DIFFERENTIABLE DIGITAL SIGNAL PROCESSING MIXTURE MODEL FOR SYNTHESIS PARAMETER EXTRACTION FROM MIXTURE OF HARMONIC SOUNDS

Masaya Kawamura[1], Tomohiko Nakamura[1], Daichi Kitamura[2],
Hiroshi Saruwatari[1], Yu Takahashi[3], Kazunobu Kondo[3]

[1] The University of Tokyo, Tokyo, Japan
[2] National Institute of Technology, Kagawa College, Kagawa, Japan
[3] Yamaha Corporation, Shizuoka, Japan

ABSTRACT

A differentiable digital signal processing (DDSP) autoencoder is a musical sound synthesizer that combines a deep neural network (DNN) and spectral modeling synthesis. It allows us to flexibly edit sounds by changing the fundamental frequency, timbre feature, and loudness (synthesis parameters) extracted from an input sound. However, it is designed for a monophonic harmonic sound and cannot handle mixtures of harmonic sounds. In this paper, we propose a model (DDSP mixture model) that represents a mixture as the sum of the outputs of multiple pretrained DDSP autoencoders. By fitting the output of the proposed model to the observed mixture, we can directly estimate the synthesis parameters of each source. Through synthesis parameter extraction experiments, we show that the proposed method has high and stable performance compared with a straightforward method that applies the DDSP autoencoder to the signals separated by an audio source separation method.

Index Terms— Differentiable digital signal processing, music sound synthesis, deep learning, music audio editing

1. INTRODUCTION

Musical instrument sound synthesizers based on deep neural networks (DNNs) have been actively studied [1–6]. Such synthesizers can generate high-quality musical instrument sounds and also allow us to edit the sounds by appropriately changing their inputs and parameters. Recently, an approach called differentiable digital signal processing (DDSP) has gathered attention [7]. This approach utilizes classical signal processing components for a DNN-based sound synthesizer and enables us to train the synthesizer in an end-to-end manner. The DDSP autoencoder is one of the state-of-the-art DNN-based synthesizers categorized in this approach [7]. It reconstructs an input audio signal by a classical signal processing technique called spectral modeling synthesis (SMS) [8], which separately models harmonic and inharmonic parts of the signal. The control signals of the SMS are computed by a DNN. As a latent representation, this DNN transforms the input into three interpretable parameters corresponding to pitch, timbre, and loudness: fundamental frequency (F_0), timbre feature, and loudness. We call these parameters the synthesis parameters. By appropriately changing the synthesis parameters, we can flexibly edit the pitch, timbre, and loudness of the input signal.

However, the DDSP autoencoder cannot be applied directly to a mixture of harmonic sounds because it is designed only for a monophonic harmonic signal. One straightforward method to solve this problem is to separate the mixture into individual sources and apply the DDSP autoencoder to each of them. Despite the recent progress

of DNN-based audio source separation methods [9], it is difficult to always obtain separated signals indistinguishable from the clean ones. Since the DDSP autoencoder is trained with clean musical instrument audio signals, the artifacts and interferer signals included in the separated signals can cause the performance degradation of the DDSP autoencoder. In fact, the separated signals obtained with a state-of-the-art score-informed source separation method partly included the interferer signals, and the signals reconstructed by the DDSP autoencoder were considerably different from the target signals, which we will show later in section 4. Furthermore, in practice, we often need to edit mixtures of sounds made by the same instruments. Although the separation of such mixtures has been studied recently [10, 11], it is more difficult than the separation of sounds made by the different instruments.

In this paper, we propose a method for directly estimating the synthesis parameters of the individual sources from a mixture audio signal. We take not the *separation-and-analysis* approach described above but an *analysis-by-synthesis* approach. That is, we construct a model that describes a generative process of the mixture, and we estimate the synthesis parameters by fitting the mixture generated with the model to an observed mixture. By removing the synthesis parameter extraction part from the input in the DDSP autoencoder, we can use it to synthesize the source from the synthesis parameters. The proposed model represents the mixture as the sum of the outputs of the source synthesizers driven with their own synthesis parameters. We call this model the *DDSP mixture model*. Using the pretrained source synthesizers, we fit the output of the proposed model to the observed mixture by a gradient descent algorithm.

Owing to the interpretability of the synthesis parameters, we can use musical score information for the initialization of the synthesis parameters. Recent source separation literature has shown that the use of score information improves the separation performance [12–15], which may be true for our problem. We experimentally examine the effect of the score-based initialization of the synthesis parameters.

2. RELATED WORKS

2.1. DDSP Autoencoder

The DDSP autoencoder consists of an encoder, a decoder, and an SMS module. Fig. 1 shows a schematic illustration of the architecture of the DDSP autoencoder. The encoder extracts the synthesis parameters of T frames from an input signal $\boldsymbol{x} \in \mathbb{R}^N$ with a length of N. Let $t = 1, \ldots, T$ be the frame index. The F_0 at frame t, denoted by $f_t \geq 0$, is computed by a pretrained CREPE model [16],
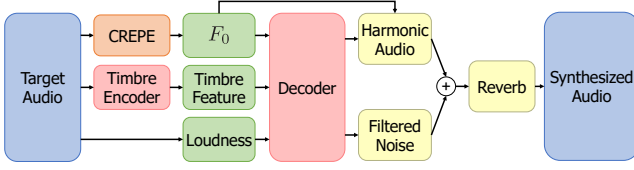
Fig. 1. Architecture of DDSP autoencoder. Red blocks consist of DNNs to be trained and "CREPE" block is pretrained DNN.

which is one of the state-of-the-art F_0 estimators. The timbre feature of size D, $z_t \in \mathbb{R}^D$, is calculated by a timbre encoder, which computes mel-frequency cepstral coefficients (MFCCs) from the input signal and feeds them into a DNN. The loudness $l_t \in \mathbb{R}$ is computed by applying A-weighting to the power spectrum of the input signal and taking its logarithm. The decoder is a DNN that transforms the synthesis parameters into the control signals of the SMS module in frames. See [7] for the detailed architecture of the decoder and timbre encoder.

The SMS module separately generates harmonic and inharmonic signals and adds them together. The harmonic signal is generated as the sum of sinusoids with piecewise linear frequencies and amplitudes. These frequencies are computed by linearly interpolating f_t and its harmonics up to the signal time resolution. The amplitudes are the linearly interpolated versions of framewise amplitudes outputted by the decoder. To generate the inharmonic signal, the decoder outputs the magnitude frequency responses of a time-varying finite impulse response filter in frames. We apply a Hann window to the discrete Fourier transforms of these responses and convolve them with a white noise signal in the frequency domain. A reverb module implemented by a convolutional layer is optionally applied to the sum of the harmonic and inharmonic signals and outputs a synthesized signal $\hat{\boldsymbol{x}} \in \mathbb{R}^N$.

The timbre encoder, decoder, and reverb module are trained so that the multiscale spectral loss between $\boldsymbol{x}$ and $\hat{\boldsymbol{x}}$ is minimized [7]. This loss uses short-time Fourier transforms (STFTs) of the two inputs with frames of I different lengths. It is defined as

$$L(\boldsymbol{x}, \hat{\boldsymbol{x}}) = \sum_{i=1}^{I} L_i(\boldsymbol{x}, \hat{\boldsymbol{x}}), \tag{1}$$

$$L_i(\boldsymbol{x}, \hat{\boldsymbol{x}}) = \|\mathcal{F}_i\boldsymbol{x} - \mathcal{F}_i\hat{\boldsymbol{x}}\|_1 + \|\log \mathcal{F}_i\boldsymbol{x} - \log \mathcal{F}_i\hat{\boldsymbol{x}}\|_1, \tag{2}$$

where $\mathcal{F}_i$ returns the magnitude STFT of the signal with the ith frame length.

2.2. Audio Source Separation

Most conventional music audio editing systems use audio source separation methods as preprocessors to extract the target sources from a mixture [17–20]. This approach can be applied to our problem. The recent literature has shown that the use of musical scores enhances the separation performance [12–15]. The method using nonnegative matrix factorization (NMF) presented in [15] is one of the state-of-the-art score-informed source separation methods. This method trains NMF bases with isolated instrument sounds in advance and separates the input mixture while aligning the performance and score information. Although the DNN-based methods show superior performance in the usual supervised source separation setting [9], in the score-informed setting, this NMF-based method works better than DNN-based methods [15].

The audio source separation method presented in [21] uses pretrained instrument sound synthesizers based on generative adversar-

ial networks (GANs). The GANs convert random vectors into audio signals. These vectors are thus difficult to interpret and introduce prior musical knowledge into the inputs. Furthermore, GANs are usually unstable during training, which requires painstaking hyperparameter exploration [22].

3. PROPOSED METHOD

3.1. Motivation and Strategy

One straightforward approach to use the DDSP autoencoder for polyphonic audio signals is to decompose the mixture into the source signals and apply the DDSP autoencoder to them. Since the DDSP autoencoder is trained with only clean instrument sounds, its synthesis performance is strongly affected by the artifacts and interferer signals included in the separated signals. Although the introduction of DNNs has rapidly increased the performance of source separation methods [9], the separated signals obtained even with the latest methods often include artifacts and interferer signals, and sometimes lack part of the target source signals. These separation failures lead to the performance degradation of the DDSP autoencoder, as we will show later in section 4.

To avoid this problem, we take an approach in which the synthesis parameters of the sources are directly extracted from the mixture. We focus on the fact that the part subsequent to the encoder of the DDSP autoencoder can be seen as a source audio synthesizer using the synthesis parameters (see Fig. 1). We call it the source synthesizer. Using multiple source synthesizers, we construct a generative model of the harmonic sound mixture as shown in Fig. 2. We also formulate the synthesis parameter extraction problem as an inverse problem using the proposed model.

3.2. DDSP Mixture Model

The proposed DDSP mixture model represents the mixture audio signal of R harmonic sources as the sum of the outputs of the source synthesizers driven with source-specific synthesis parameters. Let $r = 1, \ldots, R$ denote the source index and h_r represent the source synthesizer of source r. To distinguish the synthesis parameters and synthesized signals of each source, we hereafter add a subscript r to f_t, $\boldsymbol{z}_t$, l_t, and $\hat{\boldsymbol{x}}$. Fig. 2 shows the architecture of the DDSP mixture model. The synthesis parameters of source r, $\{f_{r,t}, \boldsymbol{z}_{r,t}, l_{r,t}\}_{t=1}^{T}$, are fed into h_r, and the synthesized signal of source r, $\hat{\boldsymbol{x}}_r$, is generated. Adding all the synthesized source signals yields the synthesized mixture signal $\hat{\boldsymbol{y}} \in \mathbb{R}^N$:

$$\hat{\boldsymbol{y}} = \sum_{r=1}^{R} \hat{\boldsymbol{x}}_r, \tag{3}$$

$$\hat{\boldsymbol{x}}_r = h_r(\{f_{r,t}, \boldsymbol{z}_{r,t}, l_{r,t}\}_{t=1}^{T}). \tag{4}$$

Note that although all h_r are separately depicted in Fig. 2, we can use the same pretrained source synthesizer for all sources when the DDSP autoencoder is trained with multiple instrument sounds.

The DDSP mixture model describes the forward process of the generation of the harmonic sound mixture. Thus, the synthesis parameter extraction problem amounts to the problem of finding the synthesis parameters of the sources so that they minimize the loss between the output of the DDSP mixture model $\hat{\boldsymbol{y}}$ and the observed mixture $\boldsymbol{y} \in \mathbb{R}^N$. As a loss function, we can use the multiscale spectral loss defined in (1). In summary, the problem of interest is

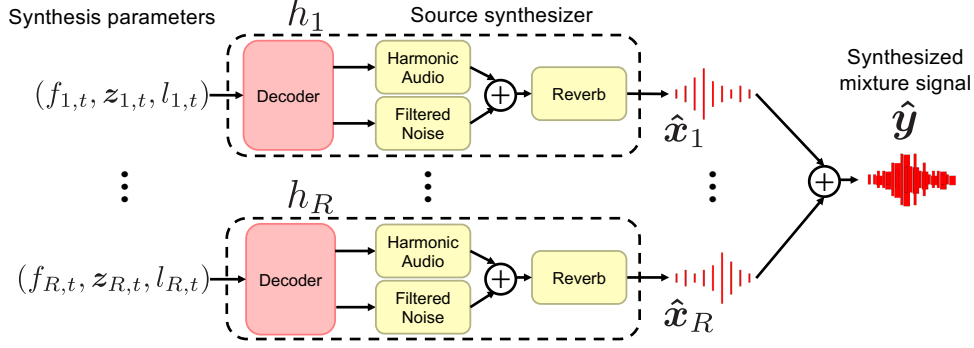**Fig. 2**. DDSP mixture model with R sources.

formulated as

$$\min_{\{f_{r,t}, \boldsymbol{z}_{r,t}, l_{r,t}\}_{r=1,t=1}^{R,T}} L(\boldsymbol{y}, \hat{\boldsymbol{y}}). \qquad (5)$$

Since h_r and this loss are differentiable, we can use a gradient descent algorithm for this problem. Note that that all h_r are trained in advance and fixed during this minimization. To distinguish the DDSP pretraining and this step, we call the latter the fitting step.

3.3. Initialization of F_0 and Loudness Using Score Information

Owing to the recent development of automatic music transcription [23–25], accurate score information can be extracted from polyphonic music signals. Since the proposed method uses the interpretable synthesis parameters, we can utilize the available score information for the initialization of $f_{r,t}$ and $l_{r,t}$. For simplicity, the score information is given in a musical instrument digital interface (MIDI) format and is assumed to be aligned in time with the input mixture. Let $p_{r,t} = -1, 0, \ldots, 127$ denote the MIDI note number of source r at time t, where $p_{r,t} = -1$ means that there are no played notes at that time, i.e., silence. By converting $p_{r,t}$ into the corresponding frequency, we can initialize $f_{r,t}$ as

$$f_{r,t} = \begin{cases} 440 \times 2^{(p_{r,t}-69)/12} & (p_{r,t} \geq 0) \\ 440 \times 2^{(p_r^{(\text{sil})}-69)/12} & (p_{r,t} = -1), \end{cases} \qquad (6)$$

where $p_r^{(\text{sil})}$ is the time average of nonnegative $p_{r,t}$s. The loudnesses $l_{r,t}$ are initialized with $l^{(\text{high})}$ for the active notes and $l^{(\text{low})}$ for the silences.

4. EXPERIMENTAL EVALUATION

4.1. Experimental Conditions

To evaluate the effectiveness of the proposed method, we conducted synthesis parameter extraction experiments on mixtures of two harmonic sources. We created test data using the PHENICX-Anechoic dataset [14, 26]. This dataset includes separate audio recordings of musical instruments of four symphonies: Symphony no. 1, fourth movement by G. Mahler (Mahler), an aria of Donna Elvira from the opera Don Giovanni by W. A. Mozart (Mozart), and Symphony no. 7, first movement by L. van Beethoven (Beethoven). It also includes time-aligned MIDI data. The test data consisted of mixtures of two different instruments or two of the same instrument. We used audio signals played with the viola (Va.), flute (Fl.), double bass (Db.), cello (Vc.), and bassoon (Bn.) and downsampled all audio signals to

16 kHz. We divided each mixture into 12-s segments from the beginning to the end and applied synthesis parameter extraction methods to them. The segments shorter than 12 s were not used for the evaluation.

We compared the following three methods.

SISS+DDSP: We separated the mixtures by the state-of-the-art score-informed source separation method (SISS) presented in [15] and applied the DDSP autoencoder to the separated signals. In our preliminary experiment, we found that this method provided higher separation performance than the recent DNN-based score-informed source separation method presented in [27]. We used the official implementation of SISS available at https://github.com/AntonioJMM/OISS_Minus-One.github.io and the same hyperparameters as those used in [15].

SISS+Proposed: We initialized $f_{r,t}$ and $l_{r,t}$ with those obtained with SISS+DDSP and ran the proposed method. The initial values of $z_{r,t}$ were drawn from a standard normal distribution. We used the Adam optimizer and set its learning rate at 0.1, which was decreased to 0.01 at the 1000th iteration and 0.001 at the 2000th iteration. For the multiscale spectral loss, we used Hann windows of 8, 16, 32, 64, 128, and 256-ms lengths and hop sizes of half of the corresponding frame lengths.

SI-Proposed: This model is a score-informed (SI) version of the proposed method. We initialized $f_{r,t}$ and $l_{r,t}$ with the score information described in section 3.3 and ran the proposed method. We experimentally determined that $l^{(\text{high})} = -6$ and $l^{(\text{low})} = -10$. The other conditions were the same as those of SISS+Proposed.

These methods used the same pretrained DDSP autoencoder. It was trained using the University of Rochester multimodal music performance (URMP) dataset [28], which consists of 44 classical chamber music pieces and audio signals played with 13 musical instruments. We used 35 out of the 44 music pieces as the training data (total 11523 s) and divided all instrument signals into 12-s segments. The segments shorter than 12 s were zero-padded up to 12-s length. We trained 3000 epochs using the Adam optimizer with a learning rate of 0.001. We used the multiscale spectral loss with the same frame lengths as in the fitting step. The synthesis parameters were computed at 32-ms intervals ($T = 375$).

As evaluation measures, we used the mean absolute errors (MAEs) in F_0, MFCC, and loudness between the estimates and the ground truths extracted from the source signals. The ground truths of F_0 were extracted using the CREPE model. Following [7], the F_0 MAEs were computed at the frames for which the confidences of the ground truths of F_0 were greater than or equal to 0.85. Note that since the estimated $f_{r,t}$ values may be negative, we floored them with 10^{-7} Hz. The MFCC estimates were computed from the audio

Table 1. Averages and standard errors of MAEs in F_0, MFCC, and loudness obtained with separation-based and proposed methods

Label	Instruments	Total dur. [s]	Method	F_0 [cent]	MFCC	Loudness [dB]
Va./_.	Va./Db. (Mahler), Va./Fl. (Mahler)	240	SISS+DDSP	**125 ± 35**	3.52 ± 0.12	**10.21 ± 1.75**
			SISS+Proposed	129 ± 34	2.84 ± 0.13	12.06 ± 1.89
			SI-Proposed	217 ± 48	**2.79 ± 0.13**	10.86 ± 0.63
Fl./_.	Fl./Bn. (Mozart), Fl./Va. (Mahler)	300	SISS+DDSP	130 ± 45	2.04 ± 0.27	33.43 ± 2.92
			SISS+Proposed	132 ± 45	**2.01 ± 0.18**	34.31 ± 3.28
			SI-Proposed	**90 ± 18**	2.25 ± 0.17	**10.54 ± 0.59**
Db./_.	Db./Vc. (Beethoven), Db./Va. (Mahler)	300	SISS+DDSP	1588 ± 1286	2.88 ± 0.15	16.33 ± 1.96
			SISS+Proposed	902 ± 682	**2.03 ± 0.15**	17.52 ± 2.61
			SI-Proposed	**164 ± 56**	2.05 ± 0.15	**10.41 ± 0.61**
Vc./_.	Vc./Db. (Beethoven), Vc./Bn. (Mozart)	260	SISS+DDSP	1572 ± 983	3.29 ± 0.09	10.49 ± 2.04
			SISS+Proposed	1178 ± 777	2.32 ± 0.13	10.29 ± 2.29
			SI-Proposed	**111 ± 24**	**2.21 ± 0.11**	**8.09 ± 0.26**
Bn./_.	Bn./Fl. (Mozart), Bn./Vc. (Mozart)	260	SISS+DDSP	1043 ± 506	2.45 ± 0.11	24.24 ± 1.65
			SISS+Proposed	911 ± 490	**1.90 ± 0.13**	31.39 ± 1.94
			SI-Proposed	**113 ± 10**	2.25 ± 0.15	**10.47 ± 0.59**
Va./Va.	Va. (Mahler)/Va. (Mozart)	120	SISS+DDSP	**140 ± 72**	3.83 ± 0.13	13.38 ± 1.16
			SISS+Proposed	146 ± 71	3.03 ± 0.12	13.16 ± 1.65
			SI-Proposed	155 ± 38	**2.89 ± 0.11**	**11.84 ± 0.62**
Fl./Fl.	Fl. (Mahler)/Fl. (Mozart)	120	SISS+DDSP	129 ± 68	1.72 ± 0.27	31.32 ± 4.05
			SISS+Proposed	116 ± 57	1.62 ± 0.21	33.84 ± 4.33
			SI-Proposed	**90 ± 26**	**1.44 ± 0.23**	**9.13 ± 0.52**
Db./Db.	Db. (Beethoven)/Db. (Mahler)	120	SISS+DDSP	844 ± 754	2.33 ± 0.17	15.74 ± 2.86
			SISS+Proposed	877 ± 752	**1.91 ± 0.13**	20.23 ± 3.32
			SI-Proposed	**195 ± 68**	1.97 ± 0.13	**10.51 ± 0.81**
Vc./Vc.	Vc. (Beethoven)/Vc. (Mahler)	120	SISS+DDSP	3438 ± 1851	2.79 ± 0.18	20.05 ± 3.21
			SISS+Proposed	2775 ± 1737	**2.27 ± 0.14**	24.35 ± 3.85
			SI-Proposed	**347 ± 144**	2.32 ± 0.16	**12.06 ± 0.91**
Bn./Bn.	Bn./Bn. (Beethoven)	180	SISS+DDSP	1139 ± 527	3.36 ± 0.15	9.75 ± 2.02
			SISS+Proposed	1151 ± 528	**2.38 ± 0.15**	10.87 ± 2.16
			SI-Proposed	**567 ± 248**	2.39 ± 0.16	**8.54 ± 0.32**

signals synthesized with the estimated synthesis parameters. We calculated the MFCCs using the log-mel-spectrogram with 128-ms frames, a 32-ms hop size, and 128 frequency bins ranging from 20 to 8000 Hz, and we used the first 30 coefficients.

4.2. Results

Table 1 shows the results of all methods, where the evaluation measures were computed at each segment and their averages and standard errors were computed. Here, the instrument names followed by the slash and underline denote the results for the mixtures of the different instruments, and those followed by the slash and same instrument names denote the results for the mixtures of the same instruments. The names inside the parentheses of the "Instruments" column indicate the music pieces in which the instrument audio signals are included. Although SISS+DDSP provided moderate performance in terms of F_0 for the Va. and Fl. mixtures, it showed much lower performance for the other mixtures. A similar tendency was observed for MFCC and loudness. When we listened to the separated signals of SISS+DDSP, we found that these signals lack part of the target sources and included artifacts and interferer signals. We also found that some of the synthesized signals of SISS+DDSP were considerably different from the target source signals. These results show that the separation-based method has unstable performance and that separation failures greatly degrade the synthesis parameter extraction performance.

Compared with SISS+DDSP, SISS+Proposed and SI-Proposed provided comparable and higher performances for most of the mix-

tures, particularly in terms of MFCC, showing the effectiveness of the proposed methods. Although the SISS+Proposed performances of F_0 and loudness were still low for the mixtures that SISS failed to separate, SI-Proposed consistently provided higher performances for all measures. Furthermore, we observed that the source signals synthesized by the proposed method had much more similar timbres to the target sources and audibly outperformed those obtained with SISS+DDSP. These results clearly show that the proposed method works stably and effectively compared with the separation-based method. Some synthesized examples are available at `https://sarulab-audio.github.io/DDSP_Mixture_Model/`.

Importantly, SI-Proposed had much lower standard errors in F_0 and loudness than the other methods for most of the mixtures. This result shows that the score information is useful for the proposed method and can decrease the deviations of the estimates.

5. CONCLUSION

We proposed the DDSP mixture model that represents the generation process of a mixture of harmonic audio signals, using part of the pretrained DDSP autoencoder as a source audio synthesizer. We also developed a synthesis parameter extraction method by fitting the output of the DDSP mixture model to the observed mixture. Through experiments using mixtures of sounds made by the different and same instruments, we showed that the proposed method outperforms a straightforward method that applies the DDSP autoencoder to signals separated with an existing audio source separation method.

6. REFERENCES

[1] C. Donahue, J. McAuley, and M. Puckette, "Adversarial audio synthesis," in *Proceedings of International Conference on Learning Representations*, 2019.

[2] B. Hayes, C. Saitis, and G. Fazekas, "Neural waveshaping synthesis," in *Proceedings of International Society for Music Information Retrieval Conference*, 2021.

[3] M. Michelashvili and L. Wolf, "Hierarchical timbre-painting and articulation generation," in *Proceedings of International Society for Music Information Retrieval Conference*, 2020, pp. 916–922.

[4] J. Nistal, S. Lattner, and G. Richard, "Drumgan: Synthesis of drum sounds with timbral feature conditioning using generative adversarial networks," in *Proceedings of International Society for Music Information Retrieval Conference*, 2020, pp. 590–597.

[5] A. Ramires, P. Chandna, X. Favory, E. Gómez, and X. Serra, "Neural percussive synthesis parameterised by high-level timbral features," in *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing*, 2020, pp. 786–790.

[6] S. Rouard and G. Hadjeres, "CRASH: Raw audio score-based generative modeling for controllable high-resolution drum sound synthesis," in *Proceedings of International Society for Music Information Retrieval Conference*, 2021.

[7] J. Engel, L. Hantrakul, C. Gu, and A. Roberts, "DDSP: Differentiable digital signal processing," in *Proceedings of International Conference on Learning Representations*, 2020.

[8] X. Serra and J. Smith, "Spectral modeling synthesis: A sound analysis/synthesis system based on a deterministic plus stochastic decomposition," *Computer Music Journal*, vol. 14, no. 4, pp. 12–24, 1990.

[9] F.-R. Stöter, A. Liutkus, and N. Ito, "The 2018 signal separation evaluation campaign," in *Proceedings of International Conference on Latent Variable Analysis and Signal Separation*, 2018, pp. 293–305.

[10] D. Petermann, P. Chandna, H. Cuesta, J. Bonada, and E. Gómez, "Deep learning based source separation applied to choir ensembles," in *Proceedings of International Society for Music Information Retrieval Conference*, 2020, pp. 733–739.

[11] H. Zhao, C. Gan, W.-C. Ma, and A. Torralba, "The sound of motions," in *Proceedings of IEEE/CVF International Conference on Computer Vision*, 2019, pp. 1735–1744.

[12] Y.-N. Hung, G. Wichern, and J. Le Roux, "Transcription is all you need: Learning to separate musical mixtures with score as supervision," in *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing*, 2021, pp. 46–50.

[13] M. Miron, J. Janer, and E. Gómez, "Monaural score-informed source separation for classical music using convolutional neural networks," in *Proceedings of International Society for Music Information Retrieval Conference*, 2017, pp. 55–62.

[14] M. Miron, J. J. Carabias-Orti, J. J. Bosch, E. Gómez, and J. Janer, "Score-informed source separation for multichannel orchestral recordings," *Journal of Electrical and Computer Engineering*, vol. 2016, 2016.

[15] A. J. Munoz-Montoro, J. J. Carabias-Orti, P. Vera-Candeas, F. J. Canadas-Quesada, and N. Ruiz-Reyes, "Online/offline score informed music signal decomposition: application to minus one," *EURASIP Journal on Audio, Speech, and Music Processing*, vol. 2019, no. 23, 2019.

[16] J. W. Kim, J. Salamon, P. Li, and J. P. Bello, "Crepe: A convolutional representation for pitch estimation," in *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing*, 2018, pp. 161–165.

[17] T. Nakamura, H. Kameoka, K. Yoshii, and M. Goto, "Timbre replacement of harmonic and drum components for music audio signals," in *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing*, 2014, pp. 7470–7474.

[18] N. Ono, K. Miyamoto, H. Kameoka, and S. Sagayama, "A real-time equalizer of harmonic and percussive components in music signals," in *Proceedings of International Conference on Music Information Retrieval*, 2008, pp. 139–144.

[19] N. Yasuraoka, T. Abe, K. Itoyama, T. Takahashi, T. Ogata, and H. G. Okuno, "Changing timbre and phrase in existing musical performances as you like: Manipulations of single part using harmonic and inharmonic models," in *Proceedings of ACM International Conference on Multimedia*, 2009, pp. 203–212.

[20] T. Nakamura and H. Kameoka, "Harmonic-temporal factor decomposition for unsupervised monaural separation of harmonic sounds," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 29, pp. 68–82, 2021.

[21] V. Narayanaswamy, J. J. Thiagarajan, R. Anirudh, and A. Spanias, "Unsupervised audio source separation using generative priors," in *Proceedings of INTERSPEECH*, 2020, pp. 2657–2661.

[22] A. Creswell, T. White, V. Dumoulin, K. Arulkumaran, B. Sengupta, and A. A. Bharath, "Generative adversarial networks: An overview," *IEEE Signal Processing Magazine*, vol. 35, no. 1, pp. 53–65, 2018.

[23] E. Benetos, S. Dixon, Z. Duan, and S. Ewert, "Automatic music transcription: An overview," *IEEE Signal Processing Magazine*, vol. 36, no. 1, pp. 20–30, 2019.

[24] Y.-T. Wu, B. Chen, and L. Su, "Multi-instrument automatic music transcription with self-attention-based instance segmentation," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 28, pp. 2796–2809, 2020.

[25] K. Shibata, E. Nakamura, and K. Yoshii, "Non-local musical statistics as guides for audio-to-score piano transcription," *Information Sciences*, vol. 566, pp. 262–280, 2021.

[26] J. Pätynen, V. Pulkki, and T. Lokki, "Anechoic recording system for symphony orchestra," *Acta Acustica united with Acustica*, vol. 94, no. 6, pp. 856–865, 2008.

[27] M. Gover and P. Depalle, "Score-informed source separation of choral music," in *Proceedings of International Society for Music Information Retrieval Conference*, 2020, pp. 231–239.

[28] B. Li, X. Liu, K. Dinesh, Z. Duan, and G. Sharma, "Creating a multitrack classical music performance dataset for multimodal music analysis: Challenges, insights, and applications," *IEEE Transactions on Multimedia*, vol. 21, no. 2, pp. 522–535, 2019.