# THE METHODOLOGY FOR STEREO IMAGE LEARNING REPRESENTATION

*Seongrae Kim*[⋆] *Changwoon Choi*[†] *Sangwoo Han*[‡]

[⋆] Mechanical Engineering, Seoul National University
[†] Electrical & Computer Engineering, Seoul National University
[‡] Earth & Environmental Sciences, Seoul National University

## ABSTRACT

Automating audio generation and synthesis is a key building block of advanced computer listening applications such as auto composition. Whilst various audio generative neural networks exist that target producing audio with both high-fidelity and global structure, far less attention has been paid to generate stereo audio, which is essential to satisfy the listeners. In this paper, we propose a simple but effective training scheme for stereo audio generation task by reparameterizing left-right channels to Mid-Side channels. To address this problem, we first introduce a novel public dataset[1] which features 960 short high-fidelity stereo audios. We also propose new representations, namely *'Side Distance'* and *'Short-time Side Distance'* that effectively capture the stereo image of stereo audio. Our results clearly show that proposed method is superior to the conventional method in generating stereo audio samples on both quantitative and qualitative evaluations.

***Index Terms*—** Generative model, Multi-channel audio, Learning representation

## 1. INTRODUCTION

## 2. RELATED WORK

### 2.1. Audio Generative Model

The earliest audio-generated models for audio tends to focus on speech synthesis. These datasets require handling variable length, and WaveNet [1] used autoregressive method for variable length inputs and outputs. A flow-based WaveGlow [2] has emerged that compensates for slow speed of autoregressive models. In comparison to speech, audio generation for music is relatively in the development stage. [3] proposed to use WaveNet for generate single musical note, but it was still slow and global latent conditioning was impossible. GANSynth [4] with Generative Adversarial Network solves these shortcomings and provides high-fidelity and locally-coherent audio by modeling log magnitudes and instantaneous frequencies with sufficient frequency resolution in the spectral domain. With the current technology, it is possible to learn a single note of a variety of instruments to convincingly describe the sound of a real instrument, and to create a variety of synthesizers for interpolation between two instruments. In spite of such breakthrough development, the reason that the generated musical notes cannot be used for commercial music is that the generated result is a single channel. Despite this breakthrough, the generated musical notes cannot replace commonly used virtual instruments due to the limitation of a single channel.

### 2.2. Multi-Channel Audio

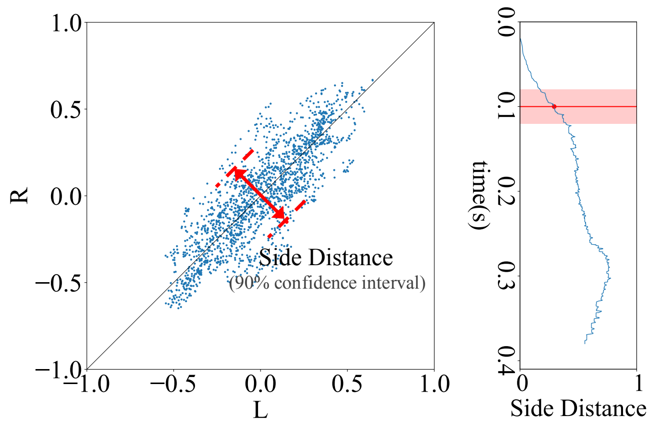
In deep learning, multi-channel settings are mainly used for speech separation [5], speech enhancement [6], and speech recognition [7] due to their ability to utilize information about speech source location. On the other hand, [8] proposed a upmixing conversion of the mono signal to pseudo stereo in order to enhance the audio effect. However, as far as we know, there are still no attempts to generate high-fidelity multi-channel audio. It is thought that it is necessary to learn the difference while maintaining the coherency of both channels to form a spatial sense of sound. A technique referred to as "mid-side coding" exploits the common part of a stereophonic input signal by encoding the sum and difference signals of the two input signals rather than the input signals themselves.( [9]) Therefore, for a high-performance multichannel audio generative model in the future, we would like to train the GANSynth baseline through the mid-side coding to verify whether this attempt is able to effectively learn stereo image.

## 3. PROPOSED METHOD

## 4. EXPERIMENTS

In this section, we first propose our novel representation of stereo audio, *'Side distance' ($D_{side}$) and 'Short-time Side Distance (STSD)'*. Next, we introduce existing metrics for evaluating sample generation, then propose our new metrics for evaluating stereo image generation by combining

[1]Dataset and code are available on `https://github.com/changwoonchoi/ml2020`




(a) Side Distance (b) STSD

**Fig. 1**: Stereo audio representation. (a) shows the vectorscope and side distance, (b) shows the STSD.

existing metrics with side distance and STSD. We then compare the proposed method using M-S channels with previous method(GANSynth [4]) which consumes L-R channels.

### 4.1. Representation for stereo audio

The goal of generative model is to learn to produce samples that look similar to the ones on which it has been trained. Therefore, in order to evaluate the generative model, a proper distance metric between the generated samples and the samples from the dataset is essential. Until now, however, a metric of measuring the similarity between stereo images has never been suggested. In this section, we propose a novel representation of stereo audio that allows us to define the distance metric between them.

#### *4.1.1. Side distance*

When $y$ is a stereo audio of length $T$, we define *'side distance'* $D_{side}$ as following:

$$D_{side}(y) = \frac{\sqrt{2}}{2}\left(\max_{t\in[0,T)}[y_L(t)-y_R(t)] - \min_{t\in[0,T)}[y_L(t)-y_R(t)]\right) \tag{1}$$

where $y_L$ and $y_R$ are left and right channel of stereo audio $y$. For robustness, we use the 0.95, 0.05 quantiles of $y_L(t) - y_R(t)$ instead of maximum and minimum. The side distance can be viewed as an index indicating how far the given stereo audio is spread as visualized in Fig. 1a

#### *4.1.2. Short-time Side Distance (STSD)*

Although side distance is an indicator of how wide the given stereo audio is spread in the auditory space, it cannot express the change of the stereo image over time. Therefore, we introduce the concept of *'Short-time Side Distance (STSD)'* to capture the characteristics of the stereo image over time. The STSD is a sequence of side distance of a windowed signal. The procedure to obtain STSD is to divide a longer time signal into shorter segments of equal length and then compute side distance on each shorter segment. Formally,

$$STFT(y(t)) = D_{side}(y(\tau)w(t)) \tag{2}$$

where $w(t)$ is a window function which is nonzero for short period of time. As depicted in Fig.1b, STSD contains the change in the side distance of the windowed short fragment of signal over time.

### 4.2. Evaluation metrics

Following prior work, we use earth mover's distance (EMD), proposed by [10] to measure the similarity between two stereo audios' STSDs. Formally, EMD is defined as follows:

$$\text{EMD}(s_1, s_2) = \min_{\phi: s_1 \to s_2} \sum_{x \in s_1} \|x - \phi(x)\|_2$$

where $s_1$ and $s_2$ are two distributions and $\phi$ is a bijection between them. Note that $s_1$ and $s_2$ can be any distribution. One can use STSD of stereo audio as $s_1$ and $s_2$.

Let $S_g$ be the set of generated stereo audios and $S_r$ be the set of reference audios with $|S_g| = |S_r|$. To evaluate generative models, we consider the three metrics, MMD, COV which are introduced by [11] and 1-NNA proposed by [12].

- **Coverage (COV)** measures the fraction of stereo audios in the reference set that are matched to at least one stereo audio in the generated set. For each stereo audio in the generated set, its nearest neighbor in the reference set is marked as a match:

$$\text{COV}(S_g, S_r) = \frac{|\{\arg\min_{Y \in S_r} D(X,Y) | X \in S_g\}|}{|S_r|},$$

  where $D(X,Y)$ is a distance metric between two stereo audios. While coverage is able to detect mode collapse, it does not evaluate the quality of generated stereo audios.

- **Minimum matching distance (MMD)** is proposed to complement coverage as a metric that measures quality. For each stereo audio in the reference set, the distance to its nearest neighbor in the generated set is computed and averaged:

$$\text{MMD}(S_g, S_r) = \frac{1}{|S_r|} \sum_{Y \in S_r} \min_{X \in S_g} D(X,Y),$$

  where $D(X,Y)$ is a distance metric between two stereo audios.

| | COV (↑) | MMD (↓) | 1-NNA (↓) |
|---|---|---|---|
| GANSynth [4] | 34.55 | 1.42 | 76.22 |
| Ours | **45.31** | **1.05** | **68.66** |

**Table 1**: Quantitative comparison results of stereo audio generation on our dataset. The best results are marked in bold. Note that MMD is multiplied by $10^3$.

- **1-nearest neighbor accuracy (1-NNA)** is proposed by Lopez-Paz and Oquab [12] for two-sample tests, assessing whether two distributions are identical. Let $S_{-X} = S_r \cup S_g - \{X\}$ and $N_X$ be the nearest neighbor of $X$ in $S_{-X}$. 1-NNA is the leave-one-out accuracy of the 1-NN classifier with given distance metric:

$$\text{1-NNA}(S_g, S_r) = \frac{\sum_{X \in S_g} \mathbb{1}[N_X \in S_g] + \sum_{Y \in S_r} \mathbb{1}[N_Y \in S_r]}{|S_g| + |S_r|},$$

  where $\mathbb{1}[\cdot]$ is the indicator function. For each sample, the 1-NN classifier classifies it as coming from $S_r$ or $S_g$ according to the label of its nearest sample. If $S_g$ and $S_r$ are sampled from the same distribution, the accuracy of such a classifier should converge to $50\%$ given a sufficient number of samples. The closer the accuracy is to $50\%$, the more similar $S_g$ and $S_r$ are, and therefore the better the model is at learning the target distribution.

As definition of COV, MMD, 1-NNA, $D(X,Y)$ can be any distance metric between two audio samples. We use EMD(STSD($X$), STSD($Y$)) as a distance metric $D(X,Y)$.

### 4.3. Experimental results

#### *4.3.1. Dataset*

In this work, we propose a new dataset contains stereo audios with rich stereo images. The dataset consists of EDM stabs with 400ms time duration. Further details can be found in Appendix B.

#### *4.3.2. Quantitative results*

For fair comparison, we trained both our model and GANSynth [4] with exactly same hyperparameters (including network architectures, learning rate, epochs) in our training set. The only difference during the training scheme between GANSynth [4] and proposed method is the channel encoding. Our proposed method consumes two channel audio that is reparameterized from LR representation to MS representation while GANSynth [4] takes input as raw LR represented stereo audios. The quantitative results are reported in Table 1. We observe that it is improved significantly by simply

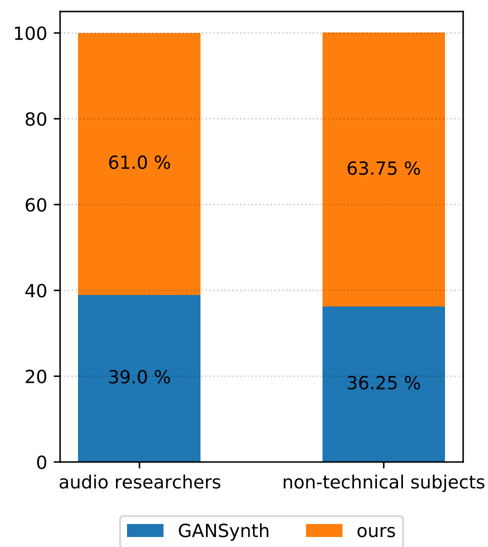



**Fig. 2**: caption

reparameterize the stereo audio in terms of every evaluation metrics (COV, MMD, 1-NNA).

#### *4.3.3. Comparing audio quality to concurrent work*

To better compare our method with concurrent work, we perform a subjective analysis over the stereo audios generated by both methods. In Fig[], we show the percentages of participants based on how they voted for the plausibility comparisons between ours and GANSynth [4]. The study employed 15 participants with different backgrounds - 5 audio researchers, 10 non-technical subjects. The participants are asked to choose the better audio between ours and other work. We can observe that our method received better results on both audio researcher group and non-technical subject group.

## 5. CONCLUSION

In this paper, we propose a novel, yet simple, training scheme for a stereo audio generative model. Also we introduce a new dataset composed of two channel stereo audios with rich stereo images. By reparameterizing the L-R channel into M-S channel, the experiments on the proposed dataset demonstrate that our proposed training scheme gives promising results on every evaluation metrics (MMD, COV, 1-NNA). Furthermore, the study on human evaluations shows that our method is superior to the conventional method in terms of audio quality.

## 6. REFERENCES


[1] Aaron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, and Koray Kavukcuoglu, "Wavenet: A generative model for raw audio," 2016.

[2] R. Prenger, R. Valle, and B. Catanzaro, "Waveglow: A flow-based generative network for speech synthesis," in *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2019, pp. 3617–3621.

[3] Jesse Engel, Cinjon Resnick, Adam Roberts, Sander Dieleman, Mohammad Norouzi, Douglas Eck, and Karen Simonyan, "Neural audio synthesis of musical notes with WaveNet autoencoders," in *Proceedings of the 34th International Conference on Machine Learning*, Doina Precup and Yee Whye Teh, Eds., International Convention Centre, Sydney, Australia, 06–11 Aug 2017, vol. 70 of *Proceedings of Machine Learning Research*, pp. 1068–1077, PMLR.

[4] Jesse Engel, Kumar Krishna Agrawal, Shuo Chen, Ishaan Gulrajani, Chris Donahue, and Adam Roberts, "GANSynth: Adversarial neural audio synthesis," in *International Conference on Learning Representations*, 2019.

[5] A. A. Nugraha, A. Liutkus, and E. Vincent, "Multichannel audio source separation with deep neural networks," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 24, no. 9, pp. 1652–1664, 2016.

[6] S. Araki, T. Hayashi, M. Delcroix, M. Fujimoto, K. Takeda, and T. Nakatani, "Exploring multi-channel features for denoising-autoencoder-based speech enhancement," in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2015, pp. 116–120.

[7] X. Xiao, S. Watanabe, H. Erdogan, L. Lu, J. Hershey, M. L. Seltzer, G. Chen, Y. Zhang, M. Mandel, and D. Yu, "Deep beamforming networks for multi-channel speech recognition," in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2016, pp. 5745–5749.

[8] Marco Fink, Sebastian Kraft, and Udo Zölzer, "Downmixcompatibe conversion from mono to stereo in time- and frequency-domain," in *in Proc. of the 18th Int. Conference on Digital Audio Effects*, 2015.

[9] J. D. Johnston and A. J. Ferreira, "Sum-difference stereo transform coding," in *[Proceedings] ICASSP-92: 1992 IEEE International Conference on Acoustics, Speech, and Signal Processing*, 1992, vol. 2, pp. 569–572 vol.2.

[10] Y. Rubner, C. Tomasi, and L. J. Guibas, "A metric for distributions with applications to image databases," in *Sixth International Conference on Computer Vision (IEEE Cat. No.98CH36271)*, 1998, pp. 59–66.

[11] Panos Achlioptas, Olga Diamanti, Ioannis Mitliagkas, and Leonidas J Guibas, "Learning representations and generative models for 3d point clouds," *arXiv preprint arXiv:1707.02392*, 2017.

[12] David Lopez-Paz and Maxime Oquab, "Revisiting Classifier Two-Sample Tests," *arXiv e-prints*, p. arXiv:1610.06545, Oct. 2016.

## A. ADDITIONAL IMPLEMENTATION DETAILS

In this section we provide several additional details that are not provided in the main sections.

### A.1. Neural Network Architecture

| Generator | Output Size | $k_{width}$ | $k_{height}$ | $k_{filters}$ | Nonlinearity |
|---|---|---|---|---|---|
| concat(Z, pitch) | (1, 1, 264) | - | - | - | - |
| conv2d | (2, 16, 256) | 2 | 16 | 256 | PN(LReLU) |
| conv2d | (2, 16, 256) | 3 | 3 | 256 | PN(LReLU) |
| upsample 2x2 | (4, 32, 256) | - | - | - | - |
| conv2d | (4, 32, 256) | 3 | 3 | 256 | PN(LReLU) |
| conv2d | (4, 32, 256) | 3 | 3 | 256 | PN(LReLU) |
| upsample 2x2 | (8, 64, 256) | - | - | - | - |
| conv2d | (8, 64, 256) | 3 | 3 | 256 | PN(LReLU) |
| conv2d | (8, 64, 256) | 3 | 3 | 256 | PN(LReLU) |
| upsample 2x2 | (16, 128, 256) | - | - | - | - |
| conv2d | (16, 128, 256) | 3 | 3 | 256 | PN(LReLU) |
| conv2d | (16, 128, 256) | 3 | 3 | 256 | PN(LReLU) |
| upsample 2x2 | (32, 256, 256) | - | - | - | - |
| conv2d | (32, 256, 128) | 3 | 3 | 128 | PN(LReLU) |
| conv2d | (32, 256, 128) | 3 | 3 | 128 | PN(LReLU) |
| upsample 2x2 | (32, 512, 128) | - | - | - | - |
| conv2d | (32, 512, 64) | 3 | 3 | 64 | PN(LReLU) |
| conv2d | (32, 512, 64) | 3 | 3 | 64 | PN(LReLU) |
| upsample 2x2 | (32, 1024, 64) | - | - | - | - |
| conv2d | (32, 1024, 32) | 3 | 3 | 32 | PN(LReLU) |
| conv2d | (32, 1024, 32) | 3 | 3 | 32 | PN(LReLU) |
| generator output | (32, 1024, 4) | 1 | 1 | 4 | Tanh |
| **Discriminator** | **Output Size** | $k_{width}$ | $k_{height}$ | $k_{filters}$ | **Nonlinearity** |
| spectrogram image | (32, 1024, 4) | - | - | - | - |
| conv2d | (32, 1024, 32) | 1 | 1 | 32 | LReLU |
| conv2d | (32, 1024, 32) | 3 | 3 | 32 | LReLU |
| conv2d | (32, 1024, 32) | 3 | 3 | 32 | LReLU |
| downsample 1x2 | (32, 512, 32) | - | - | - | - |
| conv2d | (32, 512, 64) | 3 | 3 | 64 | LReLU |
| conv2d | (32, 512, 64) | 3 | 3 | 64 | LReLU |
| downsample 1x2 | (32, 256, 64) | - | - | - | - |
| conv2d | (32, 256, 128) | 3 | 3 | 128 | LReLU |
| conv2d | (32, 256, 128) | 3 | 3 | 128 | LReLU |
| downsample 2x2 | (16, 128, 128) | - | - | - | - |
| conv2d | (16, 128, 256) | 3 | 3 | 256 | LReLU |
| conv2d | (16, 128, 256) | 3 | 3 | 256 | LReLU |
| downsample 2x2 | (8, 64, 256) | - | - | - | - |
| conv2d | (8, 64, 256) | 3 | 3 | 256 | LReLU |
| conv2d | (8, 64, 256) | 3 | 3 | 256 | LReLU |
| downsample 2x2 | (4, 32, 256) | - | - | - | - |
| conv2d | (4, 32, 256) | 3 | 3 | 256 | LReLU |
| conv2d | (4, 32, 256) | 3 | 3 | 256 | LReLU |
| downsample 2x2 | (2, 16, 256) | - | - | - | - |
| concat(x, minibatch std.) | (2, 16, 257) | - | - | - | - |
| conv2d | (2, 16, 256) | 3 | 3 | 256 | LReLU |
| conv2d | (2, 16, 256) | 3 | 3 | 256 | LReLU |
| discriminator output | (1, 1, 1) | - | - | 1 | - |

**Table 2**: Full description of our neural network architecture.

**A.2.**

## B. DATASET DETAILS