Sequential Generation of Singing F0 Contours from Musical Note Sequences Based on WaveNet

Yusuke Wada* Ryo Nishikimi* Eita Nakamura* Katsutoshi Itoyama† Kazuyoshi Yoshii*
*Graduate School of Informatics, Kyoto University, Japan
E-mail: {wada, nishikimi, enakamura, yoshii}@sap.ist.i.kyoto-u.ac.jp
†School of Engineering, Tokyo Institute of Technology, Japan
E-mail: itoyama@ra.sc.e.titech.ac.jp

***Abstract*—This paper describes a method that can generate a continuous F0 contour of a singing voice from a monophonic sequence of musical notes (musical score) by using a deep neural autoregressive model called WaveNet. Real F0 contours include complicated temporal and frequency fluctuations caused by singing expressions such as vibrato and portamento. Although explicit models such as hidden Markov models (HMMs) have often used for representing the F0 dynamics, it is difficult to generate realistic F0 contours due to the poor representation capability of such models. To overcome this limitation, WaveNet, which was invented for modeling raw waveforms in an unsupervised manner, was recently used for generating singing F0 contours from a musical score with lyrics in a supervised manner. Inspired by this attempt, we investigate the capability of WaveNet for generating singing F0 contours without using lyric information. Our method conditions WaveNet on pitch and contextual features of a musical score. As a loss function that is more suitable for generating F0 contours, we adopted the modified cross-entropy loss weighted with the square error between target and output F0s on the log-frequency axis. The experimental results show that these techniques improve the quality of generated F0 contours.**

I. Introduction

Singing expressions observed in pitch fluctuations such as vibrato and portamento, volume dynamics, and vocal qualities play an important role in characterizing singing voices. In particular, it is useful to build a generative model of singing F0 contours including pitch fluctuations for automatically synthesizing natural and expressive singing voices. Such a model can also be used for singing style conversion and automatic parameter tuning of singing voice synthesizers including VOCALOID (a commercial singing voice synthesizer) [1]. Combining an F0 contour generation technique with a voice conversion technique [2]–[4], a singing voice of an arbitrary singer could be changed to that of another singer.

Conventional methods for generating singing F0 contours are based on explicit parametric modeling using a second-order linear system [5], hidden Markov models (HMMs) [6], [7], or a mixture of Gaussian process experts [8]. Although these models are useful for *analyzing* singer-specific pitch fluctuations, they are insufficient for *generating* natural F0 contours due to the limitation of the representation power (*e.g.*, linearity). Deep neural autoregressive models, which have the potential to overcome this limitation with their nonlinear representations, have recently been developed [9], [10]. A convolutional neural network (CNN) called WaveNet [10] was originally designed for modeling raw waveforms and applied for voice conversion [4], text-to-speech (TTS) synthesis [11], and audio synthesis of musical notes [12].

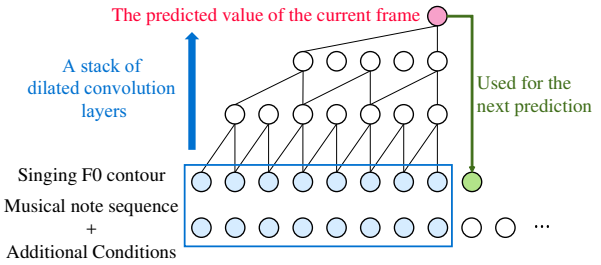

Fig. 1: Sequential prediction of a singing F0 contour from a musical note sequence with WaveNet. The past F0 sequence before the current frame and additional conditions are fed into a stack of dilated convolution layers. The singing F0 value of the current frame is predicted in an autoregressive manner. The prediction of the current frame is then used for the prediction of the next frame.

A deep generative model of singing F0 contours based on WaveNet has recently been proposed for singing voice synthesis [13]. The model can generate singing F0 contours from a sequence of musical notes with phoneme information in lyrics. Since singing F0 contours are affected by phonemes, when aiming at singing style conversion, it remains an open question whether singer-specific pitch fluctuations that are independent from phoneme information can be modeled appropriately.

In this paper, we investigate the use of WaveNet for generating a singing F0 contour from a sequence of musical notes without lyric information (Fig. 1). Using only a sequence of musical notes without lyrics, our method can deal with songs in any language. We aim to capture common patterns of pitch fluctuations that appear on specific note patterns regardless of language. Inspired by TTS synthesis [14], [15], we use four contextual features extracted only from the note sequence for conditioning WaveNet: (I) the musical note sequence after the current frame, (II) the relative position of the current frame from the start of the current musical note, (III) the duration of the current musical note, and (IV) a singer code. Feature I is used for improving the smoothness of generated F0 contours because humans can smoothly sing a song by considering

the following musical notes. Overshoot and preparation, for example, depend on the next note, and vibrato depends on the length of the current note. Features II and III are used for enhancing singing voice expressions that tend to appear at a certain position in a musical note of a certain length. Portamento, for example, tends to appear around the boundaries of musical notes, and vibrato tends to appear near the end of a relatively long note. Feature IV is used for learning the characteristics of singer-specific expressions.

We also investigate a modification of the loss function used by WaveNet. The original WaveNet uses the cross entropy for the loss function. The cross entropy function returns the same loss value for all prediction errors. When utilizing WaveNet for generation of F0 contours, the cross entropy loss cannot suppress a prediction that is far from the ground-truth F0. To overcome this problem, we introduce a weight function for the cross entropy that increases the value of the loss function if a predicted F0 becomes far from the ground-truth F0.

The main contribution of this study is to introduce contextual features extracted only from musical note sequences without phoneme information in lyrics. We also introduce a modified loss function that is more suitable for generating F0 contours than the cross entropy used in the original WaveNet. We evaluate the effectiveness of the two techniques in terms of the root mean squared errors (RMSEs) between the generated F0 contours and the ground-truth F0 contours. We compare the performances of our method and the original WaveNet. The experimental results show that the RMSEs become smaller than the original WaveNet by using those techniques. This suggests that the additional features and the modified loss function are important to prevent the predictions from being far from the ground-truth F0s, allowing natural deviations from the input musical note sequences.

II. Related Work

This section reviews related work on F0 contour modeling, singing note estimation, and singing voice synthesis. We also review F0 contour modeling in TTS that is similar to the one in singing voice synthesis.

A. Singing Voice Synthesis

Singing voice synthesis has been a hot research topic [1], [5], [6], [8], [13], [16]–[19]. The concatenative methods [1], [16], [17] are simple and powerful for synthesizing natural singing voices. A commercial software of singing voice synthesis named VOCALOID [1] is widely used for music production. Bonada *et al.* [16] made two databases of singing voices, a vowel database consisting of expressive a Capella voices containing solely vowels and a timbre database consisting of a single singer's voice. Ardaillon *et al.* [17] focused on generating natural F0 contours for concatenative singing voice synthesis and modeled variations of singing F0 contours as several separated layers (*e.g.*, vibrato, jitter, or melodic components) using B-spline curves. Statistical methods of singing voice synthesis and F0 contour generation have also been proposed [5]–[8], [18]. Sinsy [18] is an HMM-based statistical singing voice synthesizer incorporating lyrics, tones and durations simultaneously. To explicitly model singing F0 contours, a second-order linear system [5], an HMM [6], [7], and a mixture of Gaussian process experts [8] were proposed.

DNN-based methods have been studied actively [13], [19]. Nishimura *et al.* [19] replaced the HMM-based modeling of Sinsy [18] with a fully-connected DNN architecture and improved the naturalness of the synthesized singing voices. Blaauw *et al.* [13] proposed the state-of-the-art singing voice synthesizer consisting of phonetic timing, pitch, and timbre models based on WaveNet. As mentioned in Section I, the pitch model is used for generating singing F0 contours from musical note sequences while referring to the phonetic information of lyrics.

B. Text-to-Speech Synthesis

TTS synthesis has been developed as well as singing voice synthesis [11], [14], [15], [20]–[23]. The concatenative methods [20], [21] have been developed in the 1990s, resulting in high-level quality of synthesized speeches. Since the units of speech voices used in such methods contain natural accents and fluctuations, modeling F0 contours is not necessary for those methods. After that, HMM-based statistical methods have been studied [22], [23]. Zen *et al.* [22] developed an extension of the HMM, called a trajectory HMM, that incorporates explicit relations between static and dynamic features for vocoder synthesis. Kameoka *et al.* [23] proposed an HMM-based generative model of speech F0 contours on the basis of a probabilistic formulation of Fujisaki's model [24], a second-order linear system representing the control mechanism of vocal cord vibration.

DNN-based end-to-end synthesizers have recently been developed [11], [14], [15]. Fan *et al.* [15] and Zen and Sak [14] used a long short-term memory (LSTM) for generating acoustic features for vocoder synthesis. In their models, contextual features extracted from the input text were used as the input of the LSTM: phoneme-level linguistic features (*e.g.*, phoneme identities, stress marks, the number of syllables in a word, the position of the current syllable in the phrase), the position of the current frame in the current phoneme, and the duration of the current segment. Shen *et al.* [11] combined an LSTM-based generator of acoustic features and a WaveNet-based vocoder.

C. Singing Note Estimation

Estimation of musical notes from sung melodies has actively been studied [25]–[30]. Paiva *et al.* [25] proposed a method for musical note estimation from polyphonic music signals by multipitch detection, multipitch trajectory construction, trajectory segmentation, and elimination of irrelevant notes. Molina *et al.* [26] proposed a method of monophonic singing transcription that focuses on the hysteresis characteristics of singing F0 contours. HMM-based methods [27]–[29] have been used for capturing the dynamic fluctuations of singing voices. For example, Raphael [29] modeled rhythm and onset deviations of singing F0 contours for estimating pitches,

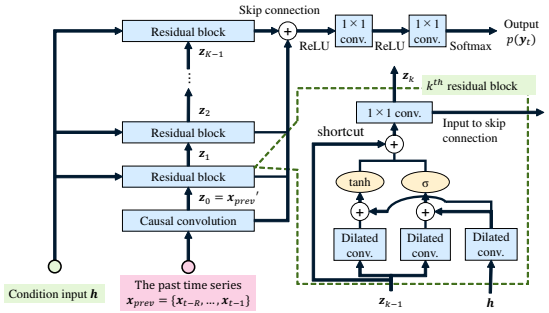

Fig. 2: Overall architecture of WaveNet.

rhythms, and tempos. Ryynänen and Klapuri [27] proposed a hierarchical HMM that captures various singing fluctuations in individual musical notes. Yang *et al.* [28] modeled the generative process of trajectories in the F0-ΔF0 plane based on a hierarchical HMM. A DNN-based method has recently been studied by Zhu *et al.* [30]. The method fuses the outputs of two DNNs, one for estimating singing pitch series from a polyphonic audio signal and one for selecting pitch series from multiple recordings of monophonic singing.

III. Proposed Method

We first review the mathematical formulation of WaveNet [10] and then explain the proposed method of modeling and generating singing F0 contours based on WaveNet with several modifications.

A. Review on WaveNet

In this study, WaveNet [10] is used for generating an F0 contours from a musical note sequence (Fig. 2). It calculates the joint probability of time-series data $\boldsymbol{x} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_T\}$ as

$$p(\boldsymbol{x}) = \prod_{t=1}^{T} p(\boldsymbol{x}_t | \boldsymbol{x}_1, \ldots, \boldsymbol{x}_{t-1}). \tag{1}$$

The time-series data are usually represented as a sequence of one-hot vectors, *i.e.*, binary representation of categorical data. Because the size of the neural network is limited, WaveNet cannot consider all the past samples from the current time step. It therefore approximates Eq. (1) by the following probability:

$$p(\boldsymbol{x}) = \prod_{t=1}^{T} p(\boldsymbol{x}_t | \boldsymbol{x}_{t-R}, \ldots, \boldsymbol{x}_{t-1}). \tag{2}$$

The R in Eq. (2), called the receptive field, determines the number of samples actually considered for predicting the joint probability.

Here we explain the process for predicting the output probability of $\boldsymbol{x}_t$ from the sequence of the past samples $\boldsymbol{x}_{\mathrm{prev}} = \{\boldsymbol{x}_{t-R}, \ldots, \boldsymbol{x}_{t-1}\}$. The joint probability shown in Eq. (2) is represented with a stack of residual blocks. Residual block is a unit that includes three dilated convolution (DC) layers and outputs the merged result of two nonlinear activation functions. The $\boldsymbol{x}_{\mathrm{prev}}$ is converted to $\boldsymbol{x}'_{\mathrm{prev}}$ through 1×1 causal convolution layer, and is input into the first residual block. 1×1 convolution means a 1-dimensional convolution whose filter size and stride are both 1. Causal convolution means a convolution whose output does not depend on future inputs. Denoting $\boldsymbol{z}_0 = \boldsymbol{x}'_{\mathrm{prev}}$, the output of the k^{th} residual block $\boldsymbol{z}_k (k = 0, 1, \ldots, K)$ is calculated from the previous output $\boldsymbol{z}_{k-1}$ as follows:

$$\boldsymbol{z}_k = \tanh(W_{f,k} * \boldsymbol{z}_{k-1}) \odot \sigma(W_{g,k} * \boldsymbol{z}_{k-1}) \tag{3}$$

where the symbol $*$ represents convolution operation, the symbol $\odot$ represents element-wise product operation, $W_{f,k}$ and $W_{g,k}$ are the filters of the k^{th} DC layers, and $\tanh(\cdot)$ and $\sigma(\cdot)$ represent the hyperbolic tangent the sigmoid function, respectively. All the outputs of the residual blocks through 1×1 convolution layers are summed by the skip connection, and the overall output of WaveNet is the softmax probability of a one-hot vector.

The dilation rates of the DC layers in the residual blocks are usually doubled for every layer up to a limit and then grouped into several dilation cycles, *e.g.*,

$$1, 2, 4, \ldots, 512, 1, 2, 4, \ldots, 512, 1, 2, 4, \ldots, 512.$$

Given the number of residual blocks K, the dilation rate d_K of the last DC layer in the last K^{th} residual block and the number of the repetitions of the dilation cycles B, the receptive field R is calculated as follows:

$$R = 2^{d_K} \cdot B. \tag{4}$$

The exponential increase of the dilation rate with the linear increase of the number of layers achieves exponential growth of the receptive field [31]. The repetitions of the dilation cycles further increases the total non-linearity and capability of the model.

WaveNet can condition the joint probability shown in Eq. (1) with an additional input sequence $\boldsymbol{h}$, which corresponds to an additional feature sequence extracted from a musical note sequence in our method, as follows:

$$p(\boldsymbol{x}) = \prod_{t=1}^{T} p(x_t | x_1, \ldots, x_{t-1}, \boldsymbol{h}). \tag{5}$$

To calculate Eq. (5), Eq. (3) is replaced with the following equation:

$$\begin{aligned} \boldsymbol{z}_k = \tanh(&W_{f,k} * \boldsymbol{z}_{k-1} + W'_{f,k} * \boldsymbol{h}) \\ &\odot \sigma(W_{g,k} * \boldsymbol{z}_{k-1} + W'_{g,k} * \boldsymbol{h}) \end{aligned} \tag{6}$$

where $W'_{f,k}$ and $W'_{g,k}$ are the filters of 1×1 convolution layers for the condition sequence.

B. Sequential Prediction of F0 Contours from Musical Note Sequences

This section describes the proposed method for generating an F0 contour from a musical note sequence with WaveNet (Fig. 3). The input musical note sequence is represented as $\boldsymbol{h} = \{\boldsymbol{h}_t\}_{t=1}^{T}$, where T is the number of frames in a musical sequence and $\boldsymbol{h}_t$ is a musical note at time t represented as a one-hot vector. The output F0 contour is represented as $\boldsymbol{x} =$

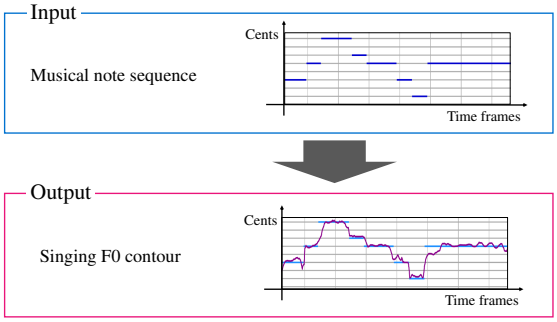

Fig. 3: Problem specification of our method. A singing F0 contour is generated from an input musical note sequence.

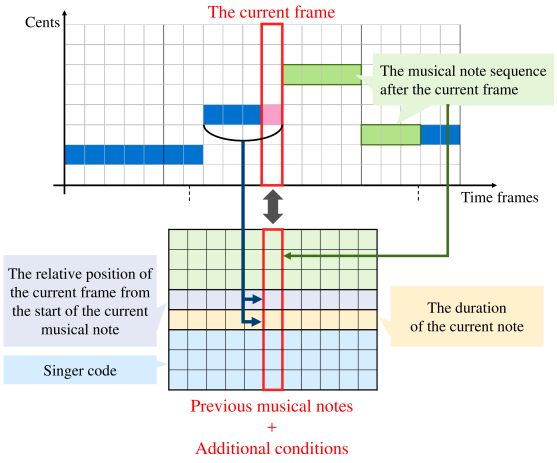

Fig. 4: Additional conditions for WaveNet extracted from the musical note sequence.

$\{\boldsymbol{x}_t\}_{t=1}^{T}$ where $\boldsymbol{x}_t$ is a log-scale F0 value represented as a one-hot vector.

The joint probability $p(\boldsymbol{x})$ is calculated according to Eq. (5). $\boldsymbol{x}$ is input to the first residual block through the initial 1×1 causal convolution layer. The output of the k^{th} residual block is calculated from the output of the k^{th} DC layer and the condition sequence $\boldsymbol{h}$ according to Eq. (6). The overall output of WaveNet is the probability of a F0 contour through softmax function. In the training phase, ground-truth F0 values are used to calculate the joint probability shown in Eq. (5). In contrast, previously generated F0 values are used for the calculation in the generation phase and F0 values are drawn from the joint probability at each time step.

C. Condition Inputs for WaveNet

As mentioned in Section I, our method uses four additional feature sequences other than musical notes depicted in Fig. 4 to condition the prediction of WaveNet. All the feature sequences are concatenated in one sequence and fed into WaveNet, i.e., the condition sequence $\boldsymbol{h}$ in Eq. (5) is replaced with a sequence $\boldsymbol{h}' = \{(\boldsymbol{h}_t, \boldsymbol{c}_t)\}_{t=1}^{T}$ where $\boldsymbol{c}_t$ is a concatenated vector of the additional features, and $(\boldsymbol{h}_t, \boldsymbol{c}_t)$ represents concatenation of the two vectors. As shown in Fig. 4, the musical note sequence and singers code are represented as one-hot vectors, and the relative position and the duration as real values, respectively.

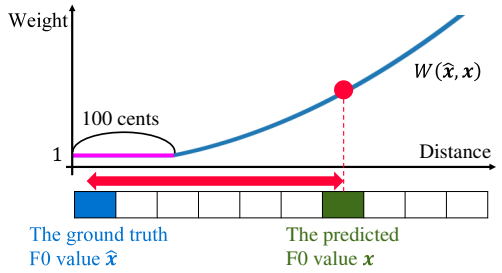

Fig. 5: The weight function used with the cross-entropy loss.

D. Loss Function

This method uses the following loss function $\mathcal{L}$ that is a modified version of cross-entropy function and acts as the square loss for WaveNet to prevent the D-dimensional softmax prediction $p(\boldsymbol{x})$ from being unnaturally far from the target F0 contour $\hat{\boldsymbol{x}}$:

$$\mathcal{L}(\hat{\boldsymbol{x}}, \boldsymbol{x}) = W(\hat{\boldsymbol{x}}, \boldsymbol{x}) H(\hat{\boldsymbol{x}}, \boldsymbol{x}) \tag{7}$$

where

$$H(\hat{\boldsymbol{x}}, \boldsymbol{x}) = -\sum_{d=1}^{D} \hat{x}_d \log p(x_d) \tag{8}$$

is the cross entropy between $\hat{\boldsymbol{x}}$ and $\boldsymbol{x}$, and given $d(\hat{\boldsymbol{x}}, \boldsymbol{x}) = |\operatorname{argmax} \hat{\boldsymbol{x}} - \operatorname{argmax} \boldsymbol{x}|$, the weight function $W(\hat{\boldsymbol{x}}, \boldsymbol{x})$ in proportion to the squared error between $\hat{\boldsymbol{x}}$ and $\boldsymbol{x}$ is defined as

$$W(\hat{\boldsymbol{x}}, \boldsymbol{x}) = \begin{cases} 1 & \text{if } d(\hat{\boldsymbol{x}}, \boldsymbol{x}) < 100 \\ (d(\boldsymbol{x}, \boldsymbol{x})/100)^2 & \text{otherwise.} \end{cases} \tag{9}$$

The graph representation of $W(\hat{\boldsymbol{x}}, \boldsymbol{x})$ is depicted in Fig. 5. The parameters of the weight function are empirically set. As shown in Eq. (9), the $W(\hat{\boldsymbol{x}}, \boldsymbol{x})$ does not apply weight to cross entropy when the distance of the prediction from the target $d(\hat{\boldsymbol{x}}, \boldsymbol{x})$ is less than 100 cents. This aims to enable natural deviations from the input musical note for the F0 prediction of WaveNet. The further an F0 value is predicted from the target F0 value, the more unnatural the predicted F0 contour is considered to become. The $W(\hat{\boldsymbol{x}}, \boldsymbol{x})$ is considered to suppress such unnatural deviations.

IV. Evaluation

This section reports experiments conducted to evaluate the performance of the proposed generative model of singing F0 contours.

A. Experimental Conditions

Among the 100 pieces of popular music in the RWC music database [32], we used 50 pieces for training of the generative model and 11 pieces for evaluation. We obtained musical note sequences and singer codes from the annotation data [33]. Singing F0 contours were also obtained from the annotation data or automatically estimated by using the state-of-the-art

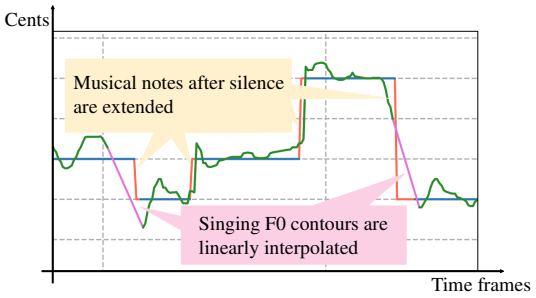

Fig. 6: Interpolation of musical note sequence and F0 contour. Silent durations that are shorter than 200 milliseconds in singing F0 contours are linearly interpolated, and such silent durations in musical note sequences are interpolated with the adjacent note pitch after the duration.

melody extraction method proposed in [34]. We used only voiced durations of the dataset. In the training phase, we used the past singing F0 samples of the annotation data rather than using the past predictions. In the generation phase, in contrast, the past predictions were used in an autoregressive manner. The initial F0 values for the generation were extracted from the estimation data.

We augmented the dataset using pitch shift and interpolation of unvoiced durations. The unvoiced durations shorter than 200 milliseconds in the musical note sequences and the singing F0 contours were interpolated (Fig. 6). Each musical note sequence and singing F0 contour was pitch-shifted based on the shift amount randomly sampled from $\{-1200, -1100, \ldots, 1200\}$. To increase variation of the F0 contours, we added Gaussian noises on the vocal F0 values of the annotation data as follows:

$$\boldsymbol{x}' = \boldsymbol{x} + \epsilon \tag{10}$$

where $\boldsymbol{x}$ is the original vocal F0 value at each frame and ϵ is a noise drawn from a Gaussian distribution $\mathcal{N}(0, 100)$ whose variance corresponds to a semitone.

The F0 values of singing voices in the range from C2 to C6 were discretized in units of 10 cents. The musical notes in the same range were also discretized in units of 100 cents. The F0 values and musical notes out of the range were treated as silent. The discretized F0 contours then transformed into 481-dimensional one-hot vectors, and the discretized musical note sequences into 49-dimensional one-hot vectors, respectively. As the musical note sequence after the current frame, the next 50 samples (500 milliseconds) were used. The dimensionality of one conditional feature vector was therefore 2526 (= 481 + 49 * 50 + 1 + 1 + 74).

For the parameters of WaveNet used in our method, we connected 15 DC layers grouped in 3 dilation cycles, i.e., the dilation rates were set to 1, 2, ..., 16, 1, 2, ..., 16, 1, 2, ..., 16. The receptive field was therefore 96 samples according to Eq. (4). The number of channels of the DC layer and the 1×1 convolution layer in each residual block were set to 64.

TABLE I: Average RMSE between the ground-truth F0 contours and the generated F0 contours.

Modification of loss function	Additional conditions	RMSE [cent]
		165.4
✓		158.2
	✓	158.1
✓	✓	**150.1**

The number of 1×1 convolution channel between the skip connection and the softmax layer was set to 1024. We used Adam optimizer [35] whose hyperparameters were set to $\alpha = 0.001, \beta_1 = 0.9, \beta_2 = 0.999, \epsilon = 10^{-8}$.

To quantitatively evaluate the prediction quality of our method, we calculated the root mean squared error (RMSE) between the generated F0 contour and the ground-truth F0 contour for each song in the evaluation dataset. The overall RMSE was calculated as the average of the framewise RMSEs of all the songs for evaluation.

B. Experimental Results

The experimental results are shown in Table I. Comparing the four RMSEs, we confirmed that the weighted cross-entropy loss function and the additional conditions both contributed to generation of more natural F0 contours from the musical note sequences. Although the RMSE is useful for measuring the quality of generated F0 contours, objective evaluation is required for further investigation of the quality of the generated F0 contours. We therefore plan to conduct listening test using the singing voices or sinusoidal waves synthesized from the generated F0 contours.

Several examples of the F0 contours generated by original WaveNet and our method are illustrated in Fig. 7. In each figure, the F0 contour shown in the upper side was generated from original WaveNet, and the F0 contour in the lower side from our method. In the three examples shown in Fig. 7a, 7c and **??**, the unnatural deviations of the F0 contours shown in the upper side examples are suppressed. In addition, onset deviations, preparations, overshoot, and undershoot are observed in these examples. Especially in Fig. 7a, some vibrato durations can be seen. These pitch fluctuations are considered to be enhanced by the additional conditions. These results indicate the effectiveness of the techniques used in our method for improving the quality of the generated F0 contours. While these three examples show the capability of our method, the example shown in Fig. 7d still contains some unnatural deviations from the input musical note sequence, especially for short notes. A possible solution is to add some extra sequences representing vibrato durations or tempo to the conditions.

V. Conclusion

This paper proposed a WaveNet-based generator of singing F0 contours from musical note sequences without lyric information. We investigated the capability of WaveNet to generate singing F0 contours using pitch and contextual features and a modified cross-entropy loss. We confirmed that the additional

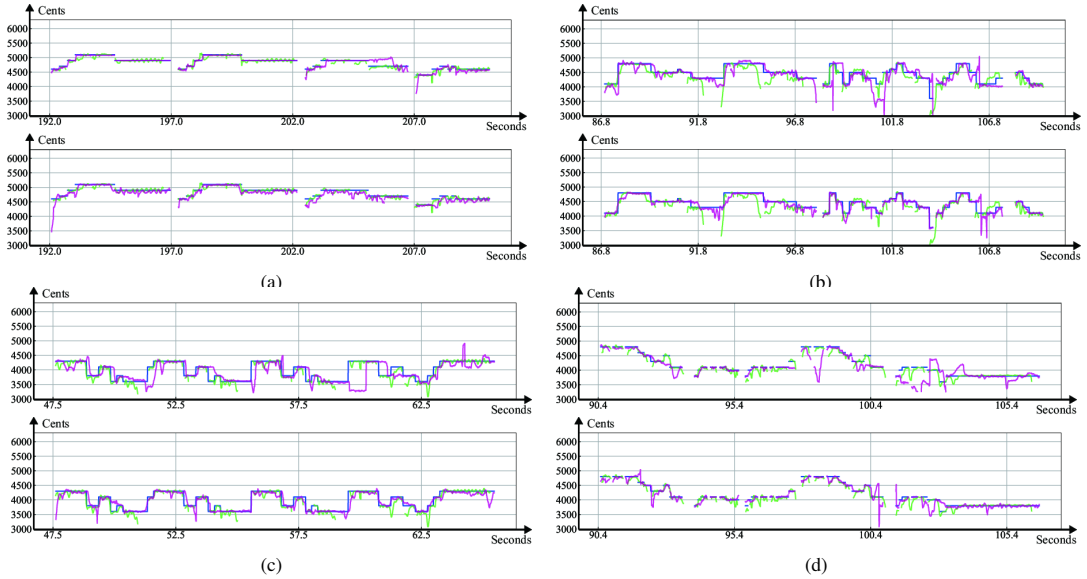

Fig. 7: Examples of the singing F0 contours generated by original WaveNet and our method. The blue line indicates the musical note sequence, the purple line the generated singing F0 contour, and the green line the F0 contour of the annotation data as a reference. The figure in the lower (upper) side depicts the F0 contour generated with (without) the additional conditions or the modification of the loss function.

features and the modified loss function both contribute to improving the quality of generated F0 contours.

It is interesting to apply the proposed method for singing style conversion. We plan to apply the same architecture of the proposed model for generating volume contours of singing voices, which are considered important for representing singing expressions as well as singing F0 contours. Singing style conversion to a specific singer's one can be achieved using the expression models for singing F0 and volume contours representing the style of the singer. When developing such expression models, the size of dataset is expected to be too small to train those models. Transfer learning is would be helpful for this problem.

VI. Acknowledgement

This work was supported in part by JST ACCEL No. JPMJAC1602, JSPS KAKENHI No. 26700020 and No. 16H01744, and Grant-in-Aid for JSPS Research Fellow No. 16J05486.

References

[1] H. Kenmochi and H. Ohshita. VOCALOID-Commercial Singing Synthesizer Based on Sample Concatenation. In *Proc. Interspeech*, pages 4009–4010, 2007.

[2] C. Hsu, H. Hwang, Y. Wu, Y. Tsao, and H. Wang. Voice Conversion from Unaligned Corpora Using Variational Autoencoding Wasserstein Generative Adversarial Networks. In *Proc. Interspeech*, 2017.

[3] T. Kinnunen, L. Juvela, P. Alku, and J. Yamagishi. Non-parallel Voice Conversion Using I-vector PLDA: Towards Unifying Speaker Verification and Transformation. In *Proc. ICASSP*, pages 5535–5539, 2017.

[4] K. Kobayashi, T. Hayashi, A. Tamamori, and T. Toda. Statistical Voice Conversion with WaveNet-based Waveform Generation. In *Proc. Interspeech*, pages 1138–1142, 2017.

[5] T. Saitou, M. Unoki, and M. Akagi. Development of an F0 Control Model Based on F0 Dynamic Characteristics for Singing-voice Synthesis. *J. Speech Communication*, 46(3):405–417, 2005.

[6] S. W. Lee, S.T. Ang, M. Dong, and H. Li. Generalized F0 Modelling with Absolute and Relative Pitch Features for Singing Voice Synthesis. In *Proc. ICASSP*, pages 429–432, 2012.

[7] Y. Ohishi, H. Kameoka, D. Mochihashi, and K. Kashino. A Stochastic Model of Singing Voice F0 Contours for Characterizing Expressive Dynamic Components. In *Proc. Interspeech*, 2012.

[8] Y. Ohishi, D. Mochihashi, H. Kameoka, and K. Kashino. Mixture of Gaussian Process Experts for Predicting Sung Melodic Contour with Expressive Dynamic Fluctuations. In *Proc. ICASSP*, pages 3714–3718, 2014.

[9] S. Mehri, K. Kumar, I. Gulrajani, R. Kumar, S. Jain, J. Sotelo, A. Courville, and Y. Bengio. SampleRNN: An Unconditional End-to-End Neural Audio Generation Model. In *Proc. ICLR*, pages 1–11, 2017.

[10] A. van den Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, and K. Kavukcuoglu. WaveNet: A Generative Model for Raw Audio. In *arXiv preprint arXiv:1609.03499*, pages 1–15, 2016.

[11] J. Shen, R. Pang, R.J. Weiss, M. Schuster, N. Jaitly, Z. Yang, Z. Chen, Y. Zhang, Y. Wang, R.J. Skerry-Ryan, R.A. Saurous, Y. Agiomyrgiannakis, and Y. Wu. Natural TTS Synthesis by Conditioning WaveNet on Mel Spectrogram Predictions. In *Proc. ICASSP*, pages 1–5, 2018.

[12] J. Engel, C. Resnick, A. Roberts, S. Dieleman, D. Eck, K. Simonyan, and M. Norouzi. Neural Audio Synthesis of Musical Notes with WaveNet Autoencoders. In *Proc. ICML*, pages 1068–1077, 2017.

[13] M. Blaauw and J. Bonada. A Neural Parametric Singing Synthesizer Modeling Timbre and Expression from Natural Songs. *J. Applied Sciences*, 7(12):1313, 2017.

[14] H. Zen and H. Sak. Unidirectional Long Short-term Memory Recurrent Neural Network with Recurrent Output Layer for Low-latency Speech Synthesis. In *Proc. ICASSP*, pages 4470–4474, 2015.

[15] Y. Fan, Y. Qian, F. Xie, and F.K. Soong. TTS Synthesis with Bidirectional LSTM Based Recurrent Neural Networks. In *Proc. Annual Conference of the International Speech Communication Association, Interspeech*, pages 1964–1968, 2014.

[16] J. Bonada, M. Umbert, and M. Blaauw. Expressive Singing Synthesis Based on Unit Selection for the Singing Synthesis Challenge 2016. In *Proc. Interspeech*, pages 1230–1234, 2016.

[17] L. Ardaillon, G. Degottex, and A. Roebel. A Multi-layer F0 Model for Singing Voice Synthesis Using A B-spline Representation with Intuitive Controls. In *Proc. Interspeech*, pages 3375–3379, 2015.
[18] K. Saino, H. Zen, Y. Nankaku, A. Lee, and K. Tokuda. An HMM-based Singing Voice Synthesis System. In *Proc. Interspeech*, pages 2274–2277, 2006.
[19] M. Nishimura, K. Hashimoto, K. Oura, Y. Nankaku, and K. Tokuda. Singing Voice Synthesis Based on Deep Neural Networks. In *Proc. Interspeech*, pages 2478–2482, 2016.
[20] M. Beutnagel, A. Conkie, J. Schroeter, Y. Stylianou, and A. Syrdal. The AT&T NextGen TTS System. In *Proc. Joint ASA/EAA/DAEA Meeting*, pages 15–19, 1999.
[21] G. Coorman, J. Fackrell, P. Rutten, and B.V. Coile. Segment Selection in the L&H Realspeak Laboratory TTS System. In *Proc. Spoken Language Processing*, pages 395–398, 2000.
[22] H. Zen, K. Tokuda, and T. Kitamura. Reformulating the HMM as a Trajectory Model by Imposing Explicit Relationships between Static and Dynamic Feature Vector Sequences. *J.Computer Speech and Language*, 21(1):153–173, 2006.
[23] H. Kameoka, K. Yoshizato, T. Ishihara, K. Kadowaki, Y. Ohishi, and K. Kashino. Generative Modeling of Voice Fundamental Frequency Contours. *J. IEEE/ACM TASLP*, 23(6):1042–1053, 2015.
[24] H. Fujisaki. A Note on the Physiological and Physical Basis for the Phrase and Accent Components in the Voice Fundamental Frequency Contour. *J. Vocal Physiology : Voice Production, Mechanisms and Functions*, 1988.
[25] R.P. Paiva, T. Mendes, and A. Cardoso. On the Detection of Melody Notes in Polyphonic Audio. In *Proc. ISMIR*, pages 175–182, 2005.
[26] E. Molina, L.J. Tardon, A.M. Barbancho, and I. Barbancho. SiPTH: Singing Transcription Based on Hysteresis Defined on the Pitch-Time Curve. *J. IEEE/ACM TASLP*, 23(2):252–263, 2015.
[27] M.P. Ryynänen and A.P. Klapuri. Automatic Transcription of Melody, Bass Line, and Chords in Polyphonic Music. *J. Computer Music Journal*, 32(3):72–86, 2008.
[28] L. Yang, A. Maezawa, and B.L.S. Jordan. Probabilistic Transcription of Sung Melody Using A Pitch Dynamic Model. In *Proc. ICASSP*, pages 301–305, 2017.
[29] C. Raphael. A Graphical Model for Recognizing Sung Melodies. In *Proc. ISMIR*, pages 658–663, 2005.
[30] B. Zhu, F. Wu, K. Li, Y. Wu, F. Huang, and Y. Wu. Fusing Transcription Results from Polyphonic and Monophonic Audio for Singing Melody Transcription in Polyphonic Music. In *Proc. ICASSP*, pages 296–300, 2017.
[31] F. Yu and V. Koltun. Multi-scale Context Aggregation by Dilated Convolutions. In *Proc. ICLR*, pages 1–13, 2016.
[32] M. Goto, H. Hashiguchi, T. Nishimura, and R. Oka. RWC Music Database: Popular, Classical, and Jazz Music Databases. In *Proc. ISMIR*, pages 229–230, 2003.
[33] M. Goto. AIST Annotation for RWC Music Database. In *Proc. ISMIR*, pages 359–360, 2006.
[34] Y. Ikemiya, K. Yoshii, and K. Itoyama. Singing Voice Analysis and Editing Based on Mutually Dependent F0 Estimation and Source Separation. In *Proc. ICASSP*, pages 574–578, 2015.
[35] D.P. Kingma and J.L. Ba. Adam: A Method for Stochastic Optimization. In *Proc. ICLR*, pages 1–15, 2015.