to retrieve the microphone signals. In this Section, the noise reduction algorithm will be explained shortly. The demonstration setup will be then introduced in Sec.2.

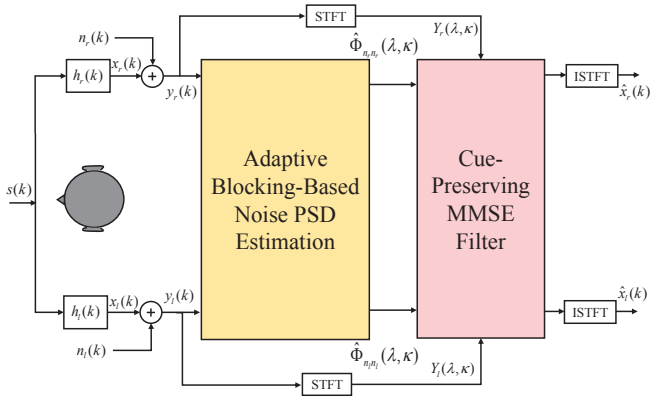

Fig. 1: Schematic block diagram of the NR system

1.1. Noise PSD Estimators

At first, the noise PSD estimators will be briefly introduced along with references to original works.

ITF-based Adaptive Blocking (ITFB): An interaural transfer function (ITF)-based estimator of the noise PSD $\Phi_{n_i n_i}$ (Fig. 1.a) with $i \neq j \in \{l, r\}$ reads [1]

$$\widehat{\Phi}_{n_i n_i} = \frac{\widehat{\Phi}_{e_i}}{1 + |\widehat{W}_j|^2 - 2\text{Re}\{e^{j\frac{2\pi}{M}\lambda\tau_a}\widehat{W}_j\Gamma_{n_i n_j}\}}, \quad (1)$$

where $\widehat{W}_j$ and $\widehat{\Phi}_{e_i}$ denote the estimated interaural transfer functions and the PSD of the left and right adaptive filter error signal $e_i(k)$, respectively. The λ, κ and τ_a are the frequency bin, frame index, and the introduced causality delay for adaptive filters $\widehat{w}_j(k)$. Moreover, $\Gamma_{n_l n_r}$ indicates the coherence between the noise signals. The head related coherence model proposed in [5] was utilized in this work.

CR-Based Adaptive Blocking (CRB): A noise PSD estimator (Fig. 1.b) based on the cross-relation (CR) error $e(k)$ obtains the noise PSD from the CR-error PSD $\widehat{\Phi}_e$ as [2]

$$\widehat{\Phi}_n = \frac{\widehat{\Phi}_e}{|\widehat{H}_r|^2 + |\widehat{H}_l|^2 - 2\text{Real}\{\widehat{H}_l\widehat{H}_r^*\Gamma_{n_l n_r}\}}, \quad (2)$$

where $\widehat{H}_i$ indicate the adaptive-filter transfer functions.

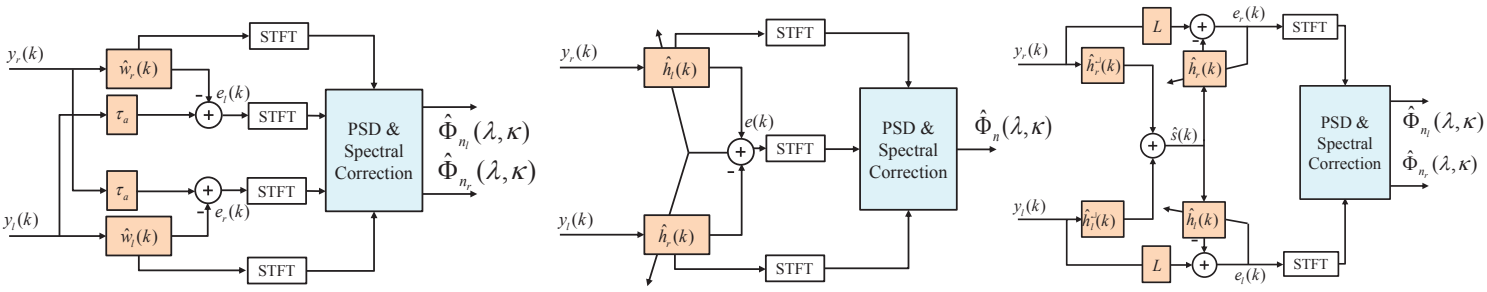

Fig. 2: Adaptive noise PSD estimation for realtime processing based on a) ITF-blocking, b) CR-blocking, and c) PCA-blocking

PCA-based Adaptive Blocking (PCAB): The noise power estimation can be obtained from a compound of error signal PSDs $\widehat{\mathbf{\Phi}}_e = [\widehat{\Phi}_{e_l}\ \widehat{\Phi}_{e_r}]^T$ of a so far unpublished PCA-based target signal cancellation (Fig. 1.c) as [3]

$$\widehat{\Phi}_n = (\mathbf{C}^T\mathbf{C})\mathbf{C}^T\widehat{\mathbf{\Phi}}_e. \qquad (3)$$

Here $\mathbf{C}=\mathbf{B}-2\text{Real}\{\widehat{H}_l^*\,\widehat{H}_r\}\Gamma_{n_l n_r}$, $\mathbf{H}'$, $\mathbf{H}'=\left[1-|\widehat{H}_l|^2 \quad 1-|\widehat{H}_r|^2\right]^T$, and $\mathbf{B}=\left[|\widehat{H}_l|^2|\widehat{H}_r|^2+(1-|\widehat{H}_l|^2)^2 \quad |\widehat{H}_l|^2|\widehat{H}_r|^2+(1-|\widehat{H}_r|^2)^2\right]^T$.

1.2. Binaural Cue-Preserving MMSE Filter

The estimated noise PSDs are then employed in a cue-preserving MMSE noise reduction filter [3, 4], such that the estimated speech signals in the frequency domain read

$$\widehat{X}_i = G_o Y_i \qquad G_o = 1 - \frac{\Phi_{n_l n_l} + \Phi_{n_r n_r}}{\Phi_{y_l y_l} + \Phi_{y_r y_r}}, \qquad (4)$$

where $\Phi_{y_i y_i}$ denote the PSDs of the microphone signals.

2. DEMONSTRATION SETUP

The binaural noise reduction demonstration setup is illustrated in Fig. 3. Here, we use the microphones embedded in the Sony MDR-NC31EM headset in order to capture the noisy signals at the left and right ears. The captured noisy signals are then fed into the Art-Dual Pre external USB sound card and transfered to Raspberry Pi for online processing.

The described noise reduction algorithm is implemented in Simulink. Through the Simulink support package, models can be easily deployed and compiled on Raspberry Pi. The ALSA Audio Capture and ALSA Audio Playback blocks are used to capture and play back the noisy and enhanced signals, respectively. The model is deployed over an IP network using the host computer and a Wi-Fi router. The host computer offers the operator the possibility to alternately provide the listener with the processed and unprocessed signal using a manual switch. In this way, we achieve an A/B comparison of sound quality, while the spatial cues of the desired speech are potentially varying. However, the demonstration can as well

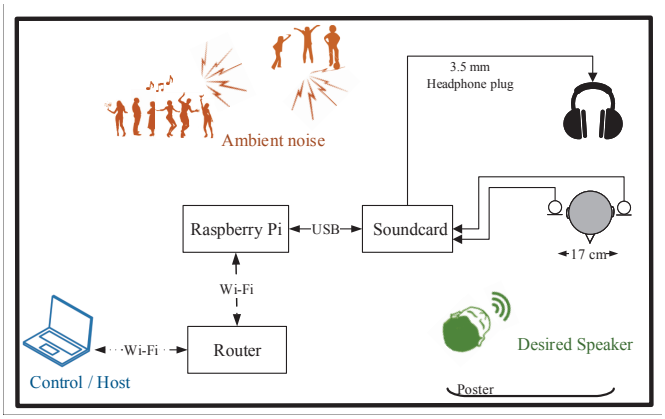

Fig. 3: Demonstration Setup

run without the host computer and the network connection after model deployment.

An important factor is the latency introduced in the system by the buffer queue length of the ALSA driver. In order to reduce the latency, the buffer size in the Raspberry Pi support package is modified to 20-30 ms from originally 500 ms.

A table with minimum 1m×2m, preferably in the middle of the show-and-tell area is needed. Moreover, the access to a nearby AC 120 V power socket is required for this demonstration. For ambient background noise, we count on the environmental noise in conference venue, e.g., show-and-tell area.

3. REFERENCES

[1] M. Azarpour, G. Enzner, and R. Martin, "Binaural noise PSD estimation for binaural speech enhancement," in *Proc. IEEE Int. Conf. on Acoustics, Speech, and Signal Process. (ICASSP)*, 2014.

[2] M. Azarpour and G. Enzner, "Fast noise PSD estimation based on blind channel identification," in *Int. Workshop on Acoustic Signal Enhancement (IWAENC)*, 2014, pp. 223–227.

[3] M. Azarpour and G. Enzner, "Binaural speech enhancement via cue-preserving MMSE filter and adaptive-blocking-based noise PSD estimation," submitted to EURASIP Journal on Advances in Signal Processing.

[4] G. Enzner, M. Azarpour, and J. Siska, "Cue-preserving MMSE filter for binaural speech enhancement," in *Int. Workshop on Acoustic Signal Enhancement (IWAENC)*, 2016.

[5] M. Jeub, M. Dörbecker, and P. Vary, "A semi-analytical model for the binaural coherence of noise fields," *IEEE Signal Process. Letters*, vol. 18, no. 3, pp. 197–200, 2011.