SPEECHSPLIT2.0: UNSUPERVISED SPEECH DISENTANGLEMENT FOR VOICE CONVERSION WITHOUT TUNING AUTOENCODER BOTTLENECKS

Chak Ho Chan[⋆] *Kaizhi Qian*[†] *Yang Zhang*[†] *Mark Hasegawa-Johnson*[⋆]

[⋆] University of Illinois at Urbana-Champaign
[†]MIT-IBM Watson AI Lab

ABSTRACT

SPEECHSPLIT can perform aspect-specific voice conversion by disentangling speech into content, rhythm, pitch, and timbre using multiple autoencoders in an unsupervised manner. However, SPEECHSPLIT requires careful tuning of the autoencoder bottlenecks, which can be time-consuming and less robust. This paper proposes SPEECHSPLIT2.0, which constrains the information flow of the speech component to be disentangled on the autoencoder input using efficient signal processing methods instead of bottleneck tuning. Evaluation results show that SPEECHSPLIT2.0 achieves comparable performance to SPEECHSPLIT in speech disentanglement and superior robustness to the bottleneck size variations. Our code is available at `https://github.com/biggytruck/SpeechSplit2`.

***Index Terms*—** Speech Disentanglement, Voice Conversion, Unsupervised Learning

1. INTRODUCTION

Human speech conveys a rich stream of information, which may contain many components entangled with each other such as content, rhythm, pitch, timbre, emotion, and accent, etc. However, most speech applications only focus on a narrow subset of the many components. For example, automatic speech recognition only focuses on the content; speaker recognition and emotion recognition focus on timbre and emotion respectively; voice conversion focuses mainly on timbre. Based on this, it is beneficial to disentangle the speech components of interest for a variety of speech applications, such as speech recognition [1], speech synthesis [2], emotion analysis [3], privacy protection [4], and voice conversion [5].

This paper focuses on speech disentanglement for voice conversion. Voice conversion aims to modify the voice characteristics of speech while keeping the linguistic content unchanged, which is an area where speech disentanglement has been frequently explored. The majority of the voice conversion systems focus on timbre conversion, where content and timbre disentanglement is the key to success. As an early attempt, VAE-VC [6] directly applies a variational autoencoder (VAE) for timbre disentanglement. Later, Chou et al. [7] and ACVAE-VC [8] disentangle timbre using an auxiliary speaker classifier. Inspired by image style transfer, StarGAN-VC [9], CycleGAN-VC [10] adapted StarGAN [11] and CycleGAN [12] respectively for voice conversion. AutoVC [5] disentangles speakers and content by directly tuning the bottleneck dimensions of a vanilla autoencoder. The following AutoVC-F0 [13] improves pitch disentanglement by conditioning on pitch contour. AutoPST [14] further disentangles rhythm using similarity-based time resampling. To improve the granularity in speech disentanglement, SPEECHSPLIT [15] disentangles speech into content, rhythm, pitch, and timbre using three encoders with carefully tuned bottlenecks. Although SPEECHSPLIT is effective for rhythm, pitch, and timbre conversion, it has two problems. First, bottleneck tuning is time- and resource-consuming. Second, re-tuning is required for a different dataset.

In this paper, we propose SPEECHSPLIT2.0, in which we apply efficient signal processing techniques to alleviate the laborious bottleneck tuning without modifying the network architecture of SPEECHSPLIT. We show that by processing the encoder inputs, we can control the information flowing into the model so that it can learn a disentangled representation for each component with reduced demands for bottleneck tuning. Experiments show that the disentangling performance of the proposed method remains robust under different bottleneck dimensions without tuning the bottleneck dimensions. It is worth mentioning that similar to SPEECHSPLIT, SPEECHSPLIT2.0 achieves speech disentanglement in an unsupervised manner by only training on reconstruction loss, which does not require parallel data or text transcriptions.

2. METHOD

2.1. SPEECHSPLIT

SPEECHSPLIT [15] is an autoencoder-based generative model that decomposes speech into four components: rhythm, content, pitch, and timbre. Three encoders, denoted as E_r, E_c, and E_f, are used to encode rhythm, content, and pitch information respectively and can be formulated as:

$$\mathbf{Z}_r = E_r(\mathbf{S}), \mathbf{Z}_c = E_c(R(\mathbf{S})), \mathbf{Z}_f = E_f(R(\mathbf{P})) \quad (1)$$

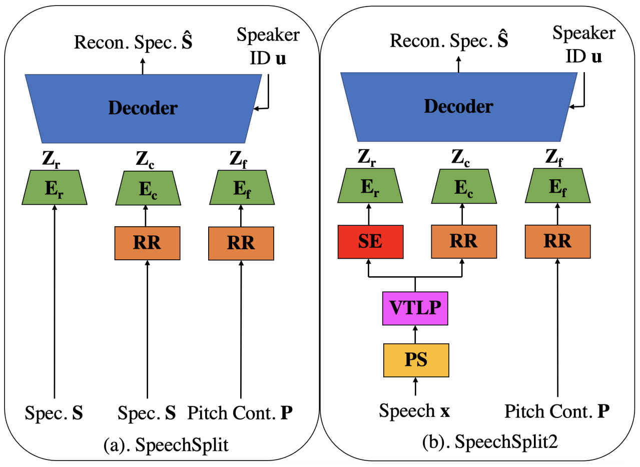

Fig. 1. (a). SPEECHSPLIT (b). SPEECHSPLIT2.0. 'RR' denotes random resampling, 'PS' denotes pitch smoother, 'VTLP' denotes vocal tract length perturbation, and 'SE' denotes extracting spectral envelope by ceptral liftering

where $\mathbf{S} = [\mathbf{s}_1, ..., \mathbf{s}_T]^T$ is the mel-spectrogram, $\mathbf{P} = [\mathbf{p}_1, ..., \mathbf{p}_T]^T$ is the one-hot representation of the quantized pitch contour that is normalized to have the same mean and variance with respect to each speaker, R denotes random resampling operation along time, and $\mathbf{Z}_r$, $\mathbf{Z}_c$, $\mathbf{Z}_f$ are the encoder outputs. The decoder D then takes all three encoded representations and a speaker embedding $\mathbf{u}$ (a one-hot embedding, or an embedding computed by an automatic speaker ID network) to generate an output spectrogram $\hat{\mathbf{S}}$:

$$\hat{\mathbf{S}} = D(\mathbf{Z}_r, \mathbf{Z}_c, \mathbf{Z}_f, \mathbf{u}) \tag{2}$$

The model is trained to reconstruct the input spectrogram:

$$\theta = \min_{\theta} \mathbb{E}[||\hat{\mathbf{S}} - \mathbf{S}||_2^2] \tag{3}$$

where θ denotes the model parameters. The model architecture is shown in Fig. 1(a). Since the random resampling operation corrupts the rhythm in the input to E_c and E_f, only E_r can observe the complete rhythm information. Thus, we can carefully tune the bottleneck dimension of E_r so that it only encodes the rhythm representation. Similarly, given that E_r only encodes the rhythm, we can carefully tune the dimension of E_c so that it only encodes the content information. Finally, the pitch contour still contains certain timbre information even after being randomly resampled, but it is assumed in [15] that normalization can remove the timbre, so E_f can have a relatively large bottleneck. Nevertheless, tuning the entire network is laborious, considerably limiting the model's applicability on different datasets and tasks.

2.2. SPEECHSPLIT2.0

The proposed algorithm, SPEECHSPLIT2.0, addresses the bottleneck tuning issue by using efficient signal processing algorithms at the front end of the encoders to corrupt the component to be disentangled, which significantly reduce the sensitivity of the model to the bottleneck dimensions. Fig. 1(b) shows the model architecture.

2.2.1. *Content Encoder Input*

To remove the pitch information from an utterance $\mathbf{x}$, we implement a *pitch smoother*, which first analyzes the signal using the WORLD vocoder [16]:

$$\mathbf{f}, \mathbf{a}, \mathbf{E} = A_{WORLD}(\mathbf{x}) \tag{4}$$

where $\mathbf{f}$, $\mathbf{a}$, and $\mathbf{E}$ denote F0, aperiodicity, and spectral envelope extracted by the WORLD analyzer A_{WORLD}. We then replace all the voiced frames in $\mathbf{f}$ with its own voiced mean to obtain a *normalized F0 contour* $\bar{\mathbf{f}}$ and re-synthesize the signal using the WORLD synthesizer S_{WORLD}:

$$\bar{\mathbf{x}} = S_{WORLD}(\bar{\mathbf{f}}, \mathbf{a}, \mathbf{E}) \tag{5}$$

where we denote $\bar{\mathbf{x}}$ as the *monotonic utterance* with a corresponding *monotonic spectrogram* $\bar{\mathbf{S}}$. Since all the elements in the F0 contour are replaced by its mean, the intonation dynamics along time are removed. Fig. 2(a) and 2(b) show the spectrogram and monotonic spectrogram for utterance *"Please call Stella."* Note that the tone movements along time in the monotonic spectrogram, e.g., the dropping tone of *"Stella"*, are completely flattened, indicating the pitch dynamics are discarded. Next, to corrupt the timbre information, we use Vocal Tract Length Perturbation (VTLP) [17], which modifies the timbre by warping the frequency and has been extensively used in various tasks [18, 19]. Formally,

$$\widetilde{\mathbf{x}} = H(\bar{\mathbf{x}}, \alpha) \tag{6}$$

where H denotes the VTLP operation, $\alpha \sim U(0.9, 1.1)$ is a warping factor, and $\widetilde{\mathbf{x}}$ is referred to as the *perturbed utterance* with a corresponding *perturbed spectrogram* $\widetilde{\mathbf{S}}$. We perturb each training utterance with a random α to prevent the content encoder from easily recovering the timbre information. Compared to Fig. 2(b), the perturbed spectrogram in Fig. 2(c) further shifts the formant frequencies down, resulting in a deeper voice. Finally, we randomly resample $\widetilde{\mathbf{S}}$ to corrupt the rhythm information to obtain the content encoder input $\mathbf{S}_c$ following the same procedure as in SPEECHSPLIT.

2.2.2. *Rhythm Encoder Input*

SPEECHSPLIT hypothesized a *"fill in the blank"* mechanism: the rhythm representation provides blanks corresponding to all the syllables and pauses in an utterance, and the decoder fills in these blanks with the respective content and pitch code. Based on this, we suggest that a good rhythm representation should (1) preserve very little about the content, pitch or timbre and (2) include some "indexing cues" to inform the

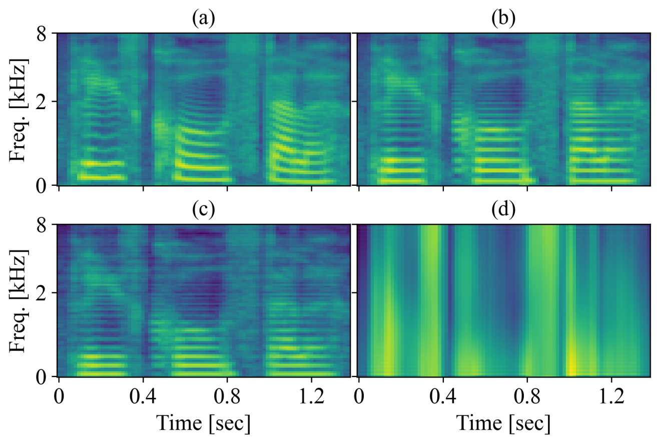

Fig. 2. (a). Spectrogram for utterance *"Please call Stella"* (b). Monotonic spectrogram (c). Perturbed spectrogram (d). Perturbed spectral envelope

decoder which part of the content and pitch representations should be filled in each blank. Thus, we propose to use a spectral envelope obtained by liftering the real cepstrum of the perturbed utterance $\widetilde{\mathbf{x}}$ as the input:

$$\widetilde{\mathbf{C}} = DFT^{-1}(\log(|\widetilde{\mathbf{Y}}|)) \quad (7)$$

$$\widetilde{\mathbf{C}}_l = \widetilde{\mathbf{C}}\mathbf{L} \quad (8)$$

$$\mathbf{S}_r = R(\exp(DFT(\widetilde{\mathbf{C}}_l))) \quad (9)$$

where $\widetilde{\mathbf{Y}} = [\widetilde{\mathbf{y}}_1, ..., \widetilde{\mathbf{y}}_T]^T$ is the spectrum of $\widetilde{\mathbf{x}}$, $\widetilde{\mathbf{C}} = [\mathbf{c}_1, ..., \mathbf{c}_T]^T$ is the real cepstrum, $\widetilde{\mathbf{C}}_l$ is the liftered real cepstrum and $\mathbf{L}$ is a diagonalized low-quefrency lifter:

$$\mathbf{L}_{ij} = \text{diag}\left(0.5u[n_c - i] + 0.5u[n_c - i - 1]\right), \quad (10)$$

where $u[\cdot]$ is the unit step function. If n_c is low, then $\mathbf{S}_r$ will contain very little content, pitch, and timbre information since it is obtained from the perturbed utterance and the liftering operation discards most of the fine details. However, the rhythm information is preserved, because the remaining spectral features are distinct for different phonemes, from which the encoder can extract the rhythm information. Fig. 2(d) shows such a spectral envelope with $n_c = 3$. As we can see, the phoneme variations are smoothed out, but the patterns are unique among different syllables and pauses.

2.2.3. *Pitch Encoder Input*

SpeechSplit assumes that by normalizing the pitch contour and randomly resampling it in time, we can discard all other information but still preserve the pitch. Thus, we do not need to further process the input to the pitch encoder.

However, SpeechSplit requires a pitch converter to restore the temporal mismatch between the source speech and the target pitch contour to perform pitch conversion. The pitch converter is a smaller variant of the model, which consists of only a rhythm and a pitch encoder. Since the rhythm encoder takes in the original spectrogram as its input, it is able to learn how to temporally realign the pitch contour. In contrast, SpeechSplit2.0 uses the spectral envelope as the rhythm input, which has insufficient information to realign the pitch contour. We fix the information gap by concatenating the perturbed spectrogram $\widetilde{\mathbf{S}} = [\widetilde{\mathbf{s}}_1, ..., \widetilde{\mathbf{s}}_T]^T$ and the one-hot quantized pitch contour $\mathbf{P} = [\mathbf{p}_1, ..., \mathbf{p}_T]^T$:

$$\mathbf{S}_p = R([\begin{bmatrix}\widetilde{\mathbf{s}}_1 \\ \mathbf{p}_1\end{bmatrix}, ..., \begin{bmatrix}\widetilde{\mathbf{s}}_T \\ \mathbf{p}_T\end{bmatrix}]^T) \quad (11)$$

where $\mathbf{S}_p$ is the pitch input in the pitch converter. By concatenating $\widetilde{\mathbf{S}}$ and $\mathbf{P}$, the perturbed spectrogram can bridge the information gap between the pitch and the rhythm encoder. During conversion, the decoder can first extract the alignment information from the coarse spectral envelope and the fine-grained spectrogram, and then adjust the pitch contour length accordingly.

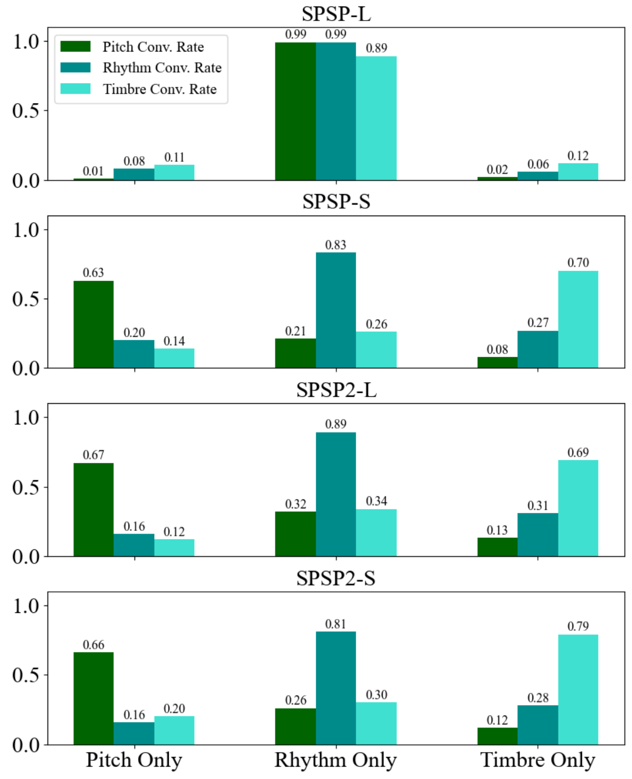

Fig. 3. Conversion rates of different models

3. EXPERIMENTS

We evaluate the results on four models: two small models, SPSP-S and SPSP2-S, which are SpeechSplit and SpeechSplit2.0 whose bottleneck configurations are carefully tuned to have the right dimension for the VCTK cor-

Model	Dim_R	Dim_C	Dim_F	DS_R	DS_C	DS_F
SPSP-L	32	32	32	1	1	1
SPSP-S	1	8	32	8	8	8
SPSP2-L	32	32	32	1	1	1
SPSP2-S	1	8	32	8	8	8

Table 1. Model configurations

pus [20]; two large models, SPSP-L and SPSP2-L, which have much wider bottlenecks by increasing the encoder dimensions and decreasing the downsampling factors. Since SPSP-L's bottleneck is too wide, the rhythm encoder can encode all four aspects. Consequently, the model will only reconstruct the signal that provides the rhythm representation and fail on all types of conversion. The SPSP2 encoders, by contrast, should be robust to different bottleneck settings so that SPSP2-L and SPSP2-S have comparable speech conversion ability as the carefully tuned SPSP-S. The model configurations are listed in Table 1, in which Dim_R, Dim_C, and Dim_F denotes the dimension for the rhythm, content and pitch encoder and DS_R, DS_C, and DS_F denotes the respective downsampling factor. We use $n_c = 3$ for the liftering operation. All other configurations are the same as in SPEECHSPLIT [15]. Audio samples are available at `https://biggytruck.github.io/spsp2-demo`.

3.1. Subjective Evaluations

We first evaluate the models' performance on *Amazon Mechanical Turk*. For each model, we apply pitch-only, rhythm-only, and timbre-only conversion on 20 utterance pairs that are perceptually distinct in the aspect of conversion and present the results to five subjects. Each subject is asked to listen to a source and a target reference in random order and determine which reference the converted speech is more similar to in terms of all three aspects. We then compute the *conversion rate* for each aspect, defined as the percentage of answers selecting the target reference, as shown in Fig. 3. For SPSP-L, all three conversion rates are very high for rhythm-only conversion but low for pitch-only and timbre-only conversion, indicating the rhythm encoder encodes all the information while the others only encode very trivial information. SPSP-S and both variants of SPSP2 successfully convert the aspect of interest without modifying the other attributes of the source utterance, which suggests that SPSP2 has the similar conversion capability as a carefully fine-tuned SPSP regardless of the bottleneck dimensions.

For all the converted utterances, we also ask the listeners to evaluate the naturalness with a 5-scale mean opinion score (MOS), as shown in Fig. 4. SPSP-L achieves the highest MOS since it essentially performs reconstruction of the input from the rhythm code. Compared to the fine-tuned SPSP-S, both variants of SPSP2 show some degree of degradation in naturalness. We hypothesize that the pitch smoother introduces

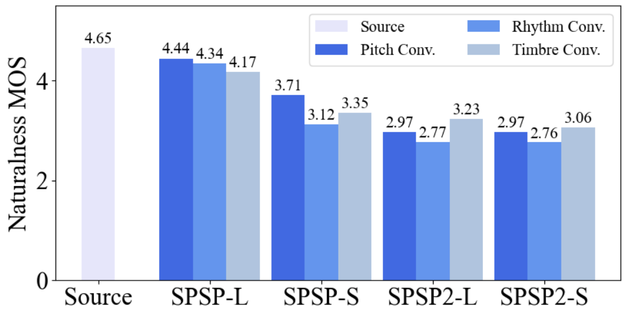

Fig. 4. Naturalness evaluation on different models

Model	Pitch Conv.	Rhythm Conv.	Timbre Conv.
SPSP-L	12.9%	14.4%	9.8%
SPSP-S	30.8%	46.3%	34.0%
SPSP2-L	37.8%	54.5%	39.2%
SPSP2-S	54.4%	62.6%	43.5%

Table 2. CER of converted utterances by model

artifacts in the smoothed audio mainly due to pitch tracking errors, which ultimately affect the naturalness.

3.2. Objective Evaluation

For objective evaluation, we are mainly interested in the intelligibility of the converted speech. Thus, we measure the character error rate (CER) of the transformed utterances using Google Cloud ASR, as shown in Table 2. Similar to the naturalness measurement, the artifacts result in higher CER in SPSP2 as compared to SPSP. It is worth noting that SPSP2-L has lower CER than SPSP2-S in all three types of conversion, suggesting a larger bottleneck preserves more content information. Given the evaluation results for naturalness and intelligibility, we may in the future improve SPSP2 by (1) replacing the pitch contour with better pitch representation and (2) designing more subtle bottleneck mechanisms to better preserve aspect-specific information with minimum content loss.

4. CONCLUSION

The proposed model, SPEECHSPLIT2.0, demonstrates equivalent capability in speech disentanglement as SPEECHSPLIT without the need to tune the bottleneck. We show that by modifying the inputs using efficient signal processing techniques, we can drive the encoders to learn a disentangled representation for different aspects of speech. In the future, we will focus on improving the naturalness and intelligibility of the generated speech and attempt to disentangle other components such as emotion and accent.

5. REFERENCES

[1] Wei-Ning Hsu, Hao Tang, and James Glass, "Unsupervised adaptation with interpretable disentangled representations for distant conversational speech recognition," in *Proc. Interspeech 2018*, 2018, pp. 1576–1580.

[2] Wei-Ning Hsu, Yu Zhang, Ron J. Weiss, Yu-An Chung, Yuxuan Wang, Yonghui Wu, and James Glass, "Disentangling correlated speaker and noise for speech synthesis via data augmentation and adversarial factorization," in *Proc. ICASSP*, 2019, pp. 5901–5905.

[3] K. Zhou, Berrak Sisman, and H. Li, "VAW-GAN for disentanglement and recomposition of emotional elements in speech," *ArXiv*, vol. abs/2011.02314, 2020.

[4] Dimitrios Stoidis and Andrea Cavallaro, "Protecting Gender and Identity with Disentangled Speech Representations," in *Proc. Interspeech 2021*, 2021, pp. 1699–1703.

[5] Kaizhi Qian, Yang Zhang, Shiyu Chang, Xuesong Yang, and Mark Hasegawa-Johnson, "AutoVC: Zero-shot voice style transfer with only autoencoder loss," in *Proceedings of the 36th International Conference on Machine Learning*, 2019, pp. 5210–5219.

[6] Chin-Cheng Hsu, Hsin-Te Hwang, Yi-Chiao Wu, Yu Tsao, and Hsin-Min Wang, "Voice conversion from non-parallel corpora using variational auto-encoder," in *Signal and Information Processing Association Annual Summit and Conference (APSIPA), 2016 Asia-Pacific*. IEEE, 2016, pp. 1–6.

[7] Ju-Chieh Chou, Cheng chieh Yeh, Hung yi Lee, and Lin-Shan Lee, "Multi-target voice conversion without parallel data by adversarially learning disentangled audio representations," in *INTERSPEECH*, 2018.

[8] Hirokazu Kameoka, Takuhiro Kaneko, Kou Tanaka, and Nobukatsu Hojo, "Acvae-vc: Non-parallel voice conversion with auxiliary classifier variational autoencoder," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 27, no. 9, pp. 1432–1443, 2019.

[9] Takuhiro Kaneko, Hirokazu Kameoka, Kou Tanaka, and Nobukatsu Hojo, "StarGAN-VC2: Rethinking conditional methods for StarGAN-based voice conversion," *Proc. Interspeech 2019*, pp. 679–683, 2019.

[10] Takuhiro Kaneko and H. Kameoka, "Cyclegan-vc: Non-parallel voice conversion using cycle-consistent adversarial networks," *2018 26th European Signal Processing Conference (EUSIPCO)*, pp. 2100–2104, 2018.

[11] Yunjey Choi, Min-Je Choi, Mun Su Kim, Jung-Woo Ha, Sunghun Kim, and Jaegul Choo, "Stargan: Unified generative adversarial networks for multi-domain image-to-image translation," *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8789–8797, 2018.

[12] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," *2017 IEEE International Conference on Computer Vision (ICCV)*, pp. 2242–2251, 2017.

[13] Kaizhi Qian, Zeyu Jin, Mark Hasegawa-Johnson, and Gautham J Mysore, "F0-consistent many-to-many non-parallel voice conversion via conditional autoencoder," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 6284–6288.

[14] Kaizhi Qian, Yang Zhang, Shiyu Chang, Jinjun Xiong, Chuang Gan, David Cox, and Mark A. Hasegawa-Johnson, "Global prosody style transfer without text transcriptions," in *ICML*, 2021.

[15] Kaizhi Qian, Yang Zhang, Shiyu Chang, Mark Hasegawa-Johnson, and David Cox, "Unsupervised speech decomposition via triple information bottleneck," in *Proceedings of the 37th International Conference on Machine Learning*, 2020, pp. 7836–7846.

[16] 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.

[17] Navdeep Jaitly and Geoffrey E. Hinton, "Vocal Tract Length Perturbation (VTLP) improves speech recognition," in *In International Conference on Machine Learning (ICML)*, 2013.

[18] Mengzhe Geng, Xurong Xie, Shansong Liu, Jianwei Yu, Shoukang Hu, Xunying Liu, and Helen Meng, "Investigation of Data Augmentation Techniques for Disordered Speech Recognition," in *Proc. Interspeech 2020*, 2020, pp. 696–700.

[19] Erica Cooper, Cheng-I Lai, Yusuke Yasuda, and Junichi Yamagishi, "Can Speaker Augmentation Improve Multi-Speaker End-to-End TTS?," in *Proc. Interspeech 2020*, 2020, pp. 3979–3983.

[20] Junichi Yamagishi, Christophe Veaux, and Kirsten MacDonald, "CSTR VCTK Corpus: English Multi-speaker Corpus for CSTR Voice Cloning Toolkit," *University of Edinburgh. The Centre for Speech Technology Research (CSTR)*, 2019.