SPSinger: Multi-Singer Singing Voice Synthesis with Short Reference Prompt

Junchuan Zhao*
School of Computing
National University of Singapore
Singapore
0009-0008-2616-6590

Chetwin Low*
School of Computing
National University of Singapore
Singapore
0009-0004-2367-5171

Ye Wang
School of Computing
National University of Singapore
Singapore
0000-0002-0123-1260

***Abstract*—Current singing voice synthesis systems often struggle in multi-singer scenarios due to limited training data that only includes a few singers. Existing zero-shot multi-singer singing voice synthesis systems are criticized for their reliance on global timbre embeddings from single reference audio, which fail to capture sufficient timbre details. This paper introduces SPSinger, a multi-singer singing voice synthesizer that generates singer-specific voices from brief reference audio (around *5 seconds*) without prior training on the singer's voice. SPSinger builds on the StableDiffusion framework by adding a global encoder to capture consistent timbre features from short reference prompts and an attention-based local encoder to capture detailed variations from long prompts, used only during training. To overcome the challenge of requiring long audio prompts during inference, we introduce the Latent Prompt Adaptation Model (LPAM), a Transformer-based module that derives timbre features from global embeddings. This approach eliminates the need for long reference prompts. Additionally, we propose a novel pitch shift algorithm that uses LPAM to predict the pitch shift values. Our experiments show that SPSinger achieves high-quality singing voice synthesis that preserves the identity of the target singer, even when using only short reference audio inputs in zero-shot scenarios.**

***Index Terms*—Singing voice synthesis, Multi-singer singing voice synthesis, Acoustic models, Diffusion models**

I. Introduction

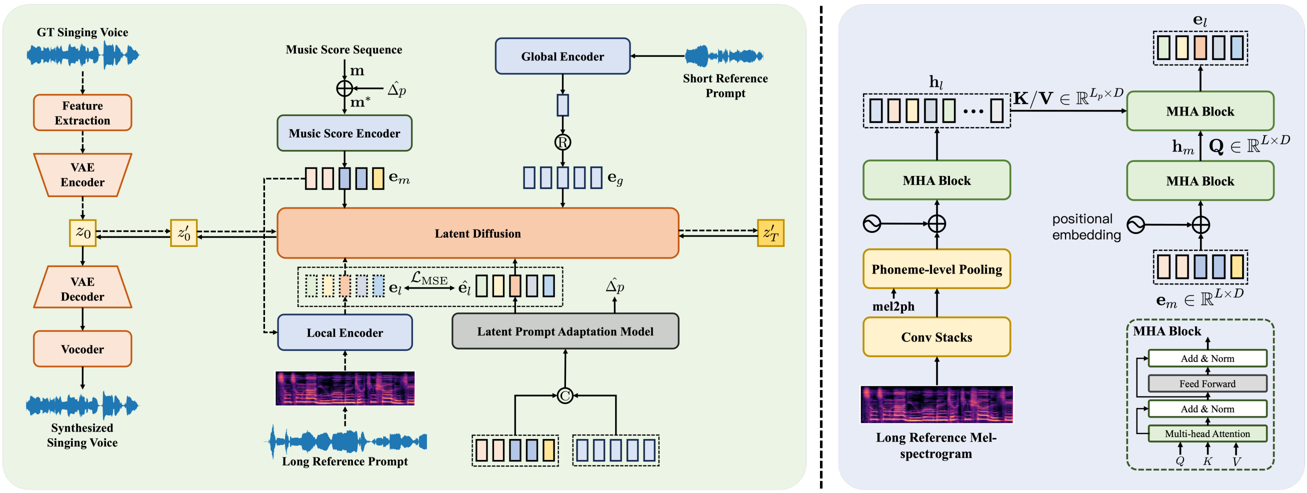
Singing Voice Synthesis (SVS) creates realistic artificial singing voices, widely used in music production and virtual singers. Recent advancements in deep learning have revolutionized SVS, enabling the generation of highly realistic and expressive vocals [1]–[7]. These systems excel in capturing pronunciation, pitch, and duration, but their potential can be expanded further by generating personalized voices, enhancing the versatility and expressiveness of virtual singers for diverse music production needs. This capability enhances the versatility and expressiveness of virtual singers and is invaluable in various music production scenarios.

Previous research in multi-singer singing voice synthesis (SVS) has often adapted methodologies from multi-speaker text-to-speech (TTS) systems, employing global singer features to encapsulate the vocal characteristics of a target singer from reference audio, subsequently feeding these into the synthesizer [2], [8]–[12]. However, the inherently expressive nature of singing—characterized by a broader range of pitches and timbres—presents a stark contrast to spoken language. This divergence renders a single, consistent timbre feature insufficient for effective multi-singer SVS systems [13]. To address these challenges, recent studies have shifted focus towards capturing the time-varying aspects of singers' performances. [14] introduced a local style token module, leveraging an attention mechanism to model time-varying features in relation to both text and pitch. [15], [16] proposed the use of multi-reference encoders to capture finer details and variations in the target timbre from multiple reference audios.

In this paper, we introduce SPSinger, a novel zero-shot multi-singer singing voice synthesis system that generates singing voices closely resembling target singers using only music scores and short reference audio prompts. Specifically, (1) we developed a multi-singer singing voice synthesis system based on StableDiffusion [17], incorporating both a global encoder and a local encoder. The global encoder captures timbre features from short reference audio prompts, while the local encoder processes longer reference prompts to extract more detailed timbre characteristics. (2) To enable high-quality synthesis with short prompts during inference, we proposed the Latent Prompt Adaptation Model (LPAM), a Transformer-based model that directly extracts local timbre features from music scores and global timbre hidden features, eliminating the need for a local encoder and long prompts. (3) Inspired by [15], we implemented a novel pitch shift method within the LPAM framework, aligning the pitch range of the input music score with the reference singer's pitch range for more accurate pitch representation. (4) To enhance zero-shot capability, we pre-train our TTS generative model on a large, diverse speech corpus, followed by fine-tuning on a smaller SVS dataset. This approach ensures SPSinger is exposed to a wide range of timbres, enabling it to effectively generalize to new singers.

We evaluated SPSinger against state-of-the-art (SOTA) multi-singer singing voice synthesis systems, employing both objective and subjective metrics to assess performance. Our results demonstrate that SPSinger not only surpasses baseline models in overall synthesis quality but also performs well in accurately capturing the target singer's style. Furthermore, ablation studies confirm the effectiveness of the key modules and methods introduced, highlighting their contributions to the

[*] These authors contributed equally to this research.



(a) Overall architecture of SPSinger

(b) Local Encoder

Fig. 1. Overall Architecture of SPSinger. $\mathbf{e}_m$, $\mathbf{e}_g$, and $\mathbf{e}_l$ correspondingly represent the music score hidden sequence, global feature hidden sequence, and local feature hidden sequence. $\hat{\mathbf{e}_l}$ represents the predicted local feature hidden. Concatenation and repeating operations are denoted by Ⓒ and Ⓡ, respectively. In this diagram, training is indicated by dashed lines, pitch shift is only performed during inference.

system's improved performance. Some samples are provided for listening[1].

II. Methodology

A. Overall Architecture

The architecture of SPSinger, illustrated in Fig. 1, takes the music score sequence $\mathbf{m}$, including pitch $\mathbf{m}^p$, lyrics $\mathbf{m}^l$, duration $\mathbf{m}^d$, and slur $\mathbf{m}^s$ sequences at the phoneme level, along with a short audio prompt $\tilde{\mathbf{a}}^s$ from the target singer. SPSinger consists of three main components: a music score encoder, an acoustic model, and a vocoder.

The music score encoder E^m, based on the Transformer architecture [18] from DeepSinger [4], converts the music score sequences $\mathbf{m}$ into hidden sequence $\mathbf{e}_m$. The acoustic model utilizes the StableDiffusion architecture [17], which has demonstrated success in generative modeling [17], [19], [20]. To control vocal characteristics precisely, we employ both a global encoder E^g and a local encoder E^l. The global encoder E^g captures consistent timbre features from short audio prompts, while the local encoder E^l extracts time-varying timbre features from longer audio prompts. To address the impracticality of using long prompts during inference, we introduce the Latent Prompt Adaptation Model (LPAM), a Transformer-based module. LPAM predicts the local feature hidden sequence $\mathbf{e}_l$ from the music score hidden sequence $\mathbf{e}_m$ and the global feature hidden sequence $\mathbf{e}_g$, obviating the need for long prompts. LPAM also predicts pitch shifts Δp to adjust the pitch sequence $\mathbf{m}^p$ during synthesis.

For the vocoder, we utilized a pre-trained HiFi-GAN model [21] designed for high-fidelity speech and singing voice synthesis[2], which is adopted by DiffSinger.

B. Global & Local Encoder

Building on prior research [9], [10], we used a pre-trained global encoder E^g to extract a fixed-size feature from a short audio prompt $\tilde{\mathbf{a}}^s$. This encoder, which includes the EPACA-TDNN [22], commonly used in multi-speaker TTS systems [23], [24], along with an additional linear layer, was adapted for our synthesis model. The consistent global timbre feature is replicated across the music score sequence, with only the linear layer trained while keeping the speaker encoder parameters fixed.

Inspired by [14], [15], the local encoder E^l captures dynamic timbre variations from long reference mel-spectrograms $\tilde{\mathbf{M}}^l$ in alignment with the music score sequence. It processes the input through convolutional stacks to extract contextual features, applies phoneme-level pooling using mel2ph to align frame-level mel-spectrograms with phoneme-level music score sequences, and utilizes multi-head self-attention (MHSA) with residual connections to capture global relationships within the music score and long prompt features. Multi-head cross-attention (MHCA) is then used to model the interdependencies between the music score hidden features $\mathbf{h}_m$ and the long prompt features $\mathbf{h}_l$.

We first compute the attention query $\mathbf{Q} = \mathbf{h}_m\mathbf{W}_q \in \mathbb{R}^{L\times D}$, key $\mathbf{K} = \mathbf{h}_l\mathbf{W}_k \in \mathbb{R}^{L_p\times D}$, and value $\mathbf{V} = \mathbf{h}_l\mathbf{W}_v \in \mathbb{R}^{L_p\times D}$. The local feature hidden sequence e_l is then calculated as:

$$\mathbf{e}_l = \text{softmax}(\frac{\mathbf{Q}\cdot\mathbf{K}^\top}{\sqrt{D}})\cdot\mathbf{V}. \tag{1}$$

C. Latent Prompt Adaptation Model (LPAM)

The local encoder requires a long reference audio prompt input, which is impractical for users. To address this, we introduce a Transformer-decoder-based Latent Prompt Adaptation Model (LPAM) that generates the local feature hidden sequence $\mathbf{e}_f$ without the need for a long reference prompt

[1] https://danny-nus.github.io/SPSinger/

[2] https://github.com/MoonInTheRiver/DiffSinger/releases/download/pretrain-model/0109_hifigan_bigpopcs_hop128.zip

$\tilde{\mathbf{a}}^l$. The LPAM module takes as input the concatenated global feature hidden sequence $\mathbf{e}_g$ and music score hidden sequence $\mathbf{e}_m$. A 1D convolutional layer projects this input to the dimension of $\mathbf{e}_g$. The resulting output is then fed into four Transformer-decoder layers to autoregressively infer $\hat{\mathbf{e}_l}$, which can be formulated as:

$$p(\hat{\mathbf{e}}_l \mid \mathbf{e}_g, \mathbf{e}_m; \theta) = \prod_{i=0}^{L-1} p(\hat{\mathbf{e}_{l,i}} \mid \hat{\mathbf{e}_{l,<i}}, \mathbf{e}_g, \mathbf{e}_m; \theta). \qquad (2)$$

1) LPAM for pitch shift: Previous research on multi-singer synthesis [15] and voice conversion [25], [26] highlights the challenge of mismatched pitch ranges between target singers and music scores. While naive pitch shift algorithms (NPS) [15], [25] adjust the fundamental frequency ($f0$) based on pitch differences, they often fail to capture the full vocal range due to the limitations of short reference prompts. To address this, we propose to utilize the LPAM module to predict the necessary pitch shift, by adding a classifier head is after the Transformer-decoder layers.

D. Training & Inference Strategy

The training of SPSinger involves two stages: (1) training without the LPAM module using speech and singing voice datasets; (2) training the LPAM module with the paired dataset inferred from the first stage.

1) Training of SPSinger w/o LPAM: In the first stage, we use StableDiffusion [17] as our backbone and follow the training process of AudioLDM [20]. We initially train the VAE encoder $\mathcal{E}$ and decoder $\mathcal{D}$ with a combined objective of reconstruction loss, adversarial loss, and Gaussian constraint loss, as described in [20]. We then train the latent diffusion model (LDM) with the objective [17], [20]:

$$\mathcal{L}_{LDM}(\theta) = \mathbb{E}_{\mathcal{E}(\mathbf{M}), \mathbf{e}, \epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I}), t}\left[\|\epsilon - \epsilon_\theta(\mathbf{z}_t, t, \mathbf{e})\|_2^2\right], \qquad (3)$$

where $\mathbf{e}$ is the hidden sequence, which is the concatenation of $\mathbf{e}_m$, $\mathbf{e}_g$ and $\mathbf{e}_l$; t is the time step. To enhance zero-shot performance, we pre-train the model on speech data [27] using a phoneme encoder from FastSpeech 2 [28] instead of the music score encoder, while other modules remain unchanged. We then fine-tune the model on a singing voice dataset [29].

2) Training of LPAM: In the second training stage, we infer hidden sequences $\mathbf{e}_m$, $\mathbf{e}_g$, and $\mathbf{e}_l$ using the pre-trained StableDiffusion model from the first stage. We then train LPAM autoregressively by minimizing the MSE loss $\mathcal{L}_{\text{MSE}}$ between predicted $\hat{\mathbf{e}}_l$ and target $\mathbf{e}_l$, excluding the classifier head. The MSE loss is defined as:

$$\mathcal{L}_{\text{MSE}}(\theta) = \sum_{i=0}^{L-1} \|\hat{\mathbf{e}}_{l,i} - \mathbf{e}_{l,i}\|_2^2 . \qquad (4)$$

We then train the LPAM classifier head while keeping other layers fixed. Specifically, we perturb the pitch sequence of the input musics score sequence by $\delta \in [-6, +6]$ to obtain the perturbed hidden sequence $\mathbf{e}'_m$. The pitch shift is computed as $\Delta p = p' - p_l \in \{0, \pm 1, \pm 2, \ldots, \pm 15\}$, where p_l is the median pitch of the long reference prompt and p' is the median pitch of the perturbed pitch sequence. Using $\mathbf{e}'_m$ and $\mathbf{e}_g$ as inputs and Δp as the target, we train the classifier head with class-distance weighted cross-entropy (DWCE) loss $\mathcal{L}_{\text{DWCE}}$, which enhances accuracy by weighting the loss based on the proximity of predicted pitch values to the target. $\mathcal{L}_{\text{DWCE}}$ is formulated as:

$$\mathcal{L}_{\text{DWCE}} = \left(1 + \frac{\sum_{i \in \mathbf{c}} p_i \cdot \left|i - \hat{\Delta p}\right|}{C - 1}\right) \cdot \text{CE}(\Delta p, \hat{\Delta p}), \qquad (5)$$

where $\hat{\Delta p}$ and Δp denote the predicted and target pitch shift values respectively; $\mathbf{c} = [-15, -14, \ldots, 15]$ represents the vector of class values; p_i is the softmax probability associated with the i-th class.

3) Inference of SPSinger: During inference, the music score and a short reference prompt are input to the StableDiffusion model to infer $\mathbf{e}_m$ and $\mathbf{e}_g$. The LPAM module then predicts the pitch shift $\hat{\Delta p}$, updating the pitch sequence to $\mathbf{m}^{p*} = \mathbf{m}^p + \hat{\Delta p}$ and deriving $\mathbf{e}_m{}^*$. Using $\mathbf{e}_m{}^*$ and $\mathbf{e}_g$, the local feature $\mathbf{e}_l$ is obtained with the LPAM module, excluding the classifier head. The LDM condition $\mathbf{e}$ is formed by concatenating $\mathbf{e}_m{}^*$, $\mathbf{e}_g$, and $\mathbf{e}_l$, which is then processed by the decoder and vocoder to generate the singing voice.

III. Experimental Setups

A. Datasets

The experiments are conducted on two Mandarin singing corpora, M4Singer [29] and OpenSinger [30], and a Mandarin speech corpus, Magicdata [27]. We pretrain on Magicdata, which provides 180 hours of speech from 663 speakers, with duration sequences extracted via Kaldi [31] and reference audio prepared similarly to M4Singer. Fine-tuning is performed on M4Singer, which includes 700 Chinese pop songs by 20 vocalists with both short (5-second) and long (180-second) prompts. For zero-shot evaluation, we leverage 10 male and 10 female singers from the OpenSinger dataset, integrating M4Singer's music score sequences to address the absence of annotations in OpenSinger. All audio is down-sampled to 24 kHz with 16-bit quantization.

B. Implementation Details

Our model, implemented in PyTorch and PyTorch Lightning, was trained on NVIDIA RTX A5000 GPUs. We adopted the UNet and VAE architectures from AudioLDM [20]. The linear layer in the global encoder has size (192, 192). The Conv1d layers in local encoder are (80, 256) with a kernel size of 5, and the Multi-Head Attention blocks have a hidden size of 256 and 8 heads [18]. The LPAM module includes a 1D convolution layer with dimensions (448, 256) and kernel size 3, and Transformer-decoder layers with a hidden size of 256, 8 attention heads, and a feed-forward dimension of 512. Training involved 50k steps for the VAE with a batch size of 48, 200k steps of pre-training and 50k steps of fine-tuning for the LDM, and 50k steps for the LPAM module, utilizing the AdamW optimizer ($\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 10^{-9}$).

TABLE I
OBJECTIVE AND SUBJECTIVE EVALUATION FOR SEEN SINGER SVS. LE DENOTES THE LOCAL ENCODER; MOS-N DENOTES THE NATURALNESS AND MOS-SQ DENOTES THE SOUND QUALITY.

Model	MCD (dB) (↓)	F0-RMSE (logHz) (↓)	COS (↑)	MOS-N (↑)	MOS-SQ (↑)	SMOS (↑)
DiffSinger	4.82	0.0430	0.742	3.69±0.21	3.84±0.15	3.91±0.13
MR-SVS	5.83	0.2216	0.547	2.68±0.20	2.78±0.17	3.54±0.20
SPSinger - w/o Magicdata	4.62	0.0222	0.843	3.96±0.22	4.32±0.18	4.20±0.15
SPSinger - w/o LE & LPAM	4.78	0.0391	0.745	3.78±0.19	4.26±0.20	4.16±0.17
SPSinger	**4.76**	**0.0312**	**0.855**	**3.82±0.16**	**4.25±0.22**	**4.32±0.12**
Ground Truth	-	-	0.920	4.27±0.19	4.60±0.18	4.51±0.14

TABLE II
OBJECTIVE AND SUBJECTIVE EVALUATION FOR UNSEEN SINGER SVS.

Model	COS (↑)	MOS-N (↑)	MOS-SQ (↑)	SMOS (↑)
DiffSinger	0.629	3.84±0.17	3.79±0.16	3.28±0.19
MR-SVS	0.435	2.45±0.13	2.90±0.12	2.55±0.14
SPSinger - w/o Magicdata	0.735	4.06±0.18	4.25±0.13	3.85±0.18
SPSinger - w/o LE & LPAM	0.790	3.89±0.13	4.19±0.18	4.06±0.14
SPSinger	**0.837**	**3.95±0.15**	**4.25±0.22**	**4.18±0.15**
Ground Truth	0.907	4.15±0.11	4.37±0.15	4.40±0.12

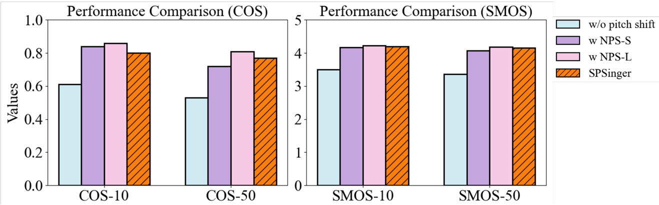


Fig. 2. Comparative analysis of pitch shift algorithms across models, evaluated using singer similarity metrics with music score sequences of lengths 10 and 50.

C. Evaluation Methods

To evaluate SPSinger, we compared it with multi-singer DiffSinger [3] and MR-SVS [15], all trained on the M4Singer dataset and using the HiFiGAN vocoder, consistent with our approach. We assessed synthesis quality using Mel Cepstral Distortion (MCD) [32] and Root Mean-Squared Error of Fundamental Frequency (F0-RMSE) [33], and multi-singer controllability with Singer Cosine Similarity (COS) [16], [25], [34], computed via the WavLM model [35][3] fine-tuned for speaker verification. For subjective evaluation, we utilized mean opinion scores (MOS) covering naturalness (MOS-N), sound quality (MOS-Q), and timbre similarity (SMOS). The assessment involved 20 highly experienced participants, each with extensive backgrounds in choir and pop singing, who rated audio samples from various systems on a scale of 1 to 5, with 5 indicating the highest quality.

IV. EXPERIMENTAL RESULTS

Tables I and II present a comparative performance analysis of SPSinger against the SOTA multi-singer SVS systems DiffSinger and MR-SVS, under both seen and unseen (zero-shot) conditions. SPSinger consistently outperforms the baselines in both scenarios across objective and subjective metrics. The advantage of SPSinger is even more pronounced in the zero-shot setting, as shown in Table II.

In our ablation study, we observed a drop in COS and SMOS performance when SPSinger was not pretrained on the Magicdata dataset, especially in zero-shot scenarios. This underscores the importance of diverse speaker data for improving zero-shot performance. Despite training SPSinger exclusively on the M4Singer dataset—just as with DiffSinger and MR-SVS—SPSinger consistently outperforms both baselines, demonstrating its robustness and effectiveness. Although omitting Magicdata pre-training slightly enhances general synthesis metrics, suggesting that a gap still exists between speech and singing data. Moreover, removing the local encoder and LPAM module led to reduced performance across both objective and subjective metrics, highlighting the critical role of these components.

We assess various pitch shift algorithms by comparing SPSinger with: (1) NPS-S, which applies naive pitch shift (NPS) to short audio prompts as discussed in Section II-C; (2) NPS-L, which applies NPS to long audio prompts; and (3) a no pitch shift condition. As shown in Figure 2, incorporating pitch shift generally leads to performance degradation. Although NPS-S yields similar results to SPSinger with short inputs, the performance gap increases with longer sequences due to the limited pitch range of short prompts. While SPSinger is slightly less effective than NPS-L, it provides greater practicality by eliminating the need for long reference prompts.

V. CONCLUSION

This paper presents SPSinger, a novel multi-singer voice synthesis system that operates effectively with short reference prompts. Our approach advances timbre feature extraction through an attention-based local encoder that captures nuanced variations in long prompts and a latent prompt adaptation model (LPAM) that derives variation features from global features and music scores, thus obviating the need for long prompts during inference. Unlike traditional pitch shift methods, SPSinger predicts pitch shift directly from the music score and short prompt using the LPAM module. Experimental results validate SPSinger's superior performance in multi-singer control and our ablation studies underscore the critical role of each component in enhancing the system's effectiveness.

[3] https://huggingface.co/microsoft/wavlm-base-plus-sv

References

[1] J. He, J. Liu, Z. Ye, R. Huang, C. Cui, H. Liu, and Z. Zhao, "RMSSinger: Realistic-music-score based singing voice synthesis," in *Findings of the Association for Computational Linguistics: ACL 2023*, A. Rogers, J. Boyd-Graber, and N. Okazaki, Eds. Toronto, Canada: Association for Computational Linguistics, Jul. 2023, pp. 236–248. [Online]. Available: https://aclanthology.org/2023.findings-acl.16

[2] P. Chandna, M. Blaauw, J. Bonada, and E. Gómez, "Wgansing: A multi-voice singing voice synthesizer based on the wasserstein-gan," in *2019 27th European signal processing conference (EUSIPCO)*. IEEE, 2019, pp. 1–5.

[3] J. Liu, C. Li, Y. Ren, F. Chen, and Z. Zhao, "Diffsinger: Singing voice synthesis via shallow diffusion mechanism," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 36, no. 10, 2022, pp. 11 020–11 028.

[4] Y. Ren, X. Tan, T. Qin, J. Luan, Z. Zhao, and T.-Y. Liu, "Deepsinger: Singing voice synthesis with data mined from the web," in *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2020, pp. 1979–1989.

[5] W. Chunhui, C. Zeng, and X. He, "Xiaoicesing 2: A High-Fidelity Singing Voice Synthesizer Based on Generative Adversarial Network," in *Proc. INTERSPEECH 2023*, 2023, pp. 5401–5405.

[6] K. Shen, Z. Ju, X. Tan, E. Liu, Y. Leng, L. He, T. Qin, sheng zhao, and J. Bian, "Naturalspeech 2: Latent diffusion models are natural and zero-shot speech and singing synthesizers," in *The Twelfth International Conference on Learning Representations*, 2024. [Online]. Available: https://openreview.net/forum?id=Rc7dAwVL3v

[7] J. Zhao, L. Q. H. Chetwin, and Y. Wang, "Sintechsvs: A singing technique controllable singing voice synthesis system," *IEEE ACM Trans. Audio Speech Lang. Process.*, vol. 32, pp. 2641–2653, 2024. [Online]. Available: https://doi.org/10.1109/TASLP.2024.3394769

[8] A. Gibiansky, S. Arik, G. Diamos, J. Miller, K. Peng, W. Ping, J. Raiman, and Y. Zhou, "Deep voice 2: Multi-speaker neural text-to-speech," *Advances in neural information processing systems*, vol. 30, 2017.

[9] L. Zhang, C. Yu, H. Lu, C. Weng, C. Zhang, Y. Wu, X. Xie, Z. Li, and D. Yu, "DurIAN-SC: Duration Informed Attention Network Based Singing Voice Conversion System," in *Proc. Interspeech 2020*, 2020, pp. 1231–1235.

[10] X. Wang, C. Zeng, J. Chen, and C. Wang, "Crosssinger: A cross-lingual multi-singer high-fidelity singing voice synthesizer trained on monolingual singers," in *2023 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*. IEEE, 2023, pp. 1–6.

[11] Z. Zhang, Y. Zheng, X. Li, and L. Lu, "Wesinger 2: Fully parallel singing voice synthesis via multi-singer conditional adversarial training," in *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2023, pp. 1–5.

[12] J. Lee, H.-S. Choi, J. Koo, and K. Lee, "Disentangling timbre and singing style with multi-singer singing synthesis system," in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 7224–7228.

[13] M. Blaauw, J. Bonada, and R. Daido, "Data efficient voice cloning for neural singing synthesis," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 6840–6844.

[14] J. Lee, H.-S. Choi, and K. Lee, "Expressive singing synthesis using local style token and dual-path pitch encoder," *arXiv preprint arXiv:2204.03249*, 2022.

[15] S. Wang, J. Liu, Y. Ren, Z. Wang, C. Xu, and Z. Zhao, "Mr-svs: Singing voice synthesis with multi-reference encoder," *arXiv preprint arXiv:2201.03864*, 2022.

[16] Z. Jiang, J. Liu, Y. Ren, J. He, Z. Ye, S. Ji, Q. Yang, C. Zhang, P. Wei, C. Wang, X. Yin, Z. MA, and Z. Zhao, "Boosting prompting mechanisms for zero-shot speech synthesis," in *The Twelfth International Conference on Learning Representations*, 2024. [Online]. Available: https://openreview.net/forum?id=mvMI3N4AvD

[17] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, "High-resolution image synthesis with latent diffusion models," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022, New Orleans, LA, USA, June 18-24, 2022*. IEEE, 2022, pp. 10 674–10 685. [Online]. Available: https://doi.org/10.1109/CVPR52688.2022.01042

[18] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.

[19] L. Zhang, A. Rao, and M. Agrawala, "Adding conditional control to text-to-image diffusion models," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 3836–3847.

[20] H. Liu, Z. Chen, Y. Yuan, X. Mei, X. Liu, D. P. Mandic, W. Wang, and M. D. Plumbley, "Audioldm: Text-to-audio generation with latent diffusion models," in *International Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA*, ser. Proceedings of Machine Learning Research, A. Krause, E. Brunskill, K. Cho, B. Engelhardt, S. Sabato, and J. Scarlett, Eds., vol. 202. PMLR, 2023, pp. 21 450–21 474. [Online]. Available: https://proceedings.mlr.press/v202/liu23f.html

[21] J. Kong, J. Kim, and J. Bae, "Hifi-gan: Generative adversarial networks for efficient and high fidelity speech synthesis," *Advances in neural information processing systems*, vol. 33, pp. 17 022–17 033, 2020.

[22] B. Desplanques, J. Thienpondt, and K. Demuynck, "ECAPA-TDNN: Emphasized Channel Attention, Propagation and Aggregation in TDNN Based Speaker Verification," in *Proc. Interspeech 2020*, 2020, pp. 3830–3834.

[23] J. Xue, Y. Deng, Y. Han, Y. Li, J. Sun, and J. Liang, "Ecapa-tdnn for multi-speaker text-to-speech synthesis," in *2022 13th International Symposium on Chinese Spoken Language Processing (ISCSLP)*. IEEE, 2022, pp. 230–234.

[24] Y. Shi, H. Bu, X. Xu, S. Zhang, and M. Li, "AISHELL-3: A Multi-Speaker Mandarin TTS Corpus," in *Proc. Interspeech 2021*, 2021, pp. 2756–2760.

[25] J.-T. Wu, J.-Y. Wang, J.-S. R. Jang, and L. Su, "A unified model for zero-shot singing voice conversion and synthesis," in *Ismir 2022 Hybrid Conference*, 2022.

[26] Y.-J. Luo, C.-C. Hsu, K. Agres, and D. Herremans, "Singing voice conversion with disentangled representations of singer and vocal technique using variational autoencoders," in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 3277–3281.

[27] "Magic Data Technology Co., Ltd." http://www.imagicdatatech.com/index.php/home/dataopensource/data_info/id/101, 05 2019.

[28] Y. Ren, C. Hu, X. Tan, T. Qin, S. Zhao, Z. Zhao, and T. Liu, "Fastspeech 2: Fast and high-quality end-to-end text to speech," in *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021. [Online]. Available: https://openreview.net/forum?id=piLPYqxtWuA

[29] L. Zhang, R. Li, S. Wang, L. Deng, J. Liu, Y. Ren, J. He, R. Huang, J. Zhu, X. Chen *et al.*, "M4singer: A multi-style, multi-singer and musical score provided mandarin singing corpus," *Advances in Neural Information Processing Systems*, vol. 35, pp. 6914–6926, 2022.

[30] R. Huang, F. Chen, Y. Ren, J. Liu, C. Cui, and Z. Zhao, "Multi-singer: Fast multi-singer singing voice vocoder with A large-scale corpus," in *MM '21: ACM Multimedia Conference, Virtual Event, China, October 20 - 24, 2021*, H. T. Shen, Y. Zhuang, J. R. Smith, Y. Yang, P. César, F. Metze, and B. Prabhakaran, Eds. ACM, 2021, pp. 3945–3954. [Online]. Available: https://doi.org/10.1145/3474085.3475437

[31] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, Y. Qian, P. Schwarz *et al.*, "The kaldi speech recognition toolkit," in *IEEE 2011 workshop on automatic speech recognition and understanding*. IEEE Signal Processing Society, 2011.

[32] T. Toda, A. W. Black, and K. Tokuda, "Voice conversion based on maximum-likelihood estimation of spectral parameter trajectory," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 15, no. 8, pp. 2222–2235, 2007.

[33] M. Nishimura, K. Hashimoto, K. Oura, Y. Nankaku, and K. Tokuda, "Singing voice synthesis based on deep neural networks." in *Interspeech*, 2016, pp. 2478–2482.

[34] S. Zhou, X. Li, Z. Wu, Y. Shan, and H. Meng, "Enhancing the vocal range of single-speaker singing voice synthesis with melody-unsupervised pre-training," in *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2023, pp. 1–5.

[35] S. Chen, C. Wang, Z. Chen, Y. Wu, S. Liu, Z. Chen, J. Li, N. Kanda, T. Yoshioka, X. Xiao *et al.*, "Wavlm: Large-scale self-supervised pre-training for full stack speech processing," *IEEE Journal of Selected Topics in Signal Processing*, vol. 16, no. 6, pp. 1505–1518, 2022.