A SYNTHETIC CORPUS GENERATION METHOD FOR NEURAL VOCODER TRAINING

Zilin Wang[1], Peng Liu[2], Jun Chen[1], Sipan Li[1,5], Jinfeng Bai[3], Gang He[3], Zhiyong Wu[1,4,6,], Helen Meng[1,6]*

[1]Shenzhen International Graduate School, Tsinghua University, Shenzhen, China
[2]Transsion, Shanghai [3]TAL Education Group, Beijing [4]Peng Cheng Lab, Shenzhen, China
[5] Tencent AI Lab, Shenzhen [6]The Chinese University of Hong Kong, Hong Kong SAR, China
wangzl21@mails.tsinghua.edu.cn, peng.liu6@transsion.com, zywu@sz.tsinghua.edu.cn

ABSTRACT

Nowadays, neural vocoders are preferred for their ability to synthesize high-fidelity audio. However, training a neural vocoder requires a massive corpus of high-quality real audio, and the audio recording process is often labor-intensive. In this work, we propose a synthetic corpus generation method for neural vocoder training, which can easily generate synthetic audio with an unlimited number at nearly no cost. We explicitly model the prior characteristics of audio from multiple target domains simultaneously (e.g., speeches, singing voices, and instrumental pieces) to equip the generated audio data with these characteristics. And we show that our synthetic corpus allows the neural vocoder to achieve competitive results without any real audio in the training process. To validate the effectiveness of our proposed method, we performed empirical experiments on both speech and music utterances in subjective and objective metrics. The experimental results show that the neural vocoder trained with the synthetic corpus produced by our method can generalize to multiple target scenarios and has excellent singing voice (MOS: 4.20) and instrumental piece (MOS: 4.00) synthesis results.

Index Terms— neural vocoder, synthetic corpus, speech synthesis

1. INTRODUCTION

Vocoder has drawn much attention as it synthesizes the final waveform from acoustic information in many applications like text-to-speech [1], singing voice synthesis [2], voice conversion [3] and speech-to-speech translation [4], etc. The mainstream vocoders can be broadly divided into traditional vocoders [5–7] and deep learning based vocoders, i.e., neural vocoders [8–11]. The latter often require large amounts of data for training. With the recent explosion of deep neural networks, neural vocoders, which can offer state-of-the-art speech synthesis quality, have gradually become the go-to choice.

As well as many other deep learning-based applications, one bottleneck to optimizing the neural vocoder systems stems from the acquisition of the training dataset, i.e., high-quality and noise-free audio. Nevertheless, the collection of usable audio data is not easy, whose process usually requires the speakers to vocalize for a long time in a professional recording environment. In addition, data-driven approaches often suffer from overfitting, i.e., neural vocoder tends to have good performance only within the domain of training data, whereas synthesis quality degrades dramatically for out-of-domain audio from unseen speakers, genders, languages, and styles [12, 13]. This issue calls for adding more audio data with more diversified forms to the training dataset, which has further pushed up the demand for high-quality corpus. Consequently, how to obtain a vocoder that can generalize to more scenarios while using less, even in the absence of training corpus, gradually becomes one of the focuses of the vocoder community.

To address this problem, some previous works [12–14] try to enhance the universality and generalization of the vocoder to avoid data consumption in fine-tuning when encountering out-of-domain inference data. However, these methods do not solve the fundamental problem of lack of training data, as these universal vocoders tend to consume more high-quality real data during their training process. A common approach to address data sparsity is data augmentation [15–17]. Nevertheless, data augmentation relies heavily on the original data and is less effective in generalizing to scenarios with significant variations. Noteworthy, in [18], authors proposed a flow-based vocoder SiD-WaveFlow for low-resource speech synthesis, which can be established with only 5 minutes speeches. Although its generalization performance is unproven, we believe low resource waveform synthesis is one of the effective solutions to mitigate the data scarcity problem indeed.

Unlike other methods, we take the perspective of a complete alternative to real data and seek to answer the question: **Is it possible to use synthetic data instead of real data for neural vocoder training?**

Why do we consider synthetic data? Compared with real data, synthetic data has at least the following advantages: It can be generated online and is easy to acquire; It is copyright-free and can be widely used in both industry and academia; It saves massive material and resource costs during real human voice recording; Its content can be freely adjusted according to our wishes without constraining from recording conditions. Actually, synthetic data has long been used with good results in many areas like computer vision [19], automatic driving [20] and speech processing [21–23] etc. Nonetheless, the feasibility of training a vocoder with synthetic data solely has yet to be proven.

Why is synthetic data feasible for vocoder? During the process of restoring waveforms from acoustic features, the vocoder's task is to recover missing phase and other detailed information. This issue is irrelevant to whether the waveform is speech, music, or other sound events. Therefore, using synthesized data to train a vocoder is feasible.

What kind of synthetic data should be used? Again, the vocoder's task is to learn a content-independent mapping from acoustic features, e.g., mel-spectrograms, to raw waveforms while leaving the audio's content and speaker identity information to be modeled by the acoustic model. Hense, we do not need to consider semantic information when modeling characteristics of audio from target domains but only need the produced audio in the corpus to have the corresponding prior patterns. The examples from our synthetic corpus and the audio synthesized by the vocoder trained with them are available at our demo page[1].

In this paper, we present a synthetic corpus generation method to tackle the real data sparsity problem during neural vocoder training. We choose speeches and musical pieces (including singing

* Corresponding author.

[1]Demo page: https://zerlinwang.github.io/synthetic-corpus-vocoder/

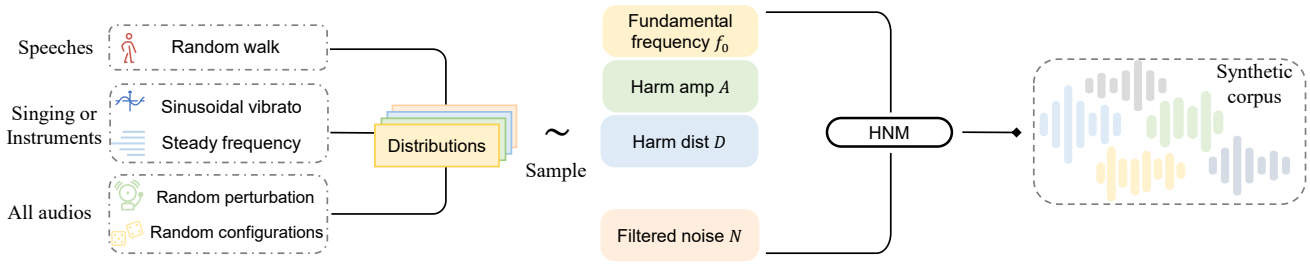

Fig. 1. The pipeline of our proposed method. At first, based on the prior knowledge from different target audio domains, we model the distributions of acoustic features. Then we sample the acoustic features, including fundamental frequency f_0, harmonic amplitude A, harmonic distribution D, and time-varying filtered noise signal N from corresponding distributions, respectively. Lastly, audio is synthesized based on the sampled acoustic features by Harmonic-plus-Noise Model and forms the synthetic corpus.

voices and instrument pieces) as the target scenarios for the audio in synthetic corpus since vocoders are mainly used in these scenarios. Specifically, we introduce the Random Walk (RW) model to fit with uncertainty in speeches and add sine and uniform constituents into the fundamental frequency to simulate the vibrato and steady long tones respectively in musical pieces. By these heuristic methods, we construct the distributions of acoustic features, including harmonic fundamental frequency, harmonics amplitude, harmonic distribution, and filtered noise. At last, we adopt Harmonic-plus-Noise Model (HNM) for corpus generation, in which acoustic features sampled from the constructed corresponding distributions are used as the *components* to synthesize audio belonging to the synthetic corpus. The experimental results show the feasibility of our method in all target scenarios.

In general, the contributions of this paper are three-fold:

- We propose a novel and effective synthetic corpus generation method for neural vocoder training.
- We demonstrate that our proposed method enables the vocoder to synthesize competitive results without real audio and simultaneously generalize to scenarios such as human voices and musical pieces.
- To the best of our knowledge, this is the first case to validate the feasibility of training neural vocoders with synthetic data alone, which is an under-explored avenue for further research.

2. METHODOLOGY

This section is organized as follows: Section 2.1 will introduce the overall pipeline of the proposed synthetic corpus generation method. Section 2.2 elaborates the detail of acoustic features production. Section 2.3 narrates the Harmonic-plus-Noise Model shortly.

2.1. Overall pipeline

The pipeline of our proposed corpus synthesis method for neural vocoder training is represented in Figure 1. It consists of three main stages. To begin with, based on the prior knowledge from different target audio domains, including speeches, singing voices, and instrument pieces, we model the distributions of acoustic features corresponding to fundamental frequency f_0, harmonic distribution D, harmonic amplitude A, and time-varying filtered noise N. Next, a bulk of acoustic features are sampled from the modeled distributions. Lastly, we synthesize the audio in the corpus after summing the harmonic and filtered noise components of the acoustic features according to the Harmonic-plus-Noise Model (HNM).

2.2. Acoustic features generation

The baseline algorithm of our acoustic features generation is motivated by [22]. In [22], authors used self-supervised loss on synthetic data to help reconstruct losses on real data for the task of timbre and pitch detection. However, our synthetic corpus serves as the *sole*

Algorithm 1 The specific algorithm flow of fundamental frequency generation

Input: N is the random length of the segment. p_s, p_o p_v and p_{rw} are the probability of silent, oscillation, vibrato and random walk, respectively. $rand()$ denotes the random number generator in $[0, 1]$. $zeros(N)$ and $ones(N)$ return all-zero and all-one vector with the length N respectively. $cumsum(\cdot)$ returns the cumulative sum of the elements. $\mathcal{N}$ means the Gaussian distribution, and μ and σ are the distribution parameters. $linspace(\cdot)$ returns evenly spaced samples. 'random perturbation' means noise sampled from normal distribution with random mean and variance. The summation of scalars and vectors (bold) adopts python broadcast.

Output: $\mathbf{f_0}$: denotes the generated fundamental frequency.

if $rand() < p_s$ **then**
  $\mathbf{f_0} \leftarrow zeros(N)$
else
  $f_0^{basis} \leftarrow$ random fundamental frequency basis
  if $rand() < p_o$ **then**
    if $rand() < p_{rw}$ **then**
      % random walk for speeches
      displacement $\leftarrow cumsum(\mathcal{N}(\mu, \sigma^2, N))$
      smoothed_dis $\leftarrow$ smooth the displacement
      curve $\leftarrow$ interpolate(smoothed_dis, N)
    else
      $\mathcal{E} \leftarrow$ random exponent
      curve $\leftarrow linspace(rand(), rand(), N)^{\mathcal{E}}$
    end if
    $v_{max}, v_{min} \leftarrow$ random maximum and minimum value
    curve $\leftarrow$ normalization(curve, v_{max}, v_{min})
    if $rand() < p_v$ **then**
      % sinusoidal vibrato for musical pieces
      $T \leftarrow$ random period
      vibrato $\leftarrow$ curve $\times sin(linspace(0, 2\pi \times T, N))$
      $\mathbf{f_0} \leftarrow f_0^{basis}$ + vibrato
    else
      $\mathbf{f_0} \leftarrow f_0^{basis}$ + curve
    end if
  else
    % steady f_0 for musicial pieces
    $\mathbf{f_0} \leftarrow f_0^{basis} \times ones(N)$
  end if
  $\mathbf{f_0} \leftarrow \mathbf{f_0}$ + random perturbation
end if
return $\mathbf{f_0}$

training data to train the *neural vocoder* without any intervention of real audio.

Specifically, in our method, feature generation is not a one-shot process. Instead, we produce a short segment with a random length at a time to populate the time-varying acoustic feature vector. For segments, at first, there is a possibility of being silent. Then, each segment that is not silent is given random sampled fundamental frequency, harmonic amplitude, harmonic distribution, and filtered noise. Finally, a random perturbation will be added to the not silent segment. For more preciously narrating, we show the specific algorithm flow of fundamental frequency generation in Algorithm 1 as an example. The other acoustic features are generated similarly.

As shown in Figure 1, besides the random perturbation and configurations for all audio, we mainly pay attention to the characteristics of speeches, singing voices, and instrumental musical pieces.

For speeches, the fundamental frequency of the human voice in the time domain is non-monotonic on the macroscopic and oscillates on the microscopic. Thus, we adopt a one-dimensional Random Walk (RW) [24] model to fit with it. Random Walk Theory is a common statistical model composed of a sequence of trajectories, each of which is random, like the record of a random process formed by a drunken person's steps. It is often leveraged to represent irregular forms of change. In our implementation, as shown in the orange part in Algorithm 1, we model the displacement at each step as a random variable sampled from a one-dimensional normal distribution and use the cumulative sum of displacement as the value for each frame step:

$$\mathrm{x}_t \sim \mathcal{N}(\mu, \sigma^2) \tag{1}$$

$$x_t = \sum_{t'=0}^{t} \mathrm{x}_{t'}, \tag{2}$$

where $t \in \{0, 1, \cdots, N-1\}$ denotes the frame step and N denotes the total length of this segment. $\mathcal{N}(\mu, \sigma^2)$ denotes a one-dimensional normal distribution with mean μ and standard deviation σ. x_t and x_t denote the step displacement and the cumulative sum of displacement at frame step t, respectively. Then, to comply with the short-time invariance of speeches, we smooth the curve:

$$y = x * w, \tag{3}$$

$$y_t = \sum_{j=0}^{l} x_{t-j} w_j, \tag{4}$$

where y_t and x_t denotes the value of smoothed curve y and original curve x at frame step t respectively. $*$ denotes the convolution operation. And w denotes a windowing function with the length l. Here we let $w = \mathbb{1}/l$ and $\mathbb{1}$ denotes an all-one vector. At last, we obtain the curve value by interpolation according to the shape of the smoothed displacement.

On the other hand, an essential feature of singing and instrumental musical pieces is the presence of long tones on some notes. Based on long tones, the presence of vibrato is a prevalent trick to increase expressiveness in singing or instrumental pieces. Therefore, whether the synthesized waveform has pretty vibrato will be a critical evaluation criterion for the performance of the vocoder. Vibrato usually contains a regular, pulsating oscillation (excursion) of the pitch. As a result, for the simulation of vibrato, the curve of fundamental frequency segment is multiplied by a sine wave-like constituent with a certain probability p_v. The cyan part in Algorithm 1 illustrate this process. In addition to vibrato, it is also possible to be a steady fundamental frequency without any oscillations during the long tone phase. As presented in the blue part of Algorithm 1, we achieve the steady fundamental frequency segment by broadcasting the f_0^{basis} to the length of the segment N.

2.3. Harmonic-plus-Noise Model recap

Harmonic-plus-Noise Model (HNM) was firstly proposed in [25]. It can flexibly adjust the amplitude, envelope, and fundamental frequency of audio respectively, which can precisely meet our goal to simulate target audio from different domains. The effectiveness of training the model using the HNM synthetic corpus has been verified in some works [22, 23, 26].

In HNM, audio signal $s(t)$ can be represented as the sum of the harmonic $s_h(t)$ and noise components $s_n(t)$:

$$s(t) = s_h(t) + s_n(t). \tag{5}$$

For the voiced part, the signal can be approximated by superimposing a series of harmonic components whose pitches are the integer multiples of the fundamental frequency:

$$s_h(t) = \sum_{k=0}^{K} A_k(t) \sin \phi_k(t), \tag{6}$$

in which K denotes the number of harmonics. $A_k(t)$ denotes the amplitude and $\phi_k(t)$ denotes the instantaneous phase. The phase $\phi_k(t)$ is obtained by integrating the instantaneous frequency $f_k(t)$:

$$\phi_k(t) = 2\pi \sum_{m=0}^{t} f_k(m) + \phi_{0,k}, \tag{7}$$

where $\phi_{0,k}$ is the initial phase. And for the unvoiced part, we use an FIR filter for noise modulation in the frequency domain:

$$S_n(t) = H_t N_t, \tag{8}$$

where $H_t = \mathrm{DFT}(h_t)$ is the FIR filter and $N_t = \mathrm{DFT}(n_t)$ is white noise signal. We recover the frame-wise filtered noise $s_n(t) = \mathrm{IDFT}(S_n(t))$.

3. EXPERIMENTS

In this section, to verify the effectiveness of our proposed synthetic corpus generation method, we constructed a neural vocoder trained with the synthetic dataset we produced and tested its vocoding performance in a wide range of scenarios.

3.1. Experiment settings

For experiments, we employed Differentiable Digital Signal Processing (DDSP) [26] as the implementation of Harmonic-plus-Noise Model to generate the synthetic corpus. We synthesized 1,000,000 pieces of synthetic audio with a sample rate of 24,000 for training. The popular HiFi-GAN [8] is selected as our test neural vocoder. **Neural Vocoder (1M)** denotes the HiFi-GAN vocoder trained with all the 1,000,000 pieces of synthetic audio. There are two other vocoders set for comparison with the vocoder trained with our synthetic corpus:

Griffin-Lim [5]: a traditional waveform synthesis method that also does not rely on real data.

Neural Vocoder (Real): a HiFi-GAN vocoder trained with real corpus. We chose the same HiFi-GAN model as the representative of neural vocoders trained with real corpus for fair comparisons. Its training dataset is LJSpeech [27], a popular open-source single female English speaker dataset has a total length of approximately 24 hours. We directly used the official open-source trained model [2] in our experiments.

In addition, to compare the performance of our synthetic corpus to the real corpus with the same amount of data, we train an extra HiFi-GAN model with a small part of the synthetic corpus

[2]Implementation and trained model: https://github.com/jik876/hifi-gan

Table 1. Subjective evaluation results (MOS values). "Neural Vocoder (Real)" denotes the vocoder trained with real corpus LJSpeech. "Neural Vocoder (10K)" denotes the vocoder trained with 10,000 pieces of synthetic audio and "Neural Vocoder (1M)" denotes the vocoder trained with 1,000,000 pieces of synthetic audio. "CI" denotes the confidence interval. † denotes the neural vocoder trained with synthetic corpus generated by our proposed method.

Model	speeches (male)		speeches (female)		singing voices		instrumental pieces	
	MOS	95% CI	MOS	95% CI	MOS	95% CI	MOS	95% CI
Ground Truth	4.76	± 0.06	4.64	± 0.07	4.91	± 0.03	4.68	± 0.07
Griffin Lim	2.03	± 0.08	1.73	± 0.07	1.64	± 0.07	1.54	± 0.07
Neural Vocoder (Real)	3.05	± 0.10	**4.47**	± 0.07	3.28	± 0.09	2.77	± 0.10
Neural Vocoder† (10K)	2.66	± 0.10	3.37	± 0.10	3.04	± 0.09	3.91	± 0.09
Neural Vocoder† (1M)	**3.18**	± 0.09	3.81	± 0.08	**4.20**	± 0.08	**4.00**	± 0.08

Table 2. Objective evaluation results (PESQ and STOI values). "Neural Vocoder (Real)" denotes the vocoder trained with real corpus LJSpeech. "Neural Vocoder (10K)" denotes the vocoder trained with 10,000 pieces of synthetic audio and "Neural Vocoder (1M)" denotes the vocoder trained with 1,000,000 pieces of synthetic audio. † denotes the neural vocoder trained with synthetic corpus generated by our proposed method.

Metric	Model	speeches (male)	speeches (female)	singing voices
PESQ	Griffin-Lim	1.989	1.382	0.517
	Neural Vocoder (Real)	3.050	**3.577**	2.891
	Neural Vocoder† (10K)	2.909	3.074	3.013
	Neural Vocoder† (1M)	**3.121**	3.272	**3.226**
STOI	Griffin-Lim	79.29	77.11	56.96
	Neural Vocoder (Real)	83.77	87.43	79.35
	Neural Vocoder† (10K)	84.43	84.93	78.38
	Neural Vocoder† (1M)	**88.28**	**88.40**	**82.06**

(10,000 pieces of synthetic audio and nearly the same total length as the LJSpeech). We name it as **Neural Vocoder (10K)**, which will be talked about in Section 3.3.

We evaluated vocoders on several dimensions, including a subjective metric, mean opinion score (MOS) of audio quality (Table 1), and two objective metrics, perceptual evaluation of speech quality (PESQ) and short-term objective intelligibility (STOI) (Table 2). For each evaluation, we selected ten clips for the MOS test and three hundred clips for the PESQ and STOI tests. A total of twenty-seven people participated in the MOS test.

3.2. Experimental results of the full synthetic corpus (1M)

3.2.1. Speeches

In this evaluation, we extracted audio clips from the VCTK dataset [28], which contains English speech from speakers with different accents. As the LJSpeech is a female speaker dataset, for a fair comparison, we evaluated the vocoders on male speech data and female speech data, respectively.

For male speech, the vocoder trained with our synthetic corpus achieved the MOS score of 3.18, PESQ score of 3.121, and STOI score of 88.28, the best of the three vocoders. The results showed that the vocoders trained with our synthetic corpus can be employed in synthesizing speech waveforms from male speakers but with average audio quality. And for female speech, the vocoder trained with our synthetic corpus achieved the MOS score of 3.81, PESQ score of 3.272, and STOI score of 88.40, above-average results. However, the results of the vocoder trained with real LJSpeech were higher than with synthetic corpus in MOS and PESQ. This defeat is likely due to the LJSpeech being a female speech English dataset, similar to the setting of the VCTK dataset, which may lead to *overfitting* on the female speech data. And the experimental results on the other scenarios confirmed our conjecture (the performances of Neural Vocoder (Real) were relatively poor for male speeches, singing voices, and instrumental pieces).

Overall, experimental results on speeches proved the effectiveness of leveraging a Random Walk model to approximate the variation of the fundamental frequency of the speeches. What's more important, the results indeed demonstrated the feasibility of using synthetic data to train neural vocoders in the field of speech.

3.2.2. Singing voices

We evaluated our method on singing voice clips extracted from the Mandarin singing corpus dataset Opencpop [29]. Neural vocoder trained with synthetic corpus achieved the highest scores in all metrics, i.e., MOS score of 4.20, PESQ score of 3.226, and STOI score of 82.06, which significantly outperformed other vocoders' results. In other words, our generation approach of the synthetic corpus has outstanding effectiveness in singing voices synthesis.

3.2.3. Musical pieces

Since PESQ and STOI are not suitable metrics for the music, we only performed subjective MOS evaluations for musical pieces. Audio clips were extracted from the single instrument musical pieces of URMP dataset [30]. As shown in Table 1, the neural vocoder trained with synthetic corpus significantly outperformed other vocoders, achieving a MOS of 4.00. This result showed that the vocoder trained with synthetic corpus generated by our method excels for music synthesis tasks.

3.3. Investigation of the small synthetic corpus (10K)

As shown in Table 1 and Table 2, although the vocoder trained with the small synthetic corpus was beaten by the vocoder trained with real data in speeches and singing voices for both objective and subjective metrics (besides STOI of male speeches), it was still usable in female speeches and musical pieces, especially MOS score of 3.91 for instrumental pieces synthesis. This showed that it is hard for synthetic data to beat high-quality real data with the same total length. However, the collection of the synthetic corpus is at nearly no cost. Compared with huge time and computational power consumption during the neural vocoder training process, the consumption between the generation of 10,000 and 1,000,000 pieces of synthetic audio is almost no different.

4. CONCLUSION

In this paper, to overcome the problem of data scarcity during neural vocoder training, we propose a synthetic corpus generation method, which can generate an unlimited amount of corpus without any real audio. Experimental results show that the neural vocoder trained with the synthetic corpus generated by our method can be employed in speech synthesis and achieves excellent performance in synthesizing singing voices and instrumental pieces. Comparisons with stronger baselines and exploring better models of speech fundamental frequencies will be carried out in future work.

Acknowledgement This work is supported by National Natural Science Foundation of China (62076144), the Major Key Project of PCL (PCL2021A06, PCL2022D01), Shenzhen Science and Technology Program (WDZC20220816140515001) and Tencent AI Lab Rhino-Bird Focused Research Program (RBFR2022005).

5. REFERENCES

[1] Yi Ren, Jinglin Liu, and Zhou Zhao, "Portaspeech: Portable and high-quality generative text-to-speech," *Advances in Neural Information Processing Systems*, vol. 34, 2021.

[2] Jiawei Chen, Xu Tan, Jian Luan, Tao Qin, and Tie-Yan Liu, "Hifisinger: Towards high-fidelity neural singing voice synthesis," *arXiv preprint arXiv:2009.01776*, 2020.

[3] Takuhiro Kaneko, Hirokazu Kameoka, Kou Tanaka, and Nobukatsu Hojo, "Maskcyclegan-vc: Learning non-parallel voice conversion with filling in frames," in *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2021, pp. 5919–5923.

[4] Ye Jia, Ron J Weiss, Fadi Biadsy, Wolfgang Macherey, Melvin Johnson, Zhifeng Chen, and Yonghui Wu, "Direct speech-to-speech translation with a sequence-to-sequence model," *Proc. Interspeech 2019*, pp. 1123–1127, 2019.

[5] D. Griffin and Jae Lim, "Signal estimation from modified short-time fourier transform," in *ICASSP '83. IEEE International Conference on Acoustics, Speech, and Signal Processing*, 1983, vol. 8, pp. 804–807.

[6] Masanori MORISE, Fumiya YOKOMORI, and Kenji OZAWA, "World: A vocoder-based high-quality speech synthesis system for real-time applications," *IEICE Transactions on Information and Systems*, vol. E99.D, no. 7, pp. 1877–1884, 2016.

[7] Hideki Kawahara, "Straight, exploitation of the other aspect of vocoder: Perceptually isomorphic decomposition of speech sounds," *Acoustical Science and Technology*, vol. 27, no. 6, pp. 349–353, 2006.

[8] Jungil Kong, Jaehyeon Kim, and Jaekyoung Bae, "Hifi-gan: Generative adversarial networks for efficient and high fidelity speech synthesis," *Advances in Neural Information Processing Systems*, vol. 33, pp. 17022–17033, 2020.

[9] Takuhiro Kaneko, Kou Tanaka, Hirokazu Kameoka, and Shogo Seki, "Istftnet: Fast and lightweight mel-spectrogram vocoder incorporating inverse short-time fourier transform," in *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2022, pp. 6207–6211.

[10] Rongjie Huang, Max WY Lam, Jun Wang, Dan Su, Dong Yu, Yi Ren, and Zhou Zhao, "Fastdiff: A fast conditional diffusion model for high-quality speech synthesis," *arXiv preprint arXiv:2204.09934*, 2022.

[11] Sang-gil Lee, Wei Ping, Boris Ginsburg, Bryan Catanzaro, and Sungroh Yoon, "Bigvgan: A universal neural vocoder with large-scale training," *arXiv preprint arXiv:2206.04658*, 2022.

[12] Jaime Lorenzo-Trueba, Thomas Drugman, Javier Latorre, Thomas Merritt, Bartosz Putrycz, Roberto Barra-Chicote, Alexis Moinet, and Vatsal Aggarwal, "Towards Achieving Robust Universal Neural Vocoding," in *Proc. Interspeech 2019*, 2019, pp. 181–185.

[13] Yunlong Jiao, Adam Gabryś, Georgi Tinchev, Bartosz Putrycz, Daniel Korzekwa, and Viacheslav Klimkov, "Universal neural vocoding with parallel wavenet," in *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2021, pp. 6044–6048.

[14] Won Jang, Dan Lim, and Jaesam Yoon, "Universal melgan: A robust neural vocoder for high-fidelity waveform generation in multiple domains," *arXiv preprint arXiv:2011.09631*, 2020.

[15] Shengyuan Xu, Wenxiao Zhao, and Jing Guo, "RefineGAN: Universally Generating Waveform Better than Ground Truth with Highly Accurate Pitch and Intensity Responses," in *Proc. Interspeech 2022*, 2022, pp. 1591–1595.

[16] Max Morrison, Zeyu Jin, Nicholas J Bryan, Juan-Pablo Caceres, and Bryan Pardo, "Neural pitch-shifting and time-stretching with controllable lpcnet," *arXiv preprint arXiv:2110.02360*, 2021.

[17] Yi-Chiao Wu, Cheng-Hung Hu, Hung-Shin Lee, Yu-Huai Peng, Wen-Chin Huang, Yu Tsao, Hsin-Min Wang, and Tomoki Toda, "Relational data selection for data augmentation of speaker-dependent multi-band melgan vocoder," *arXiv preprint arXiv:2106.05629*, 2021.

[18] Yuhan Li, Ying Shen, Dongqing Wang, and Lin Zhang, "SiD-WaveFlow: A Low-Resource Vocoder Independent of Prior Knowledge," in *Proc. Interspeech 2022*, 2022, pp. 1616–1620.

[19] Yue Yao, Liang Zheng, Xiaodong Yang, Milind Naphade, and Tom Gedeon, "Simulating content consistent vehicle datasets with attribute descent," *arXiv: Computer Vision and Pattern Recognition*, 2019.

[20] German Ros, Laura Sellart, Joanna Materzynska, David Vazquez, and Antonio M. López, "The synthia dataset: A large collection of synthetic images for semantic segmentation of urban scenes," in *Computer Vision and Pattern Recognition*, 2016.

[21] Joseph Turian, Jordie Shier, George Tzanetakis, Kirk McNally, and Max Henry, "One billion audio sounds from gpu-enabled modular synthesis," *synthesis*, vol. 20, pp. 3, 2021.

[22] Jesse Engel, Rigel Swavely, Lamtharn Hanoi Hantrakul, Adam Roberts, and Curtis Hawthorne, "Self-supervised pitch detection by inverse audio synthesis," in *ICML 2020 Workshop on Self-supervision in Audio and Speech*, 2020.

[23] Yusong Wu, Josh Gardner, Ethan Manilow, Ian Simon, Curtis Hawthorne, and Jesse Engel, "The chamber ensemble generator: Limitless high-quality mir data via generative modeling," *arXiv preprint arXiv:2209.14458*, 2022.

[24] Karl Pearson, "The problem of the random walk," *Nature*, vol. 72, no. 1865, pp. 294–294, 1905.

[25] J. Laroche, Y. Stylianou, and E. Moulines, "Hnm: a simple, efficient harmonic+noise model for speech," in *Proceedings of IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, 1993, pp. 169–172.

[26] Jesse Engel, Lamtharn (Hanoi) Hantrakul, Chenjie Gu, and Adam Roberts, "Ddsp: Differentiable digital signal processing," in *International Conference on Learning Representations*, 2020.

[27] Keith Ito and Linda Johnson, "The lj speech dataset," `https://keithito.com/LJ-Speech-Dataset/`, 2017.

[28] J. Yamagishi and K. Edwards, "Voice cloning toolkit for festival and hts," 2010.

[29] Yu Wang, Xinsheng Wang, Pengcheng Zhu, Jie Wu, Hanzhao Li, Heyang Xue, Yongmao Zhang, Lei Xie, and Mengxiao Bi, "Opencpop: A high-quality open source chinese popular song corpus for singing voice synthesis," *arXiv preprint arXiv:2201.07429*, 2022.

[30] Bochen Li, Xinzhao Liu, Karthik Dinesh, Zhiyao Duan, and Gaurav Sharma, "Creating a multitrack classical music performance dataset for multimodal music analysis: Challenges, insights, and applications," *Trans. Multi.*, vol. 21, no. 2, pp. 522–535, feb 2019.