INTERSPEECH 2021
30 August – 3 September, 2021, Brno, Czechia

Acoustic Echo Cancellation with Cross-Domain Learning

Lukas Pfeifenberger[1], *Matthias Zoehrer*[1], *Franz Pernkopf*[2]

[1]Evolve Technologies, Austria
[2]Graz University of Technology, Austria

lukas.pfeifenberger@evolve.tech, matthias.zoehrer@evolve.tech, pernkopf@tugraz.at

Abstract

This paper proposes the Cross-Domain Echo-Controller (CDEC), submitted to the Interspeech 2021 AEC-Challenge. The algorithm consists of three building blocks: (i) a Time-Delay Compensation (TDC) module, (ii) a frequency-domain block-based Acoustic Echo Canceler (AEC), and (iii) a Time-Domain Neural-Network (TD-NN) used as a post-processor. Our system achieves an overall MOS score of 3.80, while only using 2.1 million parameters at a system latency of 32ms.

Index Terms: Acoustic Echo Cancellation, Neural Networks, Residual Echo Cancellation

1. Introduction

Acoustic echo cancellation (AEC) plays an essential part in today's VoIP speech communication and video conferencing systems. Caused by the room acoustics, acoustic echoes occur between the loudspeaker and the microphone of a headset, handset or any other audio hardware used for voice communication. Depending on the reverberation time of the room, acoustic echoes can be quite prominent, up to the point where they significantly degrade both speech intelligibility and speech quality [1]. This is especially an issue in hands-free scenarios [2]. Therefore, an efficient AEC solution is an essential building block for reliable voice communication. A typical AEC models the echo impulse response (EIR) between the loudspeaker and the microphone as linear FIR filter, and adaptively adjusts this filter using the normalized least mean square (NLMS) algorithm [3,4]. Many implementations require a voice activity detector (VAD) to stop adaption during double-talk, i.e. when both the near-end and far-end speaker are talking simultaneously [3,5]. More sophisticated implementations account for double-talk by using a state-space model [6] or Kalman filters [7]. However, the linear echo model cannot regard for non-linear distortions in the echo path, or for additive noise which is picked up by the microphone. Commercial AEC frameworks like SpeexDSP [8], WebRTC [9] or PjSIP [10] rely on traditional methods for non-linear echo and noise removal such as Wiener filters [11], Volterra kernels [12], or Hammerstein models [13].

Recently, neural networks have been proposed for non-linear residual echo and noise removal [14–19]. From the deep-learning perspective, these tasks can be seen as a speech or audio source separation problem [2,14,18–23]. Although this field of research quickly progressed in recent years [24, 25], most NN-based speaker separation algorithms are computationally demanding, not causal and do not work in real-time applications. Systems that are capable of real-time processing operate on a frame-by-frame basis. In particular, recurrent neural networks (RNN) such as gated recurrent units (GRU) [26] or long short term memory (LSTM) [27] networks are used to model the temporal correlations in human speech, while adhering to the real-time constraints of a typical AEC application [2, 19, 28]. Similar architectures [29–31] have been applied to real-time signal enhancement as contributions to the deep noise suppression challenge of Interspeech 2020 [32] and the ICASSP AEC challenge [33].

This paper presents our contribution to the Interspeech 2021 AEC-Challenge, which consists of three cascading modules: (i) A Time-Delay Compensation (TDC) module based on the Generalized Cross-Correlation with PHAse Transform (GCC-PHAT) [4], which compensates the lag between the near-end loudspeaker and microphone signals. (ii) A frequency-domain state-space block-partitioned AEC algorithm [6], which removes the linear echo component. (iii) A Time-Domain Neural-Network (TD-NN), which removes both the non-linear residual echo and additive noise. We refer to our system as the Cross-Domain Echo-Controller (CDEC), as it operates in both the frequency- and time-domain. Evaluation of our model is based on perceptual speech quality metrics using the ITU P.808 framework [33], which reports the Mean Opinion Score (MOS). Further, we report additional metrics such as the MOSnet [34] and ERLE [35]. Finally, we also report the computational complexity of our CDEC system in terms of MAC operations per frame of audio data.

2. Proposed System

2.1. Problem Formulation

In a typical AEC system there are two input signals available: (i) The far-end microphone signal $x(t)$, which is played by the local loudspeaker. (ii) The near-end microphone signal $d(t)$, which can be described as a superposition of the following components:

$$d(t) = x(t - \Delta_t) \circledast h(t) + s(t) + n(t) + v(t), \qquad (1)$$

where $h(t)$ denotes the EIR between the near-end loudspeaker and near-end microphone, $s(t)$ is the desired near-end speaker, $n(t)$ is some additive noise at the location of the near-end microphone, and $v(t)$ is the residual echo caused by non-linear distortions in the loudspeaker or the amplifier(s). The convolution of the far-end signal $x(t)$ and the EIR $h(t)$ is denoted by the $\circledast$ operator. Note that the far-end signal is delayed by an unknown time lag Δ_t, which is caused by the latency of the audio front-end of the respective communication device. Further delays may be introduced by the audio driver, which typically uses interrupt-driven block processing of the audio data. On modern sound servers, this delay can be adjusted and is controlled by the kernel. However, it may change due to high system load in multi-tasking operating systems. Figure 1 shows the involved signals in both the far-end and the near-end, and our proposed cross-domain echo controller (CDEC) for the near-end. It consists of three modules: (i) A GCC-PHAT based time-delay compensation (TDC), (ii) a frequency-domain state-space block-partitioned AEC, and (iii) a time-domain post-processing neural network (TD-NN). In the following, we will describe these three modules in detail.

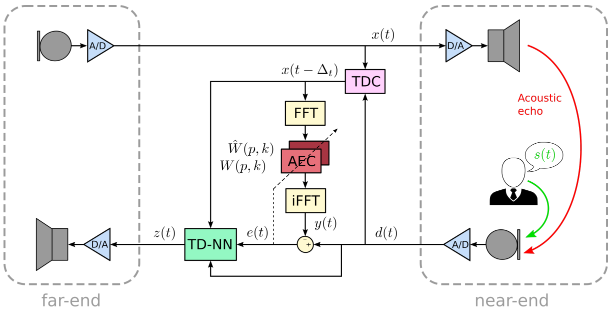

Figure 1: *Structure of the cross-domain echo controller (CDEC).*

2.2. TDC module

Generally, the AEC is be capable of modeling the time lag Δ_t between the far-end signal $x(t)$ and the near-end signal $d(t)$ as leading zeros in its filter weights. However, it is more practical to compensate this delay explicitly before the AEC in order to keep the modeled EIR short, and thus save computing resources. While this delay is potentially unknown, we adhere to assumptions typically made in real-time audio-processing frameworks [8–10]. In particular, we assume that the delay does not exceed 1s, and that it is stable for at least 10s.

We employ the GCC-PHAT algorithm [4] to compare the far-end signal $x(t)$ and the near-end signal $d(t)$ in frequency domain, i.e. we evaluate the cross-correlation $\Phi(l,k)$ as

$$\Phi(l,k) = \Phi(l,k)\alpha + (1-\alpha)X(l,k)D(l,k)^*, \tag{2}$$

where $X(l,k)$ and $D(l,k)$ denote the frequency-domain representations of the signals $x(t)$ and $d(t)$, respectively. The time frame is denoted by l, and the frequency bin by k. The smoothing constant α determines a trade-off between accuracy and reaction time to sudden changes in the time lag Δ_t. This time lag is estimated by

$$\Delta_t = \underset{t}{\text{argmax}}\, \mathcal{F}^{-1}\frac{\boldsymbol{\Phi(l)}}{|\boldsymbol{\Phi(l)}|}, \tag{3}$$

where $\mathcal{F}^{-1}$ denotes the inverse FFT, $\boldsymbol{\Phi}(l) = [\boldsymbol{\Phi}(l,1),\ldots,\boldsymbol{\Phi}(l,K)]^T$, and K is the number of frequency bins. At $f_s = 16$kHz, a maximum delay compensation of 1s equates to a required FFT size of 16384 points.

2.3. AEC module

With the time-aligned signals $x(t-\Delta_t)$ and $d(t)$, we employ the frequency-domain state-space block-partitioned AEC algorithm [6], which operates the blocks $\boldsymbol{x}'(l)$ and $\boldsymbol{d}'(l)$ of the near-end loudspeaker and microphone signals, respectively. Each block uses the latest $2T$ latest samples of the respective time-domain signal, i.e.

$$\begin{aligned}\boldsymbol{x}'(l) &= x(t+n-2T),\\ \boldsymbol{d}'(l) &= d(t+n-2T),\end{aligned} \tag{4}$$

with $n = \{0\ldots 2T-1\}$. Note that we use t as discrete time index, for the sake of a simplified notation. The blocks overlap by 50%, or T samples in time domain. The AEC divides a potentially very long echo tail into P partitions, i.e.

$$\begin{aligned}Y(l,k) &= \sum_{p=0}^{P-1} X(l-p,k)W(p,k),\\ \boldsymbol{e}'(l) &= \boldsymbol{d}'(l) - \mathcal{F}^{-1}\{\boldsymbol{Y}(l)\},\end{aligned} \tag{5}$$

where $W(p,k)$ denotes the p^{th} block of the filter weights. The time-domain block $\boldsymbol{e}'(l)$ denotes the residual signal for the l^{th} time frame. To avoid aliasing artifacts, the overlap-save method [4] is used. In particular, only the last T samples (i.e. the most recent ones) are used to reconstruct the time-domain residual signal $e(t)$, i.e.

$$e(t+n-T) = e'(l, n+T), \tag{6}$$

where $n = \{0\ldots T-1\}$. Consequently, the overall systen delay of the AEC is T samples, regardless of the number of partitions P being used. To model an echo tail of up to 0.25s, we use $P = 16$ blocks in Eq. 5. To avoid aliasing in the filter weights, the last T samples of each block of the time-domain weights are zero-padded, i.e.

$$\begin{aligned}\boldsymbol{w}(p) &= \mathcal{F}^{-1}\{\boldsymbol{W}(p)\},\\ w(p,n+T) &= 0,\\ \boldsymbol{W}(p) &\leftarrow \mathcal{F}\{\boldsymbol{w}(p)\},\end{aligned} \tag{7}$$

with $n = \{0\ldots T-1\}$. The update rule for the filter weights $W(p,k)$ can be found in [6]. To account for sudden changes in the EIR caused by spontaneous volume changes, or sudden speaker movement in the near-end, we use a second set of filter weights $\hat{W}(p,k)$ as *shadow* weights. Algorithm 1 illustrates how those weights are updated.

Algorithm 1 Shadow weights update.

```
1: if $\mathcal{E}(l) < 3dB$ then
2:     $\hat{W}(p,k) = W(p,k)$
3: end if
4: if $\hat{\mathcal{E}}(l) > \mathcal{E}(l) + 1dB$ then
5:     $W(p,k) = \hat{W}(p,k)$
6: end if
```

The shadow weights are updated based on the ERLE $\mathcal{E}(l)$, which is continuously evaluated for both the foreground weights $W(p,k)$ and the shadow weights $\hat{W}(p,k)$, i.e.

$$\begin{aligned}\mathcal{E}(l) &= 10\log_{10}\frac{\sum_k |D(l,k)|^2}{\sum_k |E(l,k)|^2},\\ \hat{\mathcal{E}}(l) &= 10\log_{10}\frac{\sum_k |D(l,k)|^2}{\sum_k |\hat{E}(l,k)|^2},\end{aligned} \tag{8}$$

where $D(l,k)$, $E(l,k)$, and $\hat{E}(l,k)$ are the FFTs of $\boldsymbol{d}'(l)$, $\boldsymbol{e}'(l)$, and $\hat{\boldsymbol{e}}'(l)$, respectively. The block $\hat{\boldsymbol{e}}'(l)$ is obtained by inserting the shadow weights $\hat{W}(p,k)$ into Eq. 5. The update rule in Algorithm 1 ensures that the weights with the highest ERLE are used for each frame. Hence, the AEC is able to quickly re-adapt to the last known good filter weights.

2.4. TD-NN module

To account for the non-linear residual echo $v(t)$ and the additive noise $n(t)$ in the system model from Eq. 1, we use a small neural network in time-domain. Similar to the AEC, it operates on blocks of T samples, with an overlap of 50%. Figure 2 illustrates the structure of the time-domain neural network (TD-NN). The upper branch derives a mask $\boldsymbol{m}(l)$ in latent space. In particular, the mask estimation branch uses a Conv1D layer with a kernel size of $F = 1600$ samples and a stride of $S = 128$ samples to transform the four signals $x(t)$, $y(t)$, $d(t)$, and $e(t)$ into a latent representation with H neurons per signal. Note that this Conv1D layer uses the past 1600 samples of the respective signals, i.e. it sees a context of the past 100ms of audio data. Each signal is individually normalized by instant layer normalization to account for individual level variations. Instant layer normalization is similar to standard layer normalization [36]. The last Feed-Forward (FF) layer in that branch uses a softplus activation function, to provide an unconstrained mask.

The lower branch in Figure 2 illustrates the application of the mask to the residual signal $e(t)$ in latent space. There, the Conv1D layer uses a kernel size of $F = 256$ samples and a stride of $S = 128$ samples to produce a latent space of $H = 200$ neurons. The mask $\boldsymbol{m}(l)$ is multiplied to the latent representation obtained by the GRU layer. Finally, a Conv1DTranspose layer is used to predict the enhanced time-domain output $z(t)$. It uses the same parameters as the Conv1D layer, i.e. $F = 256$ and $S = 128$. Signal reconstruction is achieved by the overlap-add method [4], which accounts to a total look-ahead of $F + S$ samples.

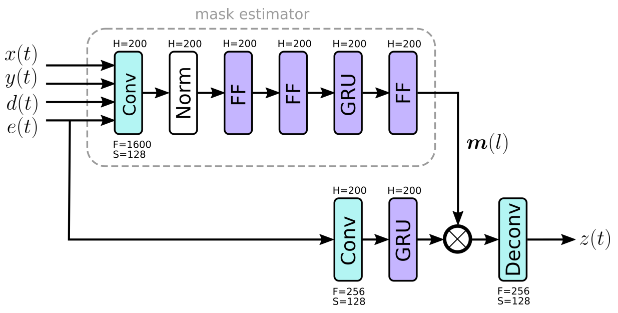

Figure 2: *Structure of the proposed post-processing time-domain neural network (TD-NN).*

3. Experiments

3.1. Dataset

The AEC challenge provides recordings from more than 2,500 audio devices and human speakers in real environments. It covers the following three scenarios: near-end single-talk (NE), far-end single-talk (FE), and double-talk (DT). For training, two datasets are provided, *real* recordings, and *synthetic* examples [33]. The synthetic dataset provides 10,000 examples representing single-talk, double-talk, near-end noise, far-end noise, and various nonlinear distortion situations. The real dataset provides a variety of over 37,000 single-talk and double-talk recordings, both with and without echo path changes, background noise and non-linear distortions. For the evaluation with the P. 808 framework, a test set of 800 utterances was provided by the challenge organizers. The test set is divided into the three scenarios, i.e. NE, FE and DT.

3.2. Data Augmentation

To have a ground truth for training, we only use the far-end single-talk files, of which there are 10,000 in the synthetic dataset, and 7,282 in the real dataset. In particular, we only use far-end $x(t)$ and near-end $d(t)$ signal pairs where the average energy of $d(t) > -40dB_{FS}$. Otherwise the signals are rejected.

We generate double-talk examples by mixing clean WSJ0 data [37] as desired near-end speech $s(t)$ into the near-end microphone signal $d(t)$. We randomly select the signal-to-interference ratios (SIR) between the echo and the desired speech signal $s(t)$ from a uniform distribution between $-6 \ldots 6dB$. To account for the great variety of different microphones in the training data, we perform random spectral shaping with a 20-band equalizer, where a uniformly distributed gain between $-12 \ldots 12dB$ is applied to each individual band.

To simulate additive noise, we use 20 hours of 20 different sound categories from YouTube [1] as noise signal $n(t)$. The SNR of the additive noise is randomly chosen from a uniform distribution between $12 \ldots 36dB$. The noise is only added to the simulated dataset.

To further increase robustness and to model various transmission effects, we introduce a single, artificial delay change in each of the far-end signals $x(t)$ of the simulated dataset. The delay change is randomly chosen from a uniform distribution between $-20 \ldots 0ms$. This causes the AEC to re-adapt during each utterance. Further, to reflect the sudden amplitude changes in the test data, we attenuate a randomly selected third of the microphone signal $d(t)$ by a uniformly distributed gain ranging from $-20 \ldots 0dB$. Lastly, to simulate clipping artifacts, we clip the microphone signal at a randomly chosen amplitude ranging from $-12 \ldots 0dB$. The real dataset already features moving speakers, additive noise, and a certain amount of non-linear distortions.

With this setup, we generate 15,000 signal pairs $x(t)$, $d(t)$ for each of the three scenarios, i.e. NE, FE and DT. We truncate each signal to a length of 10s, to be able to stack them into batches for training.

3.3. CDEC Training

During training, we first estimate the bulk delay using the GCC-PHAT from Eq. 2 once every 10s, i.e. once for each training utterance. Next, we perform the AEC from Eq. 5 through 8, which outputs the echo model $y(t)$ and the residual signal $e(t)$. Depending on the amount of non-linear distortions and additive noise, the residual is already close to the desired near-end speech $s(t)$. As illustrated in Figure 2, we use the four signals $x(t)$, $y(t)$, $d(t)$, and $e(t)$ as feature vectors, and the desired signal $s(t)$ as target vector. For the NE and DT scenarios, we use the SDR as cost function, i.e.

$$\mathcal{L}_{\text{SDR}} = 10\log_{10}\frac{\sum_t s(t)^2}{\sum_t [s(t) - z(t)]^2}, \tag{9}$$

whereas we use the ERLE as cost function for the FE scenario, i.e.

$$\mathcal{L}_{\text{ERLE}} = 10\log_{10}\frac{\sum_t d(t)^2}{\sum_t z(t)^2}. \tag{10}$$

We define the overall cost function as

$$\mathcal{L}_{\text{ERLE}} = -\mathcal{L}_{\text{SDR}} - \lambda\mathcal{L}_{\text{ERLE}}, \tag{11}$$

where we set $\lambda = 0.5$. We randomly select a batch of 40 utterances from the three scenarios NE, FE and DT with a probability of $p_{NE} = 0.25$, $p_{FE} = 0.25$, and $p_{DT} = 0.5$.

[1] https://python-pytube.readthedocs.io/en/latest/

3.4. Objective and Subjective Audio Quality Evaluation

Conventional objective metrics such as the perceptual evaluation of speech quality (PESQ) do not correlate well with subjective speech quality tests in the presence of reverberation, additive noise and non-linear distortions. Therefore, a study based on the ITU P.808 crowd-sourcing framework [33] on the Amazon Mechanical Turk platform was conducted. In total four scenarios were evaluated: single-talk near-end (P.808), single-talk far-end (P.831), double-talk echo (P.831) and double-talk other disturbances (P.831). We evaluate the distortion MOS (DMOS) and the echo MOS (EMOS) defined in ITU-T P.831 [38]. For more details on the rating process see [33]. In addition, to get a better impression on the CDEC performance, we also employ additional metrics such as the MOSnet [34] and ERLE [35].

4. Results

4.1. Objective and Subjective Quality Scores

Table 1 shows the results obtained with the evaluation script provided by the challenge organizers [33]. It can be seen that the CDEC greatly improves the EMOS scores for the DT and FE scenarios. However, the improvements for the DMOS scores in the DT and NE scenarios are less significant.

Table 1: *DMOS and EMOS scores for the blind test set.*

Score	DT DMOS	DT EMOS	FE EMOS	NE DMOS	Overall
AEC only	3.21	2.60	1.99	3.60	2.85
CDEC	3.37	3.78	4.28	3.75	3.80

Table 2 shows the ERLE for the FE scenario, and the MOSnet score for the DT and NE scenarios. Note that the MOSnet only requires the enhanced waveform to obtain a score [34]. For the FE scenario it can be seen that the CDEC system greatly improves the ERLE, compared to just using the AEC. For the NE and DT scenarios, the CDEC system reaches the same MOS as the AEC. This indicates that the CDEC preserves the speech quality with high accuracy while removing residual echoes and other interferences.

Table 2: *MOSnet and ERLE scores for the blind test set.*

Scenario	DT	FE	NE
Score	MOSnet	ERLE	MOSnet
AEC only	2.94	2.81 dB	3.08
CDEC	3.05	43.65 dB	3.09

4.2. Overall System Latency

To adhere to the challenge rules, the CDEC system operates on frames of size $T = 256$ samples. The TDC module concatenates 64 frames to compute the delay Δ_t using Eq. 3 once every 10s. Hence, the latency of the TDC module equates to 16ms at $f_s = 16$kHz. The AEC module operates on blocks of $2T$ samples, which are obtained by concatenating the two most recent frames of $x(t)$ and $d(t)$, as shown in Eq. 4. Due to the overlap-save operation, it outputs the most recent frame of the residual $e(t)$, as explained in Section 2.3. Hence, the AEC has the same latency of 16ms. The TD-NN module operates on single frames of size $F = T = 256$ samples, with a stride of $S = 128$ samples. The Deconvolution layer uses the overlap-add method, which requires two output frames to be present, in order to shift and add them together to obtain the final output signal $z(t)$. Hence, the latency of the TD-NN is $2T$ samples, equating to 32ms. Since all three modules operate on the same blocks, the total latency of the CDEC system is 32ms.

4.3. Computational Complexity

The computational complexity of the CDEC model was evaluated on a quad-core i5 2.5Ghz reference system. In particular, we measured the execution time of a single frame of the forward pass of the CDEC system, using a single-precision reference implementation in C++ with the Matrix/Vector library Eigen [2], and the FFT library FFTW [39]. The TD-NN system uses 2.1 million parameters, while the AEC uses $2P \cdot 2T = 16384$ complex-valued filter weights, including the shadow weights. One inference step takes 228 us per frame. In particular, the TDC module takes 0.16 us, the AEC 32.88 us and the DNN 195 us for processing a single frame. Note that the TDC module is executed once every 10s, hence its contribution to the execution time of a single frame is fairly small. The overall execution time of the CDEC system is 28.8 ms per 1 second of audio using a single CPU. In case of multi-threaded execution with the help of XNNPACK[3] the run-time can be decreased to 19.6 ms processing one second of audio. The model can be further downsized using sparse formats and pruning. Furthermore, the use of fixed point representation greatly reduces both memory consumption and computational complexity as shown in [40]. Table 3 shows the computational complexity of the CDEC model.

Table 3: *Computational complexity of the single-precision CDEC model, measured on a quad-core i5 2.5GHz reference system.*

Model	Threads	us/Frame	Real-time Factor
CDEC	1	228	0.0288
CDEC	4	157	0.0193

5. Conclusion

We presented our Cross Domain Echo Controller (CDEC) – a real-time AEC system developed for the Interspeech AEC Challenge 2021. The system consists of three modules, i.e. a Time-Delay Compensation (TDC) block, a frequency-domain block-based Acoustic Echo Canceler (AEC) and a Time-Domain Neural-Network (TD-NN). The CDEC was evaluated on both single- and double talk echo scenarios, using the ITU P.808 crowd-sourcing framework. In particular, it reports an average MOS score of 3.80, using a model with 2.1M parameters. The overall system has an overall latency of 32ms, with an real-time factor of 0.0288 on a 2.5 Ghz quad-core i5 system.

6. References

[1] H. Kuttruff, *Room Acoustics*, 5th ed. London–New York: Spoon Press, 2009.

[2] L. Pfeifenberger and F. Pernkopf, "Nonlinear Residual Echo Suppression Using a Recurrent Neural Network," in *Proc. Interspeech 2020*, 2020, pp. 3950–3954.

[3] S. Haykin, *Adaptive Filter Theory*, 4th ed. New Jersey: Prentice Hall, 2002.

[4] J. Benesty, M. M. Sondhi, and Y. Huang, *Springer Handbook of Speech Processing*. Berlin–Heidelberg–New York: Springer, 2008.

[2]http://eigen.tuxfamily.org

[3]https://github.com/google/XNNPACK

[5] G. Enzner, H. Buchner, A. Favrot, and F. Kuech, "Chapter 30 - acoustic echo control," in *Academic Press Library in Signal Processing: Volume 4*, ser. Academic Press Library in Signal Processing, J. Trussell, A. Srivastava, A. K. Roy-Chowdhury, A. Srivastava, P. A. Naylor, R. Chellappa, and S. Theodoridis, Eds. Elsevier, 2014, vol. 4, pp. 807–877.

[6] F. Kuech, E. Mabande, and G. Enzner, "State-space architecture of the partitioned-block-based acoustic echo controller," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2014, pp. 1295–1299.

[7] C. Wu, X. Wang, Y. Guo, Q. Fu, and Y. Yan, "Robust uncertainty control of the simplified kalman filter for acoustic echo cancelation," *Circuits, Systems, and Signal Processing*, vol. 35, no. 12, pp. 4584–4595, 2016.

[8] "Speex-dsp," Website, visited on February 19th 2020. [Online]. Available: https://github.com/xiongyihui/speexdsp-python

[9] "Webrtc toolkit," Website, 2011, visited on March 14th 2021. [Online]. Available: https://webrtc.org/

[10] "PjMedia – acoustic echo cancellation api." Website, 2008, visited on March 13th 2021. [Online]. Available: https://www.pjsip.org/pjmedia/docs/html/group__PJMEDIA__Echo__Cancel.htm

[11] H. Huang, C. Hofmann, W. Kellermann, J. Chen, and J. Benesty, "A multiframe parametric wiener filter for acoustic echo suppression," in *IEEE International Workshop on Acoustic Signal Enhancement (IWAENC)*, 2016, pp. 1–5.

[12] F. Kuech and W. Kellermann, "A novel multidelay adaptive algorithm for volterra filters in diagonal coordinate representation [nonlinear acoustic echo cancellation example]," in *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, vol. 2, 2004, pp. ii–869.

[13] S. Malik and G. Enzner, "Fourier expansion of hammerstein models for nonlinear acoustic system identification," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2011, pp. 85–88.

[14] C. M. Lee, J. W. Shin, and N. S. Kim, "Dnn-based residual echo suppression," in *Interspeech*, 2015.

[15] T. V. Huynh, "A new method for a nonlinear acoustic echo cancellation system," in *International Research Journal of Engineering and Technology*, vol. 4, 2017.

[16] H. Zhang and D. Wang, "Deep learning for acoustic echo cancellation in noisy and double-talk scenarios," in *Interspeech*, 2018, pp. 3239–3243.

[17] G. Carbajal, R. Serizel, E. Vincent, and E. Humbert, "Multiple-input neural network-based residual echo suppression," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2018, pp. 231–235.

[18] G. Carbajal, R. Serizel, E. Vincent, and E. Humbert, "Joint dnn-based multichannel reduction of acoustic echo, reverberation and noise," *CoRR*, vol. abs/1911.08934, 2019.

[19] Q. Lei, H. Chen, J. Hou, L. Chen, and L. Dai, "Deep neural network based regression approach for acoustic echo cancellation," in *International Conference on Multimedia Systems and Signal Processing (ICMSSP)*. New York, NY, USA: Association for Computing Machinery, 2019, p. 94–98.

[20] L. Pfeifenberger, M. Zöhrer, and F. Pernkopf, "Eigenvector-based speech mask estimation for multi-channel speech enhancement," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 27, no. 12, pp. 2162–2172, 2019.

[21] L. Pfeifenberger, M. Zöhrer, and F. Pernkopf, "Dnn-based speech mask estimation for eigenvector beamforming," in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2017*, Mar. 2017, pp. 66–70.

[22] H. Zhang, K. Tan, and D. Wang, "Deep learning for joint acoustic echo and noise cancellation with nonlinear distortions," *Interspeech*, pp. 4255–4259, 2019.

[23] A. Fazel, M. El-Khamy, and J. Lee, "CAD-AEC: context-aware deep acoustic echo cancellation," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 6919–6923.

[24] J. R. Hershey, Z. Chen, J. Le Roux, and S. Watanabe, "Deep clustering: Discriminative embeddings for segmentation and separation," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2016, pp. 31–35.

[25] M. Kolbak, D. Yu, Z. Tan, and J. Jensen, "Multitalker speech separation with utterance-level permutation invariant training of deep recurrent neural networks," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 25, no. 10, pp. 1901–1913, 2017.

[26] J. Chung, Ç. Gülçehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," *CoRR*, vol. abs/1412.3555, 2014.

[27] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, p. 1735–1780, 1997.

[28] L. Ma, H. Huang, P. Zhao, and T. Su, "Acoustic echo cancellation by combining adaptive digital filter and recurrent neural network," *CoRR*, vol. 2005.09237, 2020.

[29] J.-M. Valin, U. Isik, N. Phansalkar, R. Giri, K. Helwani, and A. Krishnaswamy, "A perceptually-motivated approach for low-complexity, real-time enhancement of fullband speech," *CoRR*, vol. 2008.04259, 2020.

[30] Y. Hu, Y. Liu, S. Lv, M. Xing, S. Zhang, Y. Fu, J. Wu, B. Zhang, and L. Xie, "Dccrn: Deep complex convolution recurrent network for phase-aware speech enhancement," *CoRR*, vol. 2008.00264, 2020.

[31] N. L. Westhausen and B. T. Meyer, "Dual-signal transformation lstm network for real-time noise suppression," *CoRR*, vol. 2005.07551, 2020.

[32] C. K. A. Reddy, V. Gopal, R. Cutler, E. Beyrami, R. Cheng, H. Dubey, S. Matusevych, R. Aichner, A. Aazami, S. Braun, P. Rana, S. Srinivasan, and J. Gehrke, "The interspeech 2020 deep noise suppression challenge: Datasets, subjective testing framework, and challenge results," 2020.

[33] K. Sridhar, R. Cutler, A. Saabas, T. Parnamaa, M. Loide, H. Gamper, S. Braun, R. Aichner, and S. Srinivasan, "Icassp 2021 acoustic echo cancellation challenge: Datasets, testing framework, and results," *CoRR*, vol. 2009.04972, 2020.

[34] C.-C. Lo, S.-W. Fu, W.-C. Huang, X. Wang, J. Yamagishi, Y. Tsao, and H.-M. Wang, "Mosnet: Deep learning based objective assessment for voice conversion," *CoRR*, vol. 1904.08352, 2019.

[35] I. T. Union, "Itu-t g.168: Digital network echo cancellers." 2012. [Online]. Available: https://www.itu.int/rec/T-REC-G.168/en

[36] J. L. Ba, J. R. Kiros, and G. E. Hinton, "Layer normalization," *CoRR*, vol. 1607.06450, 2016.

[37] D. B. Paul and J. M. Baker, "The design for the Wall Street Journal-based CSR corpus," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Banff, 1992, pp. 899–902.

[38] I. T. Union, "Subjective performance evaluation of network echo cancellers," 1998. [Online]. Available: https://www.itu.int/rec/T-REC-P.831/en

[39] M. Frigo and S. G. Johnson, "The design and implementation of FFTW3," *Proceedings of the IEEE*, vol. 93, no. 2, pp. 216–231, 2005, special issue on "Program Generation, Optimization, and Platform Adaptation".

[40] J.-M. Valin, S. Tenneti, K. Helwani, U. Isik, and A. Krishnaswamy, "Low-complexity, real-time joint neural echo control and speech enhancement based on percepnet," *CoRR*, vol. 2102.05245, 2021.