FAST-RIR: FAST NEURAL DIFFUSE ROOM IMPULSE RESPONSE GENERATOR

Anton Ratnarajah[1]*, Shi-Xiong Zhang*[2]*, Meng Yu*[2]*, Zhenyu Tang*[1]*, Dinesh Manocha*[1]*, Dong Yu*[2]

[1] University of Maryland, College Park, MD, USA
[2] Tencent AI Lab, Bellevue, WA, USA
[1]{jeran, zhy, dmanocha}@umd.edu, [2]{auszhang, raymondmyu, dyu}@tencent.com

ABSTRACT

We present a neural-network-based fast diffuse room impulse response generator (FAST-RIR) for generating room impulse responses (RIRs) for a given acoustic environment. Our FAST-RIR takes rectangular room dimensions, listener and speaker positions, and reverberation time (T_{60}) as inputs and generates specular and diffuse reflections for a given acoustic environment. Our FAST-RIR is capable of generating RIRs for a given input T_{60} with an average error of 0.02s. We evaluate our generated RIRs in automatic speech recognition (ASR) applications using Google Speech API, Microsoft Speech API, and Kaldi tools. We show that our proposed FAST-RIR with batch size 1 is 400 times faster than a state-of-the-art diffuse acoustic simulator (DAS) on a CPU and gives similar performance to DAS in ASR experiments. Our FAST-RIR is 12 times faster than an existing GPU-based RIR generator (gpuRIR). We show that our FAST-RIR outperforms gpuRIR by 2.5% in an AMI far-field ASR benchmark.

Index Terms— acoustic environment, speech simulation

1. INTRODUCTION

Room impulse response (RIR) generators are used to simulate large-scale far-field speech training data [1–3]. A synthetic far-field speech training dataset is created by convolving clean speech with RIRs generated for different acoustic environments and adding background noise [2,4]. The acoustic environment can be described using room geometry, speaker and listener positions, and room acoustic materials.

In recent years, an increasing number of RIR generators have been introduced to generate a realistic RIR for a given acoustic environment [5–8]. Accurate RIR generators can generate RIRs with various acoustic effects (e.g., diffraction, scattering, early reflections, late reverberation) [9]. A limitation of accurate RIR generators is that they are computationally expensive, and the time taken to generate RIRs depends on the geometric complexity of the acoustic environment. Also, many ray-based RIR simulators use the empirical Sabine formula [10] to compute the acoustic absorption coefficients from the desired reverberation time. Reverberation time (T_{60}) is the time required for the sound energy to decay by 60 decibels [11].

With advancements in deep neural-network-based far-field speech processing, the demand for on-the-fly simulation of far-field speech training datasets with hundreds of thousands of room configurations similar to the testing environment is increasing [12–15]. The CPU-based offline simulation of far-field speech with balanced T_{60} distribution requires a lot of computation time and disk space [5,7], thus it is not scalable for production-level ASR training. One strategy to improve the speed of RIR generation is parallelizing most of the stages in the existing RIR generators and making the algorithm compatible for running on GPUs [13, 16].

Main Contributions: We propose a neural-network-based fast diffuse room impulse response generator (FAST-RIR) that can be directly controlled using rectangular room dimension, listener and speaker positions, and T_{60}. T_{60} implicitly reflects the characteristics of the room materials such as the floor, ceiling, walls, furniture etc. Our FAST-RIR takes a constant amount of time to generate an RIR for any given acoustic environment, and yields accurate T_{60}.

Our FAST-RIR architecture is trained to generate both specular and diffuse reflections for a given acoustic environment. Diffuse reflection is widely observed in real-world environments and it is important to accurately model RIR. We show that our FAST-RIR can generate RIRs 400 times faster than the state-of-the-art diffuse acoustic simulator (DAS) [7] on a single CPU and 12 times faster than gpuRIR [13] on a single GPU. The RIRs generated using our FAST-RIR perform similarly to the RIRs generated using the DAS and outperform gpuRIR by up to 2.5% in far-field automatic speech recognition (ASR) experiments. Our FAST-RIR can generate RIRs for a given input T_{60} with an average error of 0.02s.

2. RELATED WORKS

The RIR generators developed over the decades can be divided into three groups: wave-based, ray-based, and neural-network-based techniques. Wave-based techniques are designed to give the most accurate results by solving wave equations [17, 18]. However, the wave-based techniques are only feasible for generating RIRs for less complicated scenes at low frequencies. Ray-based techniques [19] are less accurate

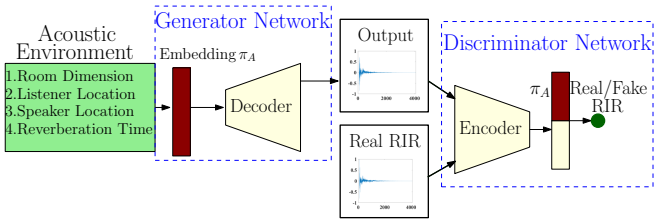

Fig. 1. The architecture of our FAST-RIR. Our Generator network takes acoustic environment details as input and generates corresponding RIR as output. Our Discriminator network discriminates between the generated RIR and the ground truth RIR for the given acoustic environment during training.

than the wave-based approach because the wave nature of the sound is neglected. The image method [5] and diffuse acoustic simulators [7] are commonly used ray-based methods in speech-related tasks. The image method only models specular reflections while diffuse acoustic simulators accurately model both diffuse and specular reflections.

Recently, neural-network-based RIR generators [20–22] have been introduced to generate RIRs for a given acoustic environment. IR-GAN [20] is a GAN-based RIR generator that is trained on a real-world RIR dataset to generate realistic RIRs. However, IR-GAN does not take conventional environmental parameters as input by design, making it less configurable than traditional RIR generators.

3. OUR APPROACH

To generate RIRs for a given acoustic environment, we propose a one-dimensional conditional generator network. Our generator network takes room geometry, listener and speaker positions, and T_{60} as inputs, which are the common input used by all traditional RIR generators, and generates RIRs as raw-waveform audio. Our FAST-RIR generates RIRs of length 4096 at 16 kHz frequency.

3.1. Modified Conditional GAN

We propose a modified conditional GAN architecture to precisely generate an RIR for a given condition. GAN [23] consists of a generator (G) and a discriminator (D) networks that are alternatingly trained to compete. The network G is trained to learn a mapping from noise vector samples (z) from distribution p_z to the data distribution p_{data}. The network G is optimized to produce samples that are difficult for the D to distinguish from real samples (x) taken from true data distribution, while D is optimized to differentiate samples generated from G and real samples. The networks G and D are trained to optimize the following two-player min-max game with value function $V(G, D)$.

$$\min_G \max_D V(G,D) = \mathbb{E}_{x\sim p_{data}}[\log D(x)] + \mathbb{E}_{z\sim p_z}[\log(1 - D(G(z)))]. \quad (1)$$

Conditional GAN (CGAN) [24,25] is an extended version of GAN where both the generator and discriminator networks are conditioned on additional information y. The generator network in CGAN is conditioned on the random noise z and y. The vector z is used to generate multiple different samples satisfying the given condition y. In our work, we train our FAST-RIR to generate a single sample precisely for a given condition. Our FAST-RIR is a modified CGAN architecture where the generator network is only conditioned on y.

3.2. FAST-RIR

We combine rectangular room dimension, listener location, and source location represented using 3D Cartesian coordinates (x, y, z) and T_{60} as a ten-dimensional vector embedding π_A. We normalize the vector embedding within the range -1.2 to 1.2 using the largest room dimension in the training dataset.

For each π_A, we generate RIR using DAS (R_D) and use it as ground truth to train our network. Our objective function for the generator network (G_N) consists of modified CGAN error, mean square error and T_{60} error. The discriminator network is trained using the modified CGAN objective function.

3.2.1. *Generator Modified CGAN Error*

The G_N is trained with the following modified CGAN error to generate RIRs that are difficult for the discriminator D_N to differentiate from RIRs generated from DAS.

$$\mathcal{L}_{CGAN} = \mathbb{E}_{\pi_A\sim p_{data}}[\log(1 - D_N(G_N(\pi_A)))]. \quad (2)$$

3.2.2. *Mean Square Error (MSE)*

We compare each sample (s) of the RIR generated using our FAST-RIR (R_N) with RIR generated using DAS (R_D) for each π_A to calculate the following MSE.

$$\mathcal{L}_{MSE} = \mathbb{E}_{\pi_A\sim p_{data}}[\mathbb{E}[(R_N(\pi_A, s) - R_D(\pi_A, s))^2]]. \quad (3)$$

3.2.3. *T_{60} Error*

We generate RIRs using our FAST-RIR and calculate their T_{60} using a method based on ISO 3382-1:2009. We compare the T_{60} of each generated RIRs with the T_{60} given as input to the network in the embedding π_A as follows:

$$\mathcal{L}_{T_{60}} = \mathbb{E}_{\pi_A\sim p_{data}}[|T_{60}(G_N(\pi_A)) - T_{60}(\pi_A)|]. \quad (4)$$

3.2.4. *Full Objective*

We train the G_N and D_N alternatingly to minimize the generator objective function $\mathcal{L}_{G_N}$ (Equation 5) and maximize the discriminator objective function $\mathcal{L}_{D_N}$ (Equation 6). We control the relative importance of the MSE ($\mathcal{L}_{MSE}$) and T_{60} error ($\mathcal{L}_{T_{60}}$) using the weights λ_{MSE} and $\lambda_{T_{60}}$, respectively.

$$\mathcal{L}_{G_N} = \mathcal{L}_{CGAN} + \lambda_{MSE}\mathcal{L}_{MSE} + \lambda_{T_{60}}\mathcal{L}_{T_{60}}. \quad (5)$$

$$\mathcal{L}_{D_N} = \mathbb{E}_{(R_D,\pi_A)\sim p_{data}}[\log(D_N(R_D(\pi_A)))] + \mathbb{E}_{\pi_A\sim p_{data}}[\log(1 - D_N(G_N(\pi_A)))]. \quad (6)$$

3.2.5. Implementation

Network Architecture: We adapt the generator network (G_N) and the discriminator network (D_N) proposed in Stage-I of StackGAN architecture [26] and modify the networks. StackGAN takes a text description and a noise vector as input and generates a photo-realistic two-dimensional (2D) image as output. Our FAST-RIR takes acoustic environment details as input and generates an RIR as a one-dimensional (1D) raw-waveform audio output. We flatten the 2D convolutions into 1D to process 1D RIR in both G_N and D_N.

Unlike photo-realistic images, raw-waveform audio exhibits periodicity. Donahue et al. [27] suggest that filters with larger receptive fields are needed to process low frequencies (large wavelength signals) in the audio. We improve the receptive field of the original G_N and the encoder in D_N by increasing the kernel size (i.e., 3×3 2D convolution becomes length 41 1D convolution) and strides (i.e., stride 2×2 becomes stride 4×1). We also replace the upsampling layer and the following convolutional layer with a transposed convolutional layer.

Dataset: The sizes of the existing real-world RIR datasets [4, 28, 29] are insufficient to train our FAST-RIR. Therefore, we generate 75,000 medium-sized room impulse responses using a DAS [7] to create a training dataset. We choose 15 evenly spaced room lengths within the range 8m to 11m, 10 evenly spaced room widths between 6m and 8m, and 5 evenly spaced room heights between 2.5m and 3.5m to generate RIRs. We position the speaker and the listener at random positions within the room and generate 100 different RIRs for each combination of room dimensions (15×10×5). The T_{60} values of our training dataset are between 0.2s and 0.7s.

Training: We iteratively train G_N and D_N using RMSprop optimizer with batch size 128 and learning rate 8x10^{-5}. For every 40 epochs, we decay the learning rate by 0.7.

4. EXPERIMENT AND RESULTS

4.1. Baselines

We randomly select 30,000 different acoustic environments within the range of the training dataset (Section 3.2.5). We generate RIRs corresponding to the selected acoustic environments using image method [5], gpuRIR [13], DAS [7] and FAST-RIR to evaluate the performance of our proposed FAST-RIR. IR-GAN [20] does not have the capability to precisely generate RIRs for a given speaker and listener positions; therefore, we did not use IR-GAN in our experiments.

Table 1. The runtime for generating 30,000 RIRs using image method, gpuRIR, DAS, and our FAST-RIR. Our FAST-RIR significantly outperforms all other methods in runtime.

RIR Generator	Hardware	Total Time	Avg Time
DAS [7]	CPU	9.01x10^5s	30.05s
Image Method [5]	CPU	4.49x10^3s	0.15s
FAST-RIR(Batch Size 1)	**CPU**	**2.15x10^3s**	**0.07s**
gpuRIR [13]	GPU	16.63s	5.5x10^{-4}s
FAST-RIR(Batch Size 1)	GPU	34.12s	1.1x10^{-3}s
FAST-RIR(Batch Size 64)	**GPU**	**1.33s**	**4.4x10^{-5}s**
FAST-RIR(Batch Size 128)	GPU	1.77s	5.9x10^{-5}s

Table 2. T_{60} error of our FAST-RIR for 30,000 testing acoustic environments. We report the T_{60} error for RIRs cropped at T_{60} and full RIRs. We only crop RIRs with T_{60} below 0.25s.

T_{60} Range	Crop RIR at T_{60}	T_{60} Error
0.2s - 0.25s	No	0.068s
0.2s - 0.25s	**Yes**	**0.033s**
0.25s - 0.7s	-	0.021s
0.2s - 0.7s	No	0.029s
0.2s - 0.7s	**Yes**	**0.023s**

4.2. Runtime

We evaluate the runtime for generating 30,000 RIRs using image method, gpuRIR, DAS and FAST-RIR on an Intel(R) Xenon(R) CPU E52699 v4 @ 2.20 GHz and a GeForce RTX 2080 Ti GPU (Table 1). The gpuRIR is optimized to run on a GPU; therefore, we generate RIR using gpuRIR only on a GPU. For a fair comparison with CPU implementations of image-method and DAS, we also generate RIRs using our FAST-RIR with batch size 1 on a CPU.

From Table 1, we can see that our proposed FAST-RIR with batch size 1 is 400 times faster than DAS [7] on a CPU. Our FAST-RIR is optimized to run on a GPU. We compare the performance of our FAST-RIR with an existing GPU-based RIR generator gpuRIR [13]. We can see that gpuRIR performs better than our FAST-RIR with batch size 1, which is not the real use case of our generator. To our best knowledge, the gpuRIR does not leverage the batch parallelization while this was supported in our FAST-RIR. We can see that our proposed FAST-RIR with batch size 64 is 12 times faster than gpuRIR.

4.3. T_{60} Error

Table 2 shows the T_{60} error of the generated RIRs calculated using Equation 4. We can see that the testing T_{60} error of our FAST-RIR is high for input T_{60} below 0.25s (0.068s) when compared to the input T_{60} greater than 0.25s (0.021s).

Our FAST-RIR is trained to generate RIRs with durations slightly above 0.25s. For the input T_{60} below 0.25s, the generated RIR has a noisy output between T_{60} and 0.25s. We

notice that cropping the generated RIRs at T_{60} improves the overall T_{60} error from 0.029s to 0.023s.

4.4. Simulated Speech Comparison

We simulate reverberant speech $x_r[t]$ by convolving clean speech $x_c[t]$ from the LibriSpeech test-clean dataset [30] with different RIRs $r[t]$ (Equation 7).

$$x_r[t] = x_c[t] \circledast r[t]. \quad (7)$$

We decode the simulated reverberant speech using Google Speech API[1] and Microsoft Speech API[2]. Table 3 shows the Word Error Rate (WER) of the decoded speech. No text normalization was applied in both cases, as only the relative WER differences between different RIR generators are concerned. For Google Speech API, we report WER for the clean and reverberant LibriSpeech test sets that are successfully decoded. The results of each speech API show that compared with the reverberant speech simulated using traditional RIR generators, reverberant speech simulated using our FAST-RIR is closer to the reverberant speech simulated using DAS [7]. We provide reverberant speech audio examples, spectrograms and the source code for reproducibility at github[3].

4.5. Far-field Automatic Speech Recognition

We want to ensure that our FAST-RIR generates RIRs that are better than or as good as existing RIR generators for ASR. We use the AMI corpus [31] for our far-field ASR experiments. AMI contains close-talk speech data recorded using Individual Headset Microphones (IHM) and distant speech data recorded using Single Distant Microphones (SDM).

We use a modified Kaldi recipe [4] to evaluate our FAST-RIR. The modified Kaldi recipe takes IHM data as the training set and tests the model using SDM data. The IHM data can be considered clean speech because the echo effects in IHM data are negligible when compared to SDM data. We augment far-field speech data by reverberating the IHM data with different RIR sets using Equation 7. The 30,000 RIRs generated using the image method, gpuRIR, DAS, and FAST-RIR are used in our experiment.

The IHM data consists of 687 long recordings. Instead of reverberating a speech recording using a single RIR, we do segment-level speech reverberation, as proposed in [4]. We split each recording at the beginning of at least continuous 3 seconds of silence. We split at the beginning to avoid inter-segment reverberated speech overlapping. We can split IHM data into 17749 segments. We reverberate each segment using a randomly selected RIR from an RIR dataset (either image method, gpuRIR, DAS, DAS-cropped or our FAST-RIR).

[1] https://cloud.google.com/speech-to-text/

[2] https://azure.microsoft.com/en-us/services/cognitive-services/speech-services/

[3] https://anton-jeran.github.io/FRIR/

[4] https://github.com/RoyJames/kaldi-reverb/

Table 3. Automatic speech recognition (ASR) results were obtained using Google Speech API and Microsoft Speech API. We simulate a reverberant speech testing dataset by convolving clean speech from the LibriSpeech dataset with different RIR datasets. We compare the reverberant speech simulated using the image method, gpuRIR and our FAST-RIR with the reverberant speech simulated using DAS. We show that the relative WER change from our method is the smallest.

Testing Dataset	**Word Error Rate** [%]	
Clean Speech ⊛ RIR	**Google API**	**Microsoft API**
Libri ⊛ DAS (baseline) [7]	6.56	2.63
Libri ⊛ gpuRIR [13]	9.39 (+43%)	3.78 (+44%)
Libri ⊛ Image Method [5]	9.03 (+38%)	3.86 (+47%)
Libri ⊛ FAST-RIR (ours)	**7.14 (+9%)**	**2.76 (+5%)**

Table 4. Far-field ASR results were obtained for far-field speech data recorded by single distance microphones (SDM) in the AMI corpus. The best results are shown in **bold**.

Training Dataset	**Word Error Rate** [%]	
Clean Speech ⊛ RIR	**dev**	**eval**
IHM ⊛ None	55.0	64.2
IHM ⊛ Image Method [5]	51.7	56.1
IHM ⊛ gpuRIR [13]	52.2	55.5
IHM ⊛ DAS [7]	47.9	**52.5**
IHM ⊛ DAS-cropped [7]	48.3	**52.6**
IHM ⊛ FAST-RIR (ours)	47.8	**53.0**

Table 4 presents far-field ASR development and test WER for far-field SDM data. We can see that our FAST-RIR outperforms gpuRIR [13] by up to 2.5% absolute WER. The DAS [7] with full duration and the DAS cropped to have the same duration as our FAST-RIR (DAS-cropped) performs similarly in the far-field ASR experiment. We see that the performance of DAS and FAST-RIR has no significant difference.

5. DISCUSSION AND FUTURE WORK

We propose a novel FAST-RIR architecture to generate a large RIR dataset on the fly. We show that our FAST-RIR performs similarly in ASR experiments when compared to the RIR generator (DAS [7]), which is used to generate a training dataset to train our FAST-RIR. Our FAST-RIR can be easily trained with RIR generated using any state-of-the-art accurate RIR generator to improve its performance in ASR experiments while keeping the speed of RIR generation the same.

Although we trained our FAST-RIR for limited room dimensions ranging from (8m,6m,2.5m) to (11m,8m,3.5m) using 75,000 RIRs, we believe that our FAST-RIR will give a similar performance when we train FAST-RIR for a larger room dimension range with a huge amount of RIRs. We would like to evaluate the performance of our FAST-RIR in the multi-channel ASR [32] and speech separation [33] tasks.

6. REFERENCES

[1] Chanwoo Kim, Ananya Misra, Kean Chin, Thad Hughes, Arun Narayanan, Tara N. Sainath, and Michiel Bacchiani, "Generation of Large-Scale Simulated Utterances in Virtual Rooms to Train Deep-Neural Networks for Far-Field Speech Recognition in Google Home," in *Proc. Interspeech 2017*, 2017, pp. 379–383.

[2] Tom Ko, Vijayaditya Peddinti, Daniel Povey, Michael L. Seltzer, and Sanjeev Khudanpur, "A study on data augmentation of reverberant speech for robust speech recognition," in *ICASSP*. 2017, pp. 5220–5224, IEEE.

[3] Tara N. Sainath et al., "Multichannel signal processing with deep neural networks for automatic speech recognition," *IEEE ACM Trans. Audio Speech Lang. Process.*, vol. 25, no. 5, pp. 965–979, 2017.

[4] Igor Szöke, Miroslav Skácel, Ladislav Mosner, Jakub Paliesek, and Jan Honza Cernocký, "Building and evaluation of a real room impulse response dataset," *IEEE J. Sel. Top. Signal Process.*, vol. 13, no. 4, pp. 863–876, 2019.

[5] Jont B. Allen and David A. Berkley, "Image method for efficiently simulating small-room acoustics," *Acoustical Society of America Journal*, vol. 65, no. 4, pp. 943–950, Apr. 1979.

[6] Carl Schissler and Dinesh Manocha, "Interactive sound propagation and rendering for large multi-source scenes," *ACM Trans. Graph.*, vol. 36, no. 1, Sept. 2016.

[7] Zhenyu Tang, Lianwu Chen, Bo Wu, Dong Yu, and Dinesh Manocha, "Improving reverberant speech training using diffuse acoustic simulation," in *ICASSP*. 2020, pp. 6969–6973, IEEE.

[8] Piotr Masztalski, Mateusz Matuszewski, Karol Piaskowski, and Michal Romaniuk, "Storir: Stochastic room impulse response generation for audio data augmentation," in *INTERSPEECH*. 2020, pp. 2857–2861, ISCA.

[9] Shiguang Liu and Dinesh Manocha, "Sound synthesis, propagation, and rendering: a survey," *arXiv preprint arXiv:2011.05538*, 2020.

[10] Wallace Clement Sabine and M David Egan, "Collected papers on acoustics," 1994.

[11] Heinrich Kuttruff, *Room acoustics*, Spon Press, 2009.

[12] Martin Karafiát, Karel Veselý, Katerina Zmolíková, Marc Delcroix, Shinji Watanabe, Lukás Burget, Jan Honza Cernocký, and Igor Szöke, "Training data augmentation and data selection," in *New Era for Robust Speech Recognition, Exploiting Deep Learning*, pp. 245–260. Springer, 2017.

[13] David Diaz-Guerra, Antonio Miguel, and Jose R Beltran, "gpurir: A python library for room impulse response simulation with gpu acceleration," *Multimedia Tools and Applications*, vol. 80, no. 4, pp. 5653–5671, 2021.

[14] Zhenyu Tang and Dinesh Manocha, "Scene-aware far-field automatic speech recognition," 2021.

[15] Martin Karafiát, František Grézl, Lukáš Burget, Igor Szöke, and Jan Černocký, "Three ways to adapt a CTS recognizer to unseen reverberated speech in BUT system for the ASpIRE challenge," in *Proc. Interspeech 2015*, 2015, pp. 2454–2458.

[16] Zhong-Hua Fu and Jian-Wei Li, "Gpu-based image method for room impulse response calculation," *Multim. Tools Appl.*, vol. 75, no. 9, pp. 5205–5221, 2016.

[17] D. Botteldooren, "Acoustical finite-difference time-domain simulation in a quasi-cartesian grid," *The Journal of the Acoustical Society of America*, vol. 95, no. 5, pp. 2313–2319, 1994.

[18] D. Botteldooren, "Finite-difference time-domain simulation of low-frequency room acoustic problems," *The Journal of the Acoustical Society of America*, vol. 98, no. 6, pp. 3302–3308, 1995.

[19] Lauri Savioja and U. Peter Svensson, "Overview of geometrical room acoustic modeling techniques," *The Journal of the Acoustical Society of America*, vol. 138, no. 2, pp. 708–730, 2015.

[20] Anton Ratnarajah, Zhenyu Tang, and Dinesh Manocha, "IR-GAN: Room Impulse Response Generator for Far-Field Speech Recognition," in *Proc. Interspeech 2021*, 2021, pp. 286–290.

[21] Anton Ratnarajah, Zhenyu Tang, and Dinesh Manocha, "Ts-rir: Translated synthetic room impulse responses for speech augmentation," *arXiv preprint arXiv:2103.16804*, 2021.

[22] Nikhil Singh, Jeff Mentch, Jerry Ng, Matthew Beveridge, and Iddo Drori, "Image2reverb: Cross-model reverb impulse response synthesis," in *ICCV*, October 2021.

[23] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron C. Courville, and Yoshua Bengio, "Generative adversarial nets," in *NIPS*, 2014, pp. 2672–2680.

[24] Mehdi Mirza and Simon Osindero, "Conditional generative adversarial nets," *arXiv preprint arXiv:1411.1784*, 2014.

[25] Jon Gauthier, "Conditional generative adversarial networks for convolutional face generation," in *Tech Report*, 2015.

[26] Han Zhang, Tao Xu, Hongsheng Li, Shaoting Zhang, Xiaogang Wang, Xiaolei Huang, and Dimitris Metaxas, "Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks," in *ICCV*, 2017.

[27] Jesse H. Engel, Kumar Krishna Agrawal, Shuo Chen, Ishaan Gulrajani, Chris Donahue, and Adam Roberts, "Gansynth: Adversarial neural audio synthesis," in *ICLR (Poster)*. 2019, OpenReview.net.

[28] Keisuke Kinoshita et al., "The REVERB challenge: A benchmark task for reverberation-robust ASR techniques," in *New Era for Robust Speech Recognition, Exploiting Deep Learning*, pp. 345–354. Springer, 2017.

[29] James Eaton, Nikolay D. Gaubitch, Alastair H. Moore, and Patrick A. Naylor, "The ACE challenge - corpus description and performance evaluation," in *WASPAA*. 2015, pp. 1–5, IEEE.

[30] Vassil Panayotov, Guoguo Chen, Daniel Povey, and Sanjeev Khudanpur, "Librispeech: An ASR corpus based on public domain audio books," in *ICASSP*. 2015, pp. 5206–5210, IEEE.

[31] J Carletta et al., "The ami meeting corpus: A pre-announcement," in *Proceedings of the Second International Conference on Machine Learning for Multimodal Interaction*. 2005, MLMI'05, p. 28–39, Springer-Verlag.

[32] A.S. Subramanian et al., "Directional ASR: A new paradigm for E2E multi-speaker speech recognition with source localization," in *ICASSP*. 2021, pp. 8433–8437, IEEE.

[33] Rongzhi Gu, Shi-Xiong Zhang, Yuexian Zou, and Dong Yu, "Complex neural spatial filter: Enhancing multi-channel target speech separation in complex domain," *IEEE Signal Process. Lett.*, vol. 28, pp. 1370–1374, 2021.