Sinsy: A Deep Neural Network-Based Singing Voice Synthesis System

Yukiya Hono, Kei Hashimoto, *Member, IEEE*, Keiichiro Oura, Yoshihiko Nankaku, *Member, IEEE*, and Keiichi Tokuda, *Fellow, IEEE*

***Abstract*—This paper presents Sinsy, a deep neural network (DNN)-based singing voice synthesis (SVS) system. In recent years, DNNs have been utilized in statistical parametric SVS systems, and DNN-based SVS systems have demonstrated better performance than conventional hidden Markov model-based ones. SVS systems are required to synthesize a singing voice with pitch and timing that strictly follow a given musical score. Additionally, singing expressions that are not described on the musical score, such as vibrato and timing fluctuations, should be reproduced. The proposed system is composed of four modules: a time-lag model, a duration model, an acoustic model, and a vocoder, and singing voices can be synthesized taking these characteristics of singing voices into account. To better model a singing voice, the proposed system incorporates improved approaches to modeling pitch and vibrato and better training criteria into the acoustic model. In addition, we incorporated PeriodNet, a non-autoregressive neural vocoder with robustness for the pitch, into our systems to generate a high-fidelity singing voice waveform. Moreover, we propose automatic pitch correction techniques for DNN-based SVS to synthesize singing voices with correct pitch even if the training data has out-of-tune phrases. Experimental results show our system can synthesize a singing voice with better timing, more natural vibrato, and correct pitch, and it can achieve better mean opinion scores in subjective evaluation tests.**

***Index Terms*—Automatic pitch correction, neural network, singing voice synthesis, timing modeling, vibrato modeling.**

I. Introduction

SINGING voice synthesis (SVS) is a technique of generating singing voices from musical scores. A unit-selection method [1], [2] can automatically synthesize a singing voice by concatenating short waveform units selected from a database. While such systems can provide good sound quality and naturalness in certain settings, it is impossible to guarantee that the units will always be connected smoothly. Moreover, since it also tends to have limited flexibility, large databases are generally required to synthesize singing voices.

Statistical parametric SVS systems such as hidden Markov model (HMM)-based SVS systems [3] have been proposed to avoid the problems described above. The singing voice waveform is synthesized from the acoustic parameters predicted by a trained HMM, thereby requiring less data to construct a system compared to unit-selection systems. However, HMM-based systems suffer from over-smoothing that degrades the naturalness of synthesized singing voices.

In recent years, deep neural networks (DNNs) have significantly improved in various speech processing tasks such as speech recognition [4], speech synthesis [5], [6], and voice conversion [7]. DNN-based SVS systems [8], [9] have also been proposed and demonstrated their superiority over HMM-based ones. A feed-forward neural network (FFNN) is utilized as an acoustic model to represent the mapping function between the musical score feature and the acoustic feature. Recently, recurrent neural networks (RNNs) with long short-term memory (LSTM), convolutional neural networks (CNNs), and deep autoregressive (AR) models have been incorporated into SVS systems to model the acoustic features more appropriately [10]–[13]. Trajectory training [14] and adversarial training [15] have also been incorporated into SVS systems to improve training criteria and achieve higher singing voice quality [9], [16].

In SVS systems, singing voices must be synthesized accurately following the input musical score. Methods such as pitch normalization [8] and data augmentation [12], [17] have been proposed for DNN-based SVS systems to generate fundamental frequency (F0) following the note pitch in the input musical score. Vibrato is the periodic fluctuation of the pitch and is another essential point of modeling a singing voice. Some systems model vibrato-like fluctuations as a part of the F0 [10], [12], [13]. Our previous work [9] separates the vibrato from the F0 and models it as sinusoidal parameters, enabling reproduction and control of the vibrato. The temporal structure of a singing voice is heavily constrained by note length in a musical score, but the start timing of musical notes and a singing voice do not always match. A framework with a time-lag model and a duration model has been proposed to determine the phone durations under note length constraints considering these timing fluctuations [9]. These techniques are essential for synthesizing a human-like natural singing voice.

When building SVS systems, a pitch correction technique is sometimes necessary to avoid generating out-of-tune singing voices. Since DNN-based SVS is a statistical approach that tries to reproduce training data, it tends to generate an out-of-tune pitch if the training data contains out-of-tune phrases. The pitch accuracy significantly impacts the subjective quality of

Manuscript received March 26, 2021; revised June 25, 2021; accepted July 26, 2021. Date of publication August 11, 2021; date of current version August 31, 2021. This work was supported in part by JSPS KAKENHI under Grant JP18K11163 and in part by CASIO SCIENCE PROMOTION FOUNDATION. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Yu Tsao. *(Corresponding author: Yukiya Hono.)*

The authors are with the Nagoya Institute of Technology, Nagoya 466-8555, Japan (e-mail: hono@sp.nitech.ac.jp; bonanza@sp.nitech.ac.jp; uratec@sp.nitech.ac.jp; nankaku@sp.nitech.ac.jp; tokuda@nitech.ac.jp).

Digital Object Identifier 10.1109/TASLP.2021.3104165

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

the singing voices; thus, a technique is needed for synthesizing singing voices with an appropriate pitch from arbitrary training data, including such out-of-tune phrases.

Recently, TTS research fields have utilized state-of-the-art systems with sequence-to-sequence (seq-to-seq) acoustic models and neural waveform generation models to achieve the same naturalness as human speech [18]. Seq-to-seq models with attention mechanisms directly map input text or phonetic sequences to the acoustic features without using an external duration model. Although some seq-to-seq models for end-to-end SVS have also been proposed [19]–[25], unlike TTS, a duration informed attention network is mainly used because of singing-specific backgrounds. For instance, the lengths of singing voices are generally longer than those of speech, and the amount of training data is insufficient. With the growth of deep learning techniques, statistical parametric SVS has been attracting attention for its various applications; however, these systems require high stability and controllability in terms of both acoustic parameters and alignments. Furthermore, since the synthesized singing voice strictly needs to be synchronized with the given musical score, it is not enough to apply the TTS-like end-to-end frameworks to the SVS systems. There is still a high demand for pipeline systems with an external time-lag model and a duration model from these perspectives.

This paper presents our DNN-based SVS system, "Sinsy." Our proposed system of this paper is an extension of our previous work [9]. All the components for synthesizing a singing voice from the analyzed score features are based on neural networks and incorporate novel techniques to better model a singing voice. Our system has a singing-specific design: 1) The combination strategy with the time-lag model and the duration model predicts phoneme boundaries under note length constraints statistically. 2) The acoustic model has improved pitch and vibrato modeling and a better training criterion for considering dynamic features. 3) The PeriodNet [26], a non-AR neural vocoder with more robustness of pitch, is adopted. 4) Automatic pitch correction techniques are incorporated into our SVS system to synthesize singing voices with the correct pitch. With these techniques, our proposed system can synthesize a high-fidelity singing voice waveform.

In the rest of this paper, Section II reviews the conventional SVS system. Section III describes the overview of our proposed SVS system. Section IV introduces the proposed techniques for modeling pitch and vibrato. Section V describes our proposed automatic pitch correction methods for DNN-based SVS. Section VI presents the experimental evaluations. Finally, Section VII concludes this paper.

II. Related Work

The usage of neural networks in SVS systems is similar to that in TTS systems. The simplest way to apply DNNs to TTS systems is to use an FFNN as a deep regression model to map a linguistic feature sequence obtained by text to an acoustic feature sequence extracted from speech [5]. A DNN-based SVS system also uses the DNN as the acoustic model; however, unlike TTS, feature vectors extracted from the musical score are used as the input instead of the linguistic feature. Architectures such as RNNs, CNNs, and AR structures are used as acoustic models for both TTS and SVS systems [10]–[13], [27]–[29].

The pitch of the synthesized singing voice must accurately follow the note pitch of the musical score even if the note pitch to be synthesized is outside the range of the training data. A pitch normalization technique has been proposed for F0 modeling in DNN-based SVS [8]. In this technique, the differences between the log F0 sequence extracted from waveforms and the note pitch are modeled. Recent studies [23], [24] introduced a residual connection between note pitch and generated log F0, which can be said to be the same approach to pitch normalization. Some systems [12], [17] utilize a data augmentation technique by pitch-shifting the training data. However, this technique requires more training time due to the increased amount of training data, and it is difficult to reproduce the voice characteristics and singing styles that change according to the tone. A post-processing strategy has also been proposed [13]. For this strategy, F0 should be modified for each voiced segment, which may generate a discontinuous F0 contour at the edge of the voiced segment.

Another unique characteristic of singing voices is that F0 includes periodic fluctuations due to vibrato. In our previous study [9], we separated the vibrato from the original F0 sequence in advance and modeled with sinusoidal parameters. This enables us to control the vibrato intensity and speed in the synthesis stage. Some systems do not use the decomposed approach, and the F0 sequence with the vibrato component is directly modeled using neural networks such as RNNs and AR models [10], [12], [13]. This approach can be expected to reproduce more complex vibrato shapes that are difficult to represent by sinusoidal parameters. However, there is a problem that the vibrato cannot be controlled during synthesis.

Since DNN-based SVS systems are data-driven approaches, the quality of the synthesized singing voice depends on the quality and quantity of the training data. Recent TTS systems often use over 20 hours of training data for a single-speaker model [18] and even hundreds of hours for a multi-speaker model [30]. However, unlike TTS, the amount of training data for SVS is often limited due to recording costs, annotation costs, and strict copyright issues in the music domain. Thus, in most cases, 1-2 hours of a singing voice corpus are used. Recently, there has been an attempt to utilize singing voice data mined from music websites as training data [31]. However, the pitch of this mined data is not always correct, despite pitch accuracy significantly impacting the quality of a singing voice. Even if the singing voice data is recorded for the training data, it may contain out-of-tune data due to various factors, such as a singer's skill, a song's tempo, and/or a melody's complexity. Although F0 contours can be manually modified after being extracted from training data, it is difficult to correct while maintaining a human-like F0 trajectory and is impractical in terms of editing cost. Therefore, there is demand for an automatic pitch correction technique in the SVS system.

Our system is based on our previous studies [8], [9] and incorporates improved singing-specific techniques. The skip connection of the note pitch improves acoustic feature estimation accuracy, particularly pitch. The differences-based vibrato modeling can achieve more accurate reproduction of vibrato

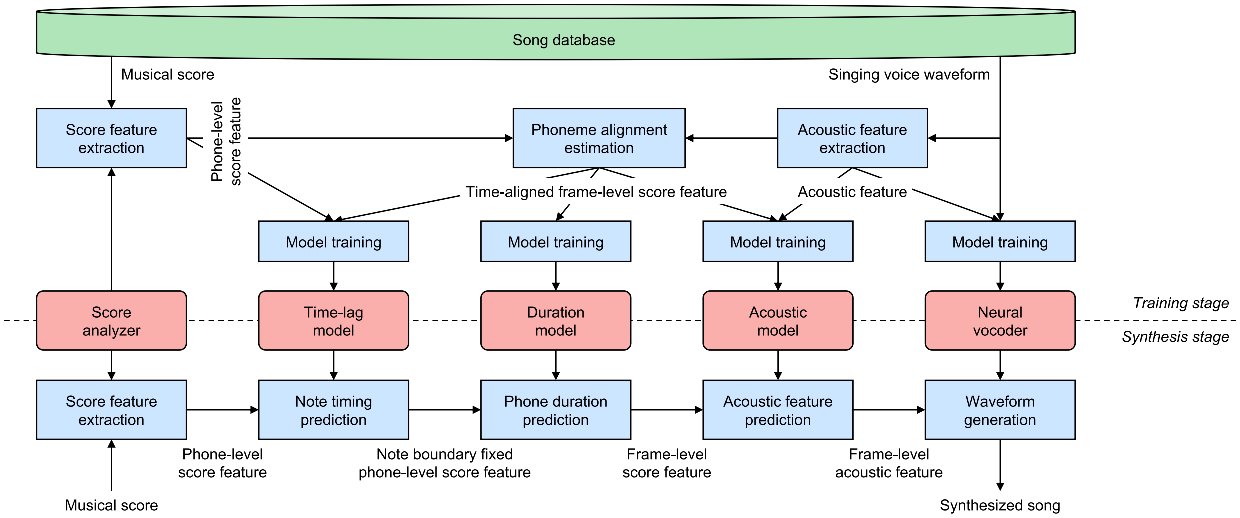


Fig. 1. Overview of the DNN-based SVS system, "Sinsy." Our system consists of a score analyzer and four neural network-based models. Singing voice can be synthesized from a musical score via these modules. Proposed singing-specific techniques, pitch normalization, vibrato modeling, and automatic pitch correction are incorporated in the acoustic model.

shape and can help control vibrato intensity. Automatic pitch correction techniques are also introduced into the SVS system. In addition, our system adopts the pitch robust neural vocoder PeriodNet. As a result, our system can synthesize a more natural singing voice that follows a given score more accurately.

III. DNN-Based SVS System

A. Overview

Figure 1 overviews our proposed DNN-based SVS system, "Sinsy." This system consists of several models: 1) a time-lag model to predict the start timing of the notes, 2) a duration model to predict phoneme durations in each note, 3) an acoustic model to generate acoustic features based on the predicted phoneme timing, and 4) a vocoder to synthesize waveforms from generated acoustic features. These models are based on the neural network.

Our system is composed of training and synthesis parts. In the training part, score and acoustic features are extracted by score analysis and vocoder encoding, then each model is trained. The score feature contains musical score information (e.g., lyrics, note keys, note lengths, tempo, dynamics, and slur), and the acoustic feature contains spectrum (e.g., mel-cepstral coefficients) and excitation parameters (e.g., F0). Time-aligned score features are needed to train the time-lag model, the duration model, and the acoustic model. Therefore, pre-trained hidden semi-Markov models (HSMMs) are used to estimate phoneme alignments [32].

In the synthesis part, first, the score features are extracted from the musical score to be synthesized. The time-lag model predicts each note's start timing, and the duration model predicts the phoneme duration in each note under the constraints of the note boundaries determined by the time-lag model. The frame-level score feature sequence is obtained using these predicted boundaries and then fed into the acoustic model to predict the acoustic feature sequence. Finally, the neural vocoder synthesizes a singing voice waveform.

Sinsy is a system for synthesizing singing voices from music scores. We adopt MusicXML [33] for representing musical scores that include lyrics. The score analyzer extracts musical contexts from the input musical score and encodes them into the categorical and numerical features that are easy for neural networks to handle. We use singing specific rich contexts, which are designed in our previous works [3].

B. Acoustic Model

In the SVS, the DNN-based acoustic model represents the mapping function from the score feature sequences to the acoustic feature sequences. There is a correlation between the spectrum and the excitation parameters in the acoustic feature (e.g., mel-cepstral coefficients and F0). Some studies have utilized a cascade structure to model this correlation [12]. In this work, a single neural network is used to model both spectrum and excitation parameters simultaneously, assuming that the correlation between them can be expressed inside the neural network.

The sequence of acoustic feature vectors $\boldsymbol{c}$ can be written in vector forms as follows:

$$\boldsymbol{c} = [\boldsymbol{c}_1^\top, \ldots, \boldsymbol{c}_t^\top, \ldots, \boldsymbol{c}_T^\top]^\top, \tag{1}$$

where $\boldsymbol{c}_t$ is a D-dimensional static feature vector that can be represented by $\boldsymbol{c}_t = [c_t(1), c_t(2), \ldots, c_t(D)]^\top$, and T is the number of frames included in a song. The optimal static feature vector sequence $\hat{\boldsymbol{c}}$ is given by

$$\hat{\boldsymbol{c}} = \arg\max_{\boldsymbol{c}} \mathcal{N}(\boldsymbol{c} \mid \bar{\boldsymbol{c}}, \boldsymbol{\Sigma}^{(\boldsymbol{c})}), \tag{2}$$

where $\mathcal{N}(\cdot \mid \bar{\boldsymbol{c}}, \boldsymbol{\Sigma}^{(\boldsymbol{c})})$ denotes the Gaussian distribution with a mean vector $\bar{\boldsymbol{c}}$ and a covariance matrix $\boldsymbol{\Sigma}^{(\boldsymbol{c})}$. In the SVS system, $\bar{\boldsymbol{c}}$ is obtained by feeding the score feature vector sequence into a trained neural network. A covariance matrix $\boldsymbol{\Sigma}^{(\boldsymbol{c})}$ is given by

$$\boldsymbol{\Sigma}^{(\boldsymbol{c})} = \mathrm{diag}[\boldsymbol{\Sigma}_1^{(\boldsymbol{c})}, \ldots, \boldsymbol{\Sigma}_t^{(\boldsymbol{c})}, \ldots, \boldsymbol{\Sigma}_T^{(\boldsymbol{c})}]. \tag{3}$$

In the DNN-based SVS, $\boldsymbol{\Sigma}^{(\boldsymbol{c})}$ is usually independent of score features; thus, $\boldsymbol{\Sigma}^{(\boldsymbol{c})}$ is a globally tied covariance matrix. Training of the DNN aims to maximize the likelihood function $\mathcal{L}^{(\mathrm{s})}$ given by

$$\mathcal{L}^{(\mathrm{s})} = \mathcal{N}(\boldsymbol{c} \mid \bar{\boldsymbol{c}}, \boldsymbol{\Sigma}^{(\boldsymbol{c})}). \tag{4}$$

Since a singing voice includes long tones, parameter discontinuity degrades the quality of the synthesized singing voice. Our previous studies [8], [9], [16] used the dynamic features to avoid this. The acoustic feature sequence of static and their dynamic feature vectors[1] $\boldsymbol{o}$ can be written in vector forms as follows:

$$\boldsymbol{o} = [\boldsymbol{o}_1^\top, \ldots, \boldsymbol{o}_t^\top, \ldots, \boldsymbol{o}_T^\top]^\top, \tag{5}$$

where $\boldsymbol{o}_t$ consists of the static and the dynamic feature vectors $\boldsymbol{o}_t = [\boldsymbol{c}_t^\top, \Delta^{(1)}\boldsymbol{c}_t^\top, \Delta^{(2)}\boldsymbol{c}_t^\top]^\top$. Relation between $\boldsymbol{o}$ and $\boldsymbol{c}$ can be represented by $\boldsymbol{o} = \boldsymbol{W}\boldsymbol{c}$, where $\boldsymbol{W}$ is a window matrix that extends $\boldsymbol{c}$ to $\boldsymbol{o}$. An acoustic model is trained by maximizing the following objective function as

$$\mathcal{L}^{(\mathrm{d})} = \mathcal{N}(\boldsymbol{o} \mid \bar{\boldsymbol{o}}, \boldsymbol{\Sigma}^{(\boldsymbol{o})}), \tag{6}$$

where $\bar{\boldsymbol{o}}$, $\boldsymbol{\Sigma}^{(\boldsymbol{o})}$ are a mean vector and a global tied covariance matrix that include the elements that correspond to dynamic features. The optimal static feature vector sequence $\hat{\boldsymbol{c}}^{(\mathrm{d})}$ is obtained from considering dynamic features by using the parameter generation algorithm [34] as follows:

$$\hat{\boldsymbol{c}}^{(\mathrm{d})} = \arg\max_{\boldsymbol{c}} \mathcal{N}(\boldsymbol{o} \mid \bar{\boldsymbol{o}}, \boldsymbol{\Sigma}^{(\boldsymbol{o})}) = \arg\max_{\boldsymbol{c}} \mathcal{N}(\boldsymbol{W}\boldsymbol{c} \mid \bar{\boldsymbol{o}}, \boldsymbol{\Sigma}^{(\boldsymbol{o})}). \tag{7}$$

Although the parameter generation algorithm can generate a smooth acoustic feature sequence, the computational cost at the synthesis stage increases. A recent study [11] introduced a different approach that considers dynamic features only during training. In this approach, the objective function that considers the dynamic features can be written as

$$\mathcal{L} = \mathcal{N}(\boldsymbol{o} \mid \boldsymbol{W}\bar{\boldsymbol{c}}, \boldsymbol{\Sigma}^{(\boldsymbol{o})}). \tag{8}$$

Since the output of the DNN contains only static feature vector $\bar{\boldsymbol{c}}$, the optimal static feature sequence can be obtained by (2) without the parameter generation algorithm in the synthesis stage. The average length of a singing voice is longer than that of speech, and the generation time is sometimes a problem. Thus, we utilize this approach for our system to generate a smooth parameter sequence without increased computational costs in the synthesis stage.

C. Time-Lag Model and Duration Model

In SVS, the phoneme duration should be determined from the note length of the musical score since a singing voice is synthesized based on the tempo and rhythm of the music. However, as Fig. 2 shows, humans generally tend to begin to utter consonants earlier than the absolute musical note onset timing. In addition, note timing may be slightly advanced or delayed as part of an individual's singing technique, so the timing varies depending on the singer. In this paper, we call the timing fluctuations caused by these factors "time-lag" and model them.

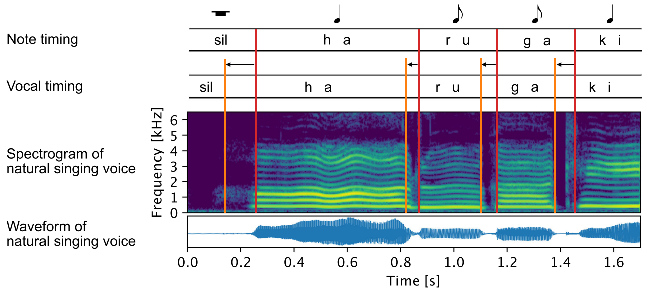


Fig. 2. Example of time-lag.

Two separated neural networks are used to model the time-lags and the phoneme durations. We define a time-lag as the difference between the note timing of the musical score and the actual timing of a reference phoneme within a note. To consider the time-lag in this work, we use the first vowel or silence for rests in each note as the reference phoneme instead of the first phoneme in the note. This reference phoneme shifting is based on the tendency of humans to sing so that the vowel onset timing is closest to the note timing in the score, and preliminary experiments have confirmed its effectiveness. Note that no additional annotation is needed in the training stage to train the time-lag and duration models because the actual phoneme timing of the singing voice can be obtained by a forced alignment using a well-trained HSMM [32].

In the synthesis stage, first, the time-lag of each note is predicted using a trained time-lag model. The sequences of note lengths obtained from the given musical score and predicted time-lags $\boldsymbol{L}$, $\hat{\boldsymbol{g}}$ can be written as follows:

$$\boldsymbol{L} = [L_1, \ldots, L_n, \ldots, L_N], \tag{9}$$

$$\hat{\boldsymbol{g}} = [\hat{g}_1, \ldots, \hat{g}_n, \ldots, \hat{g}_N], \tag{10}$$

where N is the number of notes included in a song. Note that $\hat{g}_1$ is always zero since there is no need to shift the first note boundary. Each adjusted note length $\hat{L}_n$ is obtained by

$$\hat{L}_n = \begin{cases} L_n - \hat{g}_n + \hat{g}_{n+1}, & (n < N) \\ L_n - \hat{g}_n. & (n = N) \end{cases} \tag{11}$$

Next, the phoneme durations are predicted by a trained phoneme duration model and are normalized on a note-by-note basis so that the sums of the phoneme durations within each note match the adjusted note lengths. The duration of the k-th phoneme in the n-th note is determined as follows:

$$\hat{d}_{nk} = \hat{L}_n \cdot \mu_{nk} \Big/ \sum_{k=1}^{K_n} \mu_{nk}, \tag{12}$$

where K_n is the number of phonemes in the n-th note, and μ_{nk} is the output value of the DNN-based duration model at the k-th phoneme in the n-th note.

The phoneme duration of synthesized songs can be obtained by the above strategy, considering the time-lag by the neural network. However, the distribution of phoneme durations differs greatly depending on the type of phonemes, such as consonants, vowels, and breaths. For example, durations of vowels vary greatly depending on the note length, while those of consonants

[1] We use velocity and acceleration features as dynamic features.

and breaths are barely affected by note length. Note that the breath phoneme corresponds to the breath mark in the musical score, and its duration needs to be predicted because the actual duration cannot be obtained from the musical score. Thus, it is not appropriate to fit the phoneme duration to the note length constraint by uniformly multiplying all the phonemes within a note by a constant as in (12).

Therefore, we statistically estimate the adjusted phoneme duration by considering the variance of the phoneme duration. This approach is based on constrained maximum likelihood estimation. We use a mixture density network (MDN) [35] to model the phoneme duration distribution. Note that an MDN with one mixture component is used for simplification. We assume that a single-mixture MDN has sufficient ability to represent the phoneme duration distribution.

The optimal phoneme duration sequence of n-th note $\hat{\boldsymbol{d}}_n^{(\mathrm{ML})}$ is given by

$$\hat{\boldsymbol{d}}_n^{(\mathrm{ML})} = \arg\max_{\boldsymbol{d}_n} \sum_{k=1}^{K_n} \log \mathcal{N}(d_{nk} \mid \mu_{nk}, \sigma_{nk}^2), \tag{13}$$

subject to

$$\sum_{k=1}^{K_n} \hat{d}_{nk}^{(\mathrm{ML})} = \hat{L}_n, \tag{14}$$

where μ_{nk}, σ_{nk}^2 denote the mean and the variance of the k-th phoneme duration in the n-th note, and these are obtained from a trained MDN.

To obtain optimal phoneme durations under the constraint condition in (14), we use the Lagrange multiplier method as follows:

$$\begin{aligned} F(d_{nk}, \rho_n) = & \sum_{k=1}^{K_n} \log \mathcal{N}(d_{nk} \mid \mu_{nk}, \sigma_{nk}^2) \\ & + \rho_n \left(\sum_{k=1}^{K_n} d_{nk} - \hat{L}_n \right), \end{aligned} \tag{15}$$

where ρ_n denotes the Lagrange multiplier. Hence, the optimal duration of the k-th phoneme in the n-th note is obtained by

$$\hat{d}_{nk}^{(\mathrm{ML})} = \mu_{nk} + \rho_n \sigma_{nk}^2, \tag{16}$$

where ρ_n is given by

$$\rho_n = \left(\hat{L}_n - \sum_{k=1}^{K_n} \mu_{nk} \right) \Big/ \sum_{k=1}^{K_n} \sigma_{nk}^2. \tag{17}$$

Our system assumes that each note is assigned one or more phonemes. The note with the long sound symbol "—" is assigned the appropriate phoneme based on the previous note's lyric, using language-specific heuristic rules at the score analyzing process. For example, in Japanese, the phoneme of a note with a long vowel symbol is obtained by duplicating a vowel in a previous note. In English, the syllable nucleus allocated to the previous note is duplicated, and the phoneme after the duplicated syllable nucleus is shifted to the current note. Since consecutive diphthongs due to duplication may degrade the continuity of a singing voice, we defined the duplication rules for diphthongs in our previous work [36].

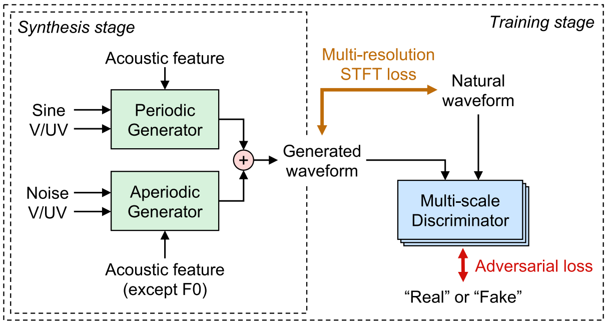


Fig. 3. The overview of PeriodNet parallel model.

D. Neural Vocoder

We use the PeriodNet [26] as a vocoder to generate singing voice waveforms from acoustic feature sequences. PeriodNet is a non-autoregressive neural vocoder with a structure that separates periodic and aperiodic components. PeriodNet consists of two sub-generators connected in parallel or series and models a speech waveform based on the following two assumptions. The first one is that the speech waveform can be represented as the sum of the periodic and aperiodic waveform. The second one is that periodic and aperiodic waveforms of speech can be easily generated from an explicit periodic signal with autocorrelation (such as sinusoidal signal) and an explicit aperiodic signal without one (such as noise), respectively. The parallel or series structure helps to model speech waveform with periodic and aperiodic components more appropriately and improves the robustness of the input acoustic features, especially F0. SVS systems require the ability to generate high-quality singing voice waveforms even if the input pitch is outside the range of training data. PeriodNet is highly suitable for the neural vocoder in SVS systems because it has superior reproducibility of accurate pitch and breath sounds.

This work adopts the parallel model (PM2 in [26]), as shown in Fig. 3. A periodic generator takes an explicit periodic signal and an aperiodic generator that takes an aperiodic signal. A sample-level voiced/unvoiced (V/UV) signal is also fed into both generators, and the periodic signal can be generated from F0 predicted by the acoustic model in the synthesis stage. Both periodic and aperiodic generators adopt WaveNet-like architecture, which has a stack of non-causal convolution layers, and are conditioned on the acoustic feature. To obtain the robustness of pitch, we exclude the F0 sequence from the condition of the aperiodic generator. PeriodNet is trained by an adversarial training framework using a multi-scale discriminator along with a multi-resolution short-time Fourier transform (STFT) auxiliary loss.

IV. Accurate and Expressive Pitch Modeling for the DNN-Based SVS System

This section describes the singing-specific techniques of accurately modeling pitch, including vibrato for the DNN-based SVS.

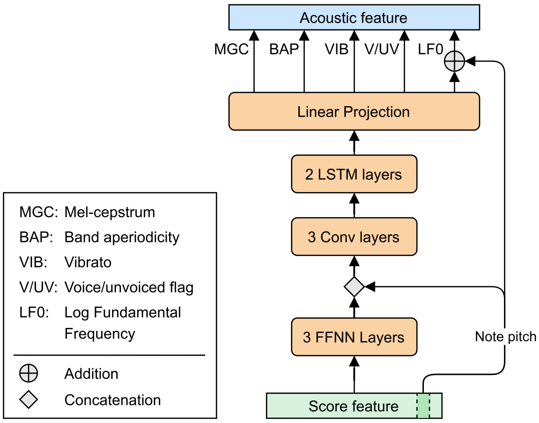


Fig. 4. Architecture of acoustic model in our system.

A. Pitch Normalization

The corpus-based nature of statistical parametric SVS approaches makes their performance highly dependent on the training data. It is challenging to express contextual factors that rarely appear in training data. Hence, DNN-based SVS systems should be trained using a database that contains various contextual factors to synthesize high-quality singing voices. In particular, since the prediction accuracy of F0 significantly affects the quality of the synthesized singing voice, the pitch must be covered correctly. However, it is almost impossible to cover all possible contextual factors because a singing voice is affected by a large number of factors, such as lyrics, key, dynamics, note duration, and note pitch.

A musical note-level pitch normalization technique for DNN-based SVS systems was proposed in our previous work [8] to address the aforementioned problem. In that technique, the F0 sequence extracted from the natural waveforms is not modeled directly but as a difference from the note pitch determined by musical notes in the score. Therefore, the acoustic model only needs to predict the human bias from the note pitch. This technique enables DNN-based SVS systems to synthesize singing voices that contain arbitrary pitches, including unseen ones. However, modeling the difference of F0 remains a challenge because F0 in unvoiced frames and the note pitch in the musical rest are unmeasurable. Thus, all unvoiced frames and musical rests in the musical score are linearly interpolated and modeled as voiced frames.

Figure 4 shows the architecture of the acoustic model with the pitch normalization technique. In the pitch normalization technique, a predicted log F0 sequence $\bar{\boldsymbol{c}}^{(\mathrm{F0})}$ is represented by using the note pitch sequence $\boldsymbol{p} = [p_1, p_2, \ldots, p_T]^\top$ and the output mean parameter sequence $\boldsymbol{\mu} = [\mu_1, \mu_2, \ldots, \mu_T]^\top$ as follows:

$$\bar{\boldsymbol{c}}^{(\mathrm{F0})} = \boldsymbol{p} + \boldsymbol{\mu}. \tag{18}$$

Note that we use log F0, the log scale of F0. The note pitch sequence $\boldsymbol{p}$ to be added can be obtained from the input score features sequence.

A note pitch transition greatly influences the F0 trajectory. Therefore, we add a skip connection between the input note pitch and a hidden layer of the acoustic model to deliver the note pitch inside the acoustic model, motivated by [11]. This helps transmit the note pitch information efficiently and predict the residual component between log F0 and the note pitch.

B. Vibrato Model

Generating an expressive F0 contour for a singing voice is also challenging. Vibrato is one of the important singing techniques, and the timing and intensity of vibrato vary from singer to singer. Thus, it should be modeled even though it cannot be explicitly described in the musical score. Some acoustic models with a recurrent or AR structure can model F0 with vibrato directly [10], [12], [13]. However, it does not enable explicit control of vibrato. Here, we assume that vibrato is a periodic fluctuation of the F0 contour and introduce two explicit vibrato modeling methods.

1) Sine-Based Vibrato Modeling: One method for modeling vibrato is to express periodic fluctuations with sinusoidal parameters [9]. The vibrato $v(\cdot)$ of the t frame in the i-th vibrato section $[t_i^{(s)}, t_i^{(e)}]$ can be defined as

$$v\big(m_a(t), m_f(t), i\big) = m_a(t) \sin\Big(2\pi m_f(t) f_s \big(t - t_i^{(s)}\big)\Big), \tag{19}$$

where $m_a(t)$ is the amplitude of vibrato in cents, $m_f(t)$ is the frequency of vibrato in Hz, and f_s is the frame shift in seconds. These parameters can be obtained from the original F0 sequence with an estimation algorithm [37]. In this work, the vibrato amplitude and frequency parameters are extracted based on the intersection points between the original F0 and a median-smoothed F0. Two-dimensional parameters, $m_a(t)$ and $m_f(t)$, are added to the acoustic feature vector and modeled by a DNN along with the spectral and excitation parameters. The vibrato parameters are unobserved outside the vibrato sections. Thus, these parameters are interpolated in the same manner as the F0 sequence and are modeled along with an additional binary flag to determine the vibrato/non-vibrato frames.

2) Difference-Based Vibrato Modeling: In this paper, we propose another method of modeling the vibrato component separated from the F0. The vibrato component is defined as the difference between the original F0 sequence and the smoothed F0 sequence. This difference-based vibrato component is a one-dimensional continuous feature and is directly modeled by the acoustic model that can model time series data such as RNN without an extra binary flag representing vibrato/non-vibrato frames. This method has the advantage of generating more complex vibrato shapes given no assumption of the vibrato shape. In particular, smoother vibrato can be obtained because the start and end of the vibrato are determined by the value of the difference rather than the binary flag. Furthermore, it is possible to control the vibrato intensity by changing the difference values, unlike the method of not separating the vibrato component explicitly.

V. Automatic Pitch Correction

The singing voice becomes out of tune when the pitch of a singing voice deviates from that of the musical score. Therefore, we introduce two automatic correction techniques to prevent synthesized singing voices from becoming out of tune: prior distribution of pitch and pseudo-note pitch.

A. Prior Distribution of Pitch

In pitch normalization, the difference between log F0 and note pitch is modeled by (18). Here, we correct the out-of-tune phrases by giving a prior distribution to the pitch normalization training, assuming that the difference follows a zero-mean Gaussian distribution.

The prior distribution of pitch is given as

$$P(\boldsymbol{\mu}) = \mathcal{N}(\boldsymbol{\mu} \mid \boldsymbol{v}, \boldsymbol{S}), \tag{20}$$

where $\boldsymbol{v}$ and $\boldsymbol{S}$ correspond to the mean and the variance of prior Gaussian distribution. Note that $\boldsymbol{\mu}$ does not contain a vibrato component because the pitch should be corrected while maintaining vibrato. Here $\boldsymbol{v} = \boldsymbol{0}$ since the prior distribution we assume always represents the difference between the F0 extracted from the natural waveforms and note pitch in the musical score. The variance $\boldsymbol{S}$ works as a parameter that controls the intensity of pitch correction. The smaller element of $\boldsymbol{S}$ is, the stronger the out-of-tune pitch is corrected. In this work, we assume that the variance always takes the fixed value σ_p^2. In the training part, an objective function in terms of F0 is defined as

$$\mathcal{L}^{(\mathrm{F0})} = \mathcal{N}(\boldsymbol{o}^{(\mathrm{F0})} \mid \boldsymbol{W}(\boldsymbol{p}+\boldsymbol{\mu}), \boldsymbol{\Sigma}^{(\boldsymbol{o}^{(\mathrm{F0})})})\,\mathcal{N}(\boldsymbol{\mu} \mid \boldsymbol{v}, \boldsymbol{S}), \tag{21}$$

where $\boldsymbol{o}^{(\mathrm{F0})}$ and $\boldsymbol{\Sigma}^{(\boldsymbol{o}^{(\mathrm{F0})})}$ are a sequence of feature vectors and a covariance matrix in terms of log F0 and their dynamic features.

In unique phenomena of a singing voice such as overshooting[2] and preparation,[3] F0 contours are deflected from the target notes before or after a note changes. Thus, it is not always optimal to correct the pitch with the same intensity at all frames. Furthermore, it sounds rather unnatural if the difference between the F0 and the note pitch becomes too small. Therefore, we introduce a dynamic weight vector $\boldsymbol{w} = [w_1, w_2, \ldots, w_T]$ whose values are changed based on the note position into (21):

$$\begin{aligned}\mathcal{L}^{(\mathrm{F0})} &= \mathcal{N}(\boldsymbol{o}^{(\mathrm{F0})} \mid \boldsymbol{W}(\boldsymbol{p}+\boldsymbol{\mu}), \boldsymbol{\Sigma}^{(\boldsymbol{o}^{(\mathrm{F0})})})\,\mathcal{N}(\boldsymbol{\mu} \mid \boldsymbol{v}, \boldsymbol{S})^{\boldsymbol{w}} \\ &= \mathcal{N}(\boldsymbol{o}^{(\mathrm{F0})} \mid \boldsymbol{W}(\boldsymbol{p}+\boldsymbol{\mu}), \boldsymbol{\Sigma}^{(\boldsymbol{o}^{(\mathrm{F0})})}) \prod_{t=1}^{T} \mathcal{N}(\mu_t \mid 0, \sigma_p^2)^{w_t},\end{aligned} \tag{22}$$

where T denotes the number of frames. The values of the weight vector increase or decrease at the beginning or end of notes, as shown in Fig. 5. In this paper, the maximum of $\boldsymbol{w}$ takes 0.5, and the width of increasing and decreasing is set to 25 frames.

B. Pseudo-Note Pitch

One cause of the phrases being out of tune is that there is a difference between an assumed note pitch by the singer and the correct note pitch. By training the acoustic model using the pseudo-note pitch that takes these differences into account, the singing voice should be synthesized with the correct pitch by using the original note pitch during synthesis. We propose two different approaches to obtaining the pseudo-note pitch.

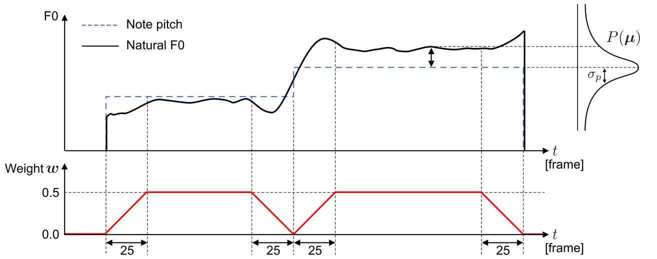


Fig. 5. Automatic pitch correction by prior distribution with dynamic weight vector.

1) Heuristic Pseudo-Note Pitch: The pseudo-note pitch is heuristically calculated from the flat part of the F0 sequence to express the singer-assumed note pitch during singing. In the training stage, the pitch normalization technique is applied using this pseudo-note pitch. In the synthesis stage, it is possible to synthesize the singing voice with the correct pitch by using the original note pitch instead of the pseudo-note pitch. An example of the heuristic pseudo-note pitch will be shown in Section VI-D.

2) Pitch Bias-Based Pseudo-Note Pitch: We introduce additional trainable parameters, note-level pitch bias, to represent the difference between the singer-assumed note pitch and the correct note pitch. In this approach, the pseudo-note pitch is defined as the sum of the original note pitch given by musical score and the note-level pitch bias. The pitch bias should absorb the average pitch shift in each note seen in the out-of-tune phrases. In the training stage, the predicted log F0 sequence is defined as

$$\bar{\boldsymbol{c}}^{(\mathrm{F0})} = \boldsymbol{p} + \boldsymbol{\mu} + \boldsymbol{b}, \tag{23}$$

where $\boldsymbol{b} = [b_1, b_2, \ldots, b_T]^\top$ is a sequence of the trainable pitch bias. The bias parameters are assigned to each note (except for musical rests), and $\boldsymbol{b}$ is obtained by duplicating each parameter according to a corresponding note length. Note that the bias values in musical rests are obtained by linearly interpolating them in the same manner as the original note pitch in pitch normalization described in Section IV-A. Hence, the total number of trainable bias parameters is equal to the total number of musical notes in the training data. These bias parameters can be trained using a back-propagation in the same fashion as the other model parameters without any additional loss function. In the synthesis stage, the F0 with the correct pitch can be generated by fixing the bias value to zero.

VI. Experiments

This section evaluates the effectiveness of the proposed system in terms of the combination strategy with the time-lag model and the duration model, improved acoustic feature modeling, and automatic pitch correction techniques.

A. Experimental Conditions

In this experiment, 70 Japanese children's songs (total: 70 min) by a female singer were used. Sixty songs were used for training, and the rest were used for testing. Singing voice signals were sampled at 48 kHz and windowed with a 5-ms shift. The

[2] Overshooting is a pitch deflection exceeding the target note after a note change.

[3] Preparation is a pitch deflection in the direction opposite to a note change that can be seen just before the note change.

acoustic feature consisted of 0-th through 49-th mel-cepstral coefficients, log F0 value, 0-th through 24-th mel-cepstral analysis aperiodicity measures, and vibrato parameters. In order to reduce an error of F0 extraction, voting results from three F0 estimators were used as F0 of acoustic features [38]. Mel-cepstral coefficients were extracted from the smoothed spectrum analyzed by WORLD [39]. Two types of explicit vibrato parameters were used: 2-dimensional sinusoidal-based parameters that consisted of amplitude and frequency parameters and a 1-dimensional vibrato component that represented the difference between the original log F0 and the smoothed log F0. The score feature was obtained by analyzing the musical score, and the contextual factor we used followed our previous work [3].

Five-state, left-to-right, no-skip HSMMs were used to obtain the time alignment of the score features and the acoustic features for training the DNN-based time-lag, duration, and acoustic models. The decision tree-based context clustering technique was separately applied to distributions for the spectrum, excitation, aperiodicity, and state duration. The spectrum and aperiodicity stream were modeled with single multivariate Gaussian distributions. The excitation stream was modeled with multi-space probability distribution HSMMs (MSD-HSMMs) [40] that each consisted of a Gaussian distribution for "voiced" frames and a discrete distribution for "unvoiced" frames. The duration stream was modeled with single Gaussian distributions. The minimum description length (MDL) criterion was employed to control the size of the decision trees for context clustering [41].

B. Objective Evaluation of Time-Lag Modeling and Duration Modeling

An objective evaluation experiment was conducted to compare the prediction accuracy of note timing and phoneme duration. In this experiment, the following three methods were compared.

- **DT**: Conventional method of predicting phoneme boundaries using the decision tree-based clustered context-dependent time-lag model and duration model in an HMM-based SVS system [42].
- **DNN**: Time-lag and phoneme duration were modeled by DNNs, and final phoneme boundaries were determined according to (12).
- **DNN+ML**: Time-lag and phoneme duration were modeled by DNN and single MDN, and final phoneme boundaries were determined using constrained maximum likelihood estimation with (16).

In **DT**, the sizes of the decision trees were determined by the MDL criterion. In **DNN** and **DNN+ML**, the input feature of DNN was an 824-dimensional feature vector consisting of 734 binary features for categorical linguistic contexts (e.g., the current phoneme identity) and 90 numerical features for numerical contexts (e.g., the number of phonemes in the current syllable). In **DNN**, the outputs of the time-lag model and duration model are one-dimensional numerical values. In **DNN+ML**, the time-lag model outputs one-dimensional numerical values, and the duration model outputs the mean and variance of the one-dimensional phoneme duration distribution. In **DNN** and **DNN+ML**, the architecture of the time-lag model was three hidden layers with 32 units per layer, and that of the duration model was three hidden layers with 256 units per layer. The sigmoid activation function was used in the hidden layers, and the linear activation function was used in the output layer. The weights of the DNNs and the MDN were initialized randomly, then the DNNs were optimized by minimizing the mean squared error, and the MDN was optimized by maximizing the likelihood. In the training phase, the Adam optimizer [43] was adopted for all neural networks.

TABLE I
Object Evaluation of Timing Prediction Accuracy

Method	DT	DNN	DNN+ML
Note duration-RMSE [frame]	15.78	**12.75**	**12.75**
Phoneme duration-RMSE [frame]	13.32	11.23	**10.94**
Note duration-CORR	0.9742	**0.9780**	**0.9780**
Phoneme duration-CORR	0.9719	0.9757	**0.9767**

The root mean square errors (RMSEs) of the note and the phoneme duration and Pearson correlations (CORRs) of the note and the phoneme duration were used as the objective evaluation metrics. Note that the phoneme boundaries of forced alignment obtained by trained HMMs were used as the correct phoneme boundaries.

The experimental results are listed in Table I. It can be seen that both **DNN** and **DNN+ML** performed better in predicting note and phoneme boundaries than **DT**. This result indicates the effectiveness of replacing the decision tree-based clustered models with DNN-based models. Also, comparing **DNN+ML** with **DNN** in terms of phoneme duration prediction accuracy, **DNN+ML** outperformed **DNN**. This suggests that constrained maximum likelihood estimation of note lengths with consideration of variances helps fit the phoneme durations.

C. Comparison of Acoustic Feature Modeling

Objective and subjective evaluations were conducted to compare the acoustic models in terms of pitch normalization, skip connection of the note pitch, vibrato modeling, and the training criterion. We used the seven systems shown in Table II.

The input feature vector for the acoustic models was an 844-dimensional feature vector with the 824-dimensional feature vector in Section VI-B and a 20-dimensional additional feature vector that included duration features. The output feature vector for the acoustic models consists of mel-cepstral coefficients, log F0 value, mel-cepstral analysis aperiodicity measures, vibrato parameters (except **System 5**), voiced/unvoiced binary value, and vibrato/non-vibrato binary value (only **System 4**). In **System 7**, the dynamic features (velocity and acceleration features) were also included in the output feature vector. A single network that modeled all acoustic features simultaneously was trained by using the Adam optimizer [43]. The architecture of the acoustic models was the stack of three fully connected layers with 2048 hidden ReLU units, three convolution blocks each containing a 1D convolutional layer with 1024 filters, batch normalization [44] and ReLU activations, two bidirectional LSTMs containing 512 units (256 in each direction), and a

TABLE II
RESULTS OF OBJECTIVE EVALUATION OF ACOUSTIC MODELS

System Index	System Details				MCD [dB]	F0-RMSE [cent]	F0+Vib-RMSE [cent]	F0-CORR	F0+Vib-CORR
	Pitch norm.[a]	Skip connect.[b]	Vibrato[c]	Criterion[d]					
System 1	✓	✓	Diff-based	$\mathcal{L}$	**5.423**	74.00	80.96	**0.9713**	**0.9647**
System 2			Diff-based	$\mathcal{L}$	5.644	264.07	265.71	0.6689	0.6677
System 3	✓		Diff-based	$\mathcal{L}$	5.627	79.80	86.69	0.9653	0.9585
System 4	✓	✓	Sine-based	$\mathcal{L}$	5.456	**72.91**	85.12	0.9712	0.9592
System 5	✓	✓	N/A	$\mathcal{L}$	5.439	-	**80.95**	-	0.9635
System 6	✓	✓	Diff-based	$\mathcal{L}^{(\mathrm{s})}$	5.462	73.49	82.00	0.9712	0.9636
System 7	✓	✓	Diff-based	$\mathcal{L}^{(\mathrm{d})}$	5.445	74.16	82.48	0.9709	0.9633

[a] Pitch normalization described in Section IV-A.
[b] Skip connection described in Section IV-A.
[c] "Sine-based" denotes sine-based vibrato modeling described in Section IV-B1, and "Diff-based" denotes the difference-based vibrato modeling described in Section IV-B2.
[d] Trainig criteria $\mathcal{L}$, $\mathcal{L}^{(\mathrm{s})}$, and $\mathcal{L}^{(\mathrm{d})}$ are given by (8), (4), and (6), respectively.

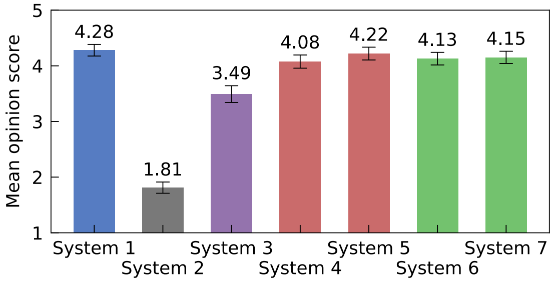


Fig. 6. Mean opinion scores of the seven SVS systems with 95% confidence intervals.

linear projection layer. The same pre-trained PeriodNet [26] neural vocoder was used to reconstruct singing waveforms from generated acoustic features in all systems.

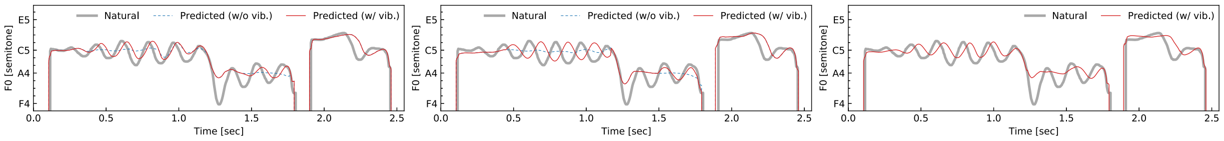
Mel-cepstral distortion (MCD) [dB], RMSEs of log F0 without/with vibrato (F0-RMSE and F0+Vib-RMSE) [cent], CORRs of log F0 without/with vibrato (F0-CORR and F0+Vib-CORR) were used to objectively evaluate the performance of systems. Phoneme durations from natural singing voices were used while performing the objective evaluation. Mean opinion score (MOS) tests were also conducted to subjectively evaluate the naturalness of synthesized waveforms. In the subjective evaluation, the phoneme durations were determined by **DNN+ML** in Section VI-B. The subjects were eleven Japanese students in our research group, and twelve phrases were chosen at random per method from the test set. The scale for the MOS test was 5 for "natural" and 1 for "poor." The demo songs can be found on the demo page [45].

Table II shows the objective evaluation results and Fig. 6 shows the subjective evaluation results. By comparing **System 2** and **System 3** in the objective and the subjective evaluations, we revealed that the pitch normalization technique is essential in modeling the pitches of singing voices. The synthesized singing voices in **System 2** were sometimes perceived as if they were sung following different note pitches because the generated F0 deviated from the target note pitches. Furthermore, **System 1** achieved a better score than **System 3** in terms of both the metrics of F0 and MCD, which also led to a good subjective evaluation score. This result suggests that it is helpful to transmit the note pitch information of the musical score to the inside of the model by using the skip connection because a singing voice is greatly affected by note pitch transition.

Comparing the methods of vibrato modeling, **System 1** and **System 5** outperformed **System 4** in terms of F0+Vib-RMSE, F0+Vib-CORR, and MOS value. The examples of generated F0 contours in each system are plotted in Fig. 7. As the figure shows, the F0 contours of **System 1** and **System 5** are closer to the natural F0 contour than that of **System 4**. Since **System 4** cannot reproduce the vibrato phase, the F0 contour deviates significantly from the natural F0 contour. This is a major factor in the deterioration of the objective evaluation. In addition, the start and end shapes of the vibrato of **System 1** and **System 5** are smoother than that of **System 4**, indicating the effectiveness of modeling the vibrato component by neural networks without using the sinusoidal parameters. A comparison of **System 1** and **System 5** shows no significant difference between them. In **System 1**, since F0 and vibrato are modeled separately, it is possible to change the vibrato intensity and introduce the pitch correction techniques described in Section V. Therefore, difference-based vibrato modeling is an effective method.

Finally, **System 1**, **System 6**, and **System 7** were compared. **System 1** was expected to enable more continuous and appropriate parameter generation without using explicit dynamic features during synthesis because this system was trained considering the dynamic features, but the effect was slight. Although **System 6** sometimes generated unstable singing voices, it was not a big problem in the subjective evaluation. Our acoustic model consisted of bidirectional LSTMs that can generate sufficiently continuous parameters without using dynamic features. Meanwhile, **System 7** may have caused parameter over-smoothing due to the parameter generation considering dynamic features explicitly.

In summary, **System 1** got a generally good objective score and achieved the best MOS value. These results indicate the effectiveness of the proposed system with pitch normalization, the skip connection of the note pitch, difference-based vibrato modeling, and the training criterion considering dynamic features.



(a) Difference-based vibrato modeling (**System 1**). (b) Sine-based vibrato modeling (**System 4**). (c) Not using explicit vibrato modeling (**System 5**).

Fig. 7. Generated F0 contours for test song. Predicted (w/o vib.) denotes the bare F0 contour without vibrato components and Predicted (w/ vib.) denotes final F0 contour with vibrato components. The frequency value of note A4 is 440 Hz in this paper.

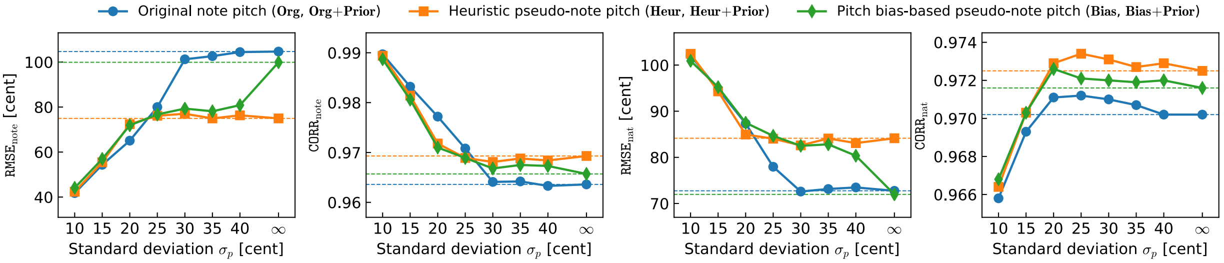


Fig. 8. Objective evaluation of comparing automatic pitch correction techniques. $\sigma_p = \infty$ denotes the systems not using prior in the training (**Org**, **Heur**, and **Bias**) and the dotted lines on each graph represent the objective evaluation values of these systems.

TABLE III
SYSTEMS FOR EVALUATION OF PITCH CORRECTION TECHNIQUES

Note pitch	w/o prior	w/ prior
Original note pitch	**Org**	**Org+Prior**
Heuristic pseudo-note pitch	**Heur**	**Heur+Prior**
Pitch bias-based pseudo-note pitch	**Bias**	**Bias+Prior**

D. Effectiveness of Automatic Pitch Correction Techniques

We also evaluated the effectiveness of the automatic pitch correction techniques using a different speaker's singing voice dataset. This dataset consisted of the same 70 songs used in the previous experiments but contained out-of-tune phrases. Other experimental conditions were the same as in Section VI-A. Time-lags and phoneme durations were modeled by **DNN+ML** in Section VI-B, and acoustic features were modeled by **System 1** in Section VI-C.

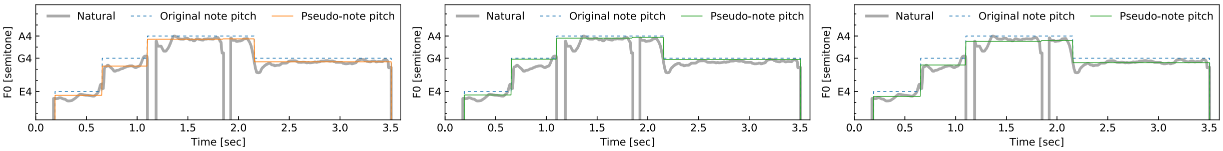
In this experiment, six systems were used for comparison as shown in Table III. Three types of note pitches were used in the training stage: the original note pitch given by the musical scores as a baseline, a heuristic pseudo-note pitch mentioned in Section V-B1, and a pitch bias-based pseudo-note pitch mentioned in Section V-B2. Note that original note pitches were always used during the synthesis stage.

1) Objective Evaluation: We compared the performance of the experimental systems objectively. In **Org+Prior**, **Heur+Prior**, and **Bias+Prior**, we compared seven different values of standard deviation σ_p [cent] in (22). We use four objective measures: the RMSE and CORR between the generated F0 and the F0 calculated from the note pitches ($\mathtt{RMSE}_{\rm note}$ and $\mathtt{CORR}_{\rm note}$), and those between the generated F0 and the F0 extracted from the natural waveform ($\mathtt{RMSE}_{\rm nat}$ and $\mathtt{CORR}_{\rm nat}$). RMSEs and CORRs are objective measures that represent how close the value and shape of the predicted sequence are to the target sequence. Note that if the target sequence includes out-of-tune phrases, it is not necessarily good that the $\mathtt{RMSE}_{\rm nat}$ achieves small and the $\mathtt{CORR}_{\rm nat}$ achieves high. A small $\mathtt{RMSE}_{\rm note}$ and high $\mathtt{CORR}_{\rm note}$ mean that the generated F0 is close to the stair-like correct note pitch.

Fig. 8 shows the results of the objective evaluations. In **Org+Prior**, by setting σ_p smaller than 30, $\mathtt{RMSE}_{\rm note}$ significantly decreased compared to **Org**. This result indicates that introducing prior distributions corrected the pitch. On the other hand, the $\mathtt{CORR}_{\rm note}$ went higher at the same time as the improvement of $\mathtt{RMSE}_{\rm note}$, indicating that the shape of the generated F0 in **Org+Prior** tends to be a stair-like note pitch. The results of $\mathtt{RMSE}_{\rm nat}$ at $\sigma_p < 30$ in **Org+Prior** also got worse because the test data for evaluation also included out-of-tune phrases. When σ_p was set in the range of 20 to 35, $\mathtt{CORR}_{\rm nat}$ in **Org+Prior** had a better score than that in **Org**. This is because it could suppress the generation of unstable pitch fluctuations, which can be seen in the out-of-tune training data.

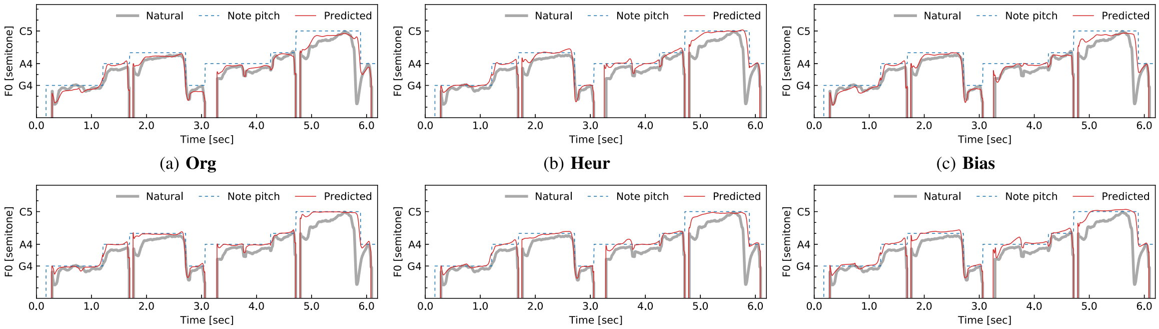
Compared **Heur** with **Org**, $\mathtt{RMSE}_{\rm note}$ was greatly improved by introducing a heuristic pseudo-note pitch. In addition, when combined with prior distributions in **Heur+Prior**, $\mathtt{RMSE}_{\rm note}$ and $\mathtt{CORR}_{\rm note}$ did not change if $\sigma_p = 20$ or more. These results show that the heuristic pseudo pitch is effective for pitch correction. Furthermore, **Heur** and **Heur+Prior** achieve higher $\mathtt{CORR}_{\rm nat}$ than **Org** and **Org+Prior**, indicating that the deviation between the original F0 and note pitch in **Heur** and **Heur+Prior** becomes smaller by using the pseudo-note pitch, thus avoiding forced pitch correction.

The results of **Bias+Prior** show a similar trend to those of **Heur+Prior**. In contrast, the objective results of **Bias** were not as good as those of **Heur**. The examples of heuristic pseudo-note pitch and pitch bias-based pseudo-note pitches in both **Bias** and **Bias+Prior** are shown in Fig. 9. The pitch bias-based pseudo-note pitch of **Bias+Prior** in Fig. 9(c) is similar to the heuristic pseudo-note pitch in Fig. 9(a). However, the pitch bias-based one sometimes yields inappropriate results, as shown at around 1.5 seconds in Fig. 9(c). This result is because the pitch bias is



(a) Heuristic pseudo-note pitch (b) Pitch bias-based pseudo-note pitch in case **Bias**. (c) Pitch bias-based pseudo-note pitch in case **Bias+Prior** ($\sigma_p = 20$).

Fig. 9. Examples of pseudo-note pitch in proposed automatic pitch correction.



(a) **Org** (b) **Heur** (c) **Bias**

(d) **Org+Prior** ($\sigma_p = 20$) (e) **Heur+Prior** ($\sigma_p = 20$) (f) **Bias+Prior** ($\sigma_p = 20$)

Fig. 10. Generated F0 contours for one test phrase.

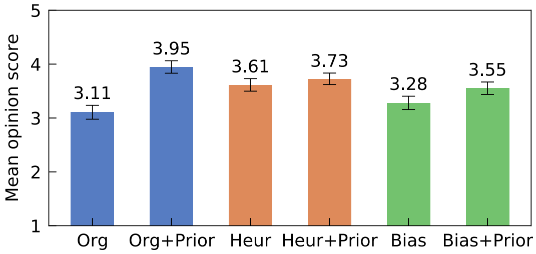


Fig. 11. Results of MOS test comparing automatic pitch correction techniques with 95% confidence intervals. In **Org+Prior**, **Heur+Prior**, and **Bias+Prior**, σ_p were set to 20.

determined by considering the entire note and is influenced by the singing expression that changes F0 within the note, such as bending and hiccups. On the other hand, the pitch bias-based pseudo-note pitch of **Bias** in Fig. 9(b) is close to the original note pitch in the musical score. This result indicates that there is ambiguity as to whether the average pitch shift of F0 at each note should be represented by the outputs of the acoustic model or the pitch biases.

2) Subjective Evaluation: We conducted the MOS test to evaluate the overall naturalness. Subjects were instructed to give high score values to test phrases that were naturally pitch-corrected and not out of tune. We compared the six systems listed in Table III. The σ_p in **Org+Prior**, **Heur+Prior**, and **Bias+Prior** were set to 20.

The generated F0 contours are plotted in Fig. 10, and the results of the MOS test are plotted in Fig. 11. **Org+Prior** achieved a higher MOS value than **Org**. This result confirmed the effectiveness of using prior distribution. **Heur** and **Heur+Prior** outperformed **Org**, but there was little difference between **Heur** and **Heur+Prior**. Since the values of the F0 differences modeled by the acoustic model were smaller on average by introducing the heuristic pseudo pitch, the prior distribution seemed to have a limited effect. On the other hand, although **Bias** and **Bias+Prior** also outperformed **Org**, these did not reach **Heur** and **Heur+Prior**, respectively. As shown in Fig. 10, **Bias** and **Bias+Prior** were more likely to generate an F0 that slightly deviates from the correct note pitch, compared with **Heur** and **Heur+Prior**. This result indicates that it is not easy to automatically obtain pseudo-note pitches from F0, which includes fluctuations such as bending and hiccups, and the heuristic method of obtaining pseudo-note pitches is powerful and effective. Overall, **Org+Prior** achieved the best MOS value even though **Org+Prior** showed a worse $\mathrm{CORR}_{\mathrm{nat}}$ than **Heur+Prior** and **Bias+Prior**. The objective evaluation results and Fig. 10(d) show that the F0 contour generated by **Org+Prior** was the most similar to a stair-like F0 contour, and the output of the acoustic model in terms of the F0 was strongly corrected by the prior distribution in the training stage. While fine fluctuations of F0 were lost, the unstable pitch fluctuation seen in the out-of-tune phrases was also suppressed, leading to good subjective evaluation results. Appropriate correction methods and standard deviation of prior σ_p should be selected based on which to prioritize, the reproducibility of F0 fluctuations particular to singers in training data, or the accuracy of the pitch. Note that none of the systems reached the subjective score of **System 1** in Section VI-C. This result indicates that it is difficult to predict the pitch accurately even if the pitch correction technique is used when the training data contains the out-of-tune phrase. This led to a decrease in MOS score because the naturalness of pitch fluctuation significantly affects the subjective quality of singing voices.

VII. Conclusion

We proposed a DNN-based SVS system called "Sinsy," designed to synthesize singing voice with singing-specific expressions at appropriate timing from a musical score. The proposed system consists of four DNN-based modules: a time-lag model, a duration model, an acoustic model, and a vocoder. The proposed system incorporates improved pitch and vibrato modeling, the better training criterion, and the pitch robust neural vocoder PeriodNet. Furthermore, we propose pitch correction techniques that enable synthesizing singing voices with the correct pitch even if the training data has out-of-tune phrases. Experimental results indicated the effectiveness of our novel techniques. Our proposed system can synthesize high-quality, high-fidelity singing voices that can follow a given musical score.

Future work includes investigating different distributions for prior in automatic pitch correction and evaluating the proposed system using the different speaker, genres, and language datasets. In our previous work [36], [46], since an HMM-based SVS system, which had a similar strategy combining the time-lag model, duration model, and acoustic models, was applied for synthesizing singing voice with other styles and languages, we think the proposed DNN-based SVS system can also support these kinds of songs. The modeling of songs in unique singing styles such as shouting and growling, which are difficult to annotate songs and extract acoustic feature representations, is also included in our future work. Furthermore, incorporating our proposed techniques, such as time-lag and vibrato modeling and automatic pitch correction, into seq-to-seq SVS systems is an important task. Recent studies [19]–[25] introduce a seq-to-seq model into the SVS system. Although such systems can model the singing voice as sequential mapping using an encoder-decoder model with an attention mechanism, they cannot model and control timing fluctuation explicitly. Extending a unified framework for simultaneously modeling acoustic feature and duration parameters [47] is one of our future works to model time-lags, durations, and acoustic features simultaneously.

References

[1] H. Kenmochi and H. Ohshita, "VOCALOID-commercial singing synthesizer based on sample concatenation," in *Proc. Interspeech*, 2007, pp. 4009–4010.

[2] J. Bonada, M. Umbert, and M. Blaauw, "Expressive singing synthesis based on unit selection for the singing synthesis challenge 2016," in *Proc. Interspeech*, 2016, pp. 1230–1234.

[3] K. Oura, A. Mase, T. Yamada, S. Muto, Y. Nankaku, and K. Tokuda, "Recent development of the HMM-based singing voice synthesis system-sinsy," in *Proc. ISCA SSW7*, 2010, pp. 211–216.

[4] G. Hinton *et al.*, "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," *IEEE Signal Process. Mag.*, vol. 29, no. 6, pp. 82–97, Nov. 2012.

[5] H. Zen, A. Senior, and M. Schuster, "Statistical parametric speech synthesis using deep neural networks," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, 2013, pp. 7962–7966.

[6] Y. Qian, Y. Fan, W. Hu, and F. K. Soong, "On the training aspects of deep neural network (DNN) for parametric TTS synthesis," in *Proc. Int. Conf. Acoust., Speech Signal Process.*, 2014, pp. 3829–3833.

[7] S. Desai, A. W. Black, B. Yegnanarayana, and K. Prahallad, "Spectral mapping using artificial neural networks for voice conversion," *IEEE Trans. Audio, Speech, Lang. Process.*, vol. 18, no. 5, pp. 954–964, Jul. 2010.

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

[9] Y. Hono *et al.*, "Recent development of the DNN-based singing voice synthesis system - sinsy," in *Proc. Asia-Pacific Signal and Inf. Process. Assoc. Annu. Summit Conf.*, 2018, pp. 1003–1009.

[10] J. Kim, H. Choi, J. Park, M. Hahn, S. Kim, and J.-J. Kim, "Korean singing voice synthesis system based on an LSTM recurrent neural network," in *Proc. Interspeech*, 2018, pp. 1551–1555.

[11] K. Nakamura, S. Takaki, K. Hashimoto, K. Oura, Y. Nankaku, and K. Tokuda, "Fast and high-quality singing voice synthesis system based on convolutional neural networks," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, 2020, pp. 7239–7243.

[12] M. Blaauw and J. Bonada, "A neural parametric singing synthesizer modeling timbre and expression from natural songs," *Appl. Sci.*, vol. 7, no. 12, 2017, Art. no. 1313.

[13] Y.-H. Yi, Y. Ai, Z.-H. Ling, and L.-R. Dai, "Singing voice synthesis using deep autoregressive neural networks for acoustic modeling," in *Proc. Interspeech*, 2019, pp. 2593–2597.

[14] K. Hashimoto, K. Oura, Y. Nankaku, and K. Tokuda, "Trajectory training considering global variance for speech synthesis based on neural networks," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, 2016, pp. 5600–5604.

[15] I. Goodfellow *et al.*, "Generative adversarial nets," in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 2672–2680.

[16] Y. Hono, K. Hashimoto, K. Oura, Y. Nankaku, and K. Tokuda, "Singing voice synthesis based on generative adversarial networks," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, 2019, pp. 6955–6959.

[17] A. Mase, K. Oura, Y. Nankaku, and K. Tokuda, "HMM-based singing voice synthesis system using pitch-shifted pseudo training data," in *Proc. Interspeech*, 2010, pp. 845–848.

[18] J. Shen *et al.*, "Natural TTS synthesis by conditioning WaveNet on mel spectrogram predictions," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, 2018, pp. 4779–4783.

[19] J. Lee, H.-S. Choi, C.-B. Jeon, J. Koo, and K. Lee, "Adversarially trained end-to-end korean singing voice synthesis system," in *Proc. Interspeech*, 2019, pp. 2588–2592.

[20] O. Angelini, A. Moinet, K. Yanagisawa, and T. Drugman, "Singing synthesis: With a little help from my attention," in *Proc. Interspeech*, 2020, pp. 1221–1225.

[21] M. Blaauw and J. Bonada, "Sequence-to-sequence singing synthesis using the feed-forward transformer," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, 2020, pp. 7229–7233.

[22] Y. Gu *et al.*, "ByteSing: A chinese singing voice synthesis system using duration allocated encoder-decoder acoustic models and wavernn vocoders," in *Proc. Int. Symp. Chin. Spoken Lang. Process.*, 2021, pp. 1–5.

[23] P. Lu, J. Wu, J. Luan, X. Tan, and L. Zhou, "XiaoiceSing: A high-quality and integrated singing voice synthesis system," in *Proc. Interspeech*, 2020, pp. 1306–1310.

[24] J. Chen, X. Tan, J. Luan, T. Qin, and T.-Y. Liu, "HiFiSinger: Towards high-fidelity neural singing voice synthesis," 2020, *arXiv:2009.01776*.

[25] J. Shi, S. Guo, N. Huo, Y. Zhang, and Q. Jin, "Sequence-to-sequence singing voice synthesis with perceptual entropy loss," in *Proc. ICASSP*, 2021, pp. 76–80.

[26] Y. Hono, S. Takaki, K. Hashimoto, K. Oura, Y. Nankaku, and K. Tokuda, "PeriodNet: A non-autoregressive waveform generation model with a structure separating periodic and aperiodic components," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, 2021, pp. 6049–6053.

[27] Y. Fan, Y. Qian, F.-L. Xie, and F. K. Soong, "TTS synthesis with bidirectional LSTM based recurrent neural networks," in *Proc. Interspeech*, 2014, pp. 964–1968.

[28] W. Wang, S. Xu, and B. Xu, "Gating recurrent mixture density networks for acoustic modeling in statistical parametric speech synthesis," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, 2016, pp. 5520–5524.

[29] X. Wang, S. Takaki, and J. Yamagishi, "An autoregressive recurrent mixture density network for parametric speech synthesis," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, 2017, pp. 4895–4899.

[30] H. Zen *et al.*, "LibriTTS: A corpus derived from LibriSpeech for text-to-speech," in *Proc. Interspeech*, 2019, pp. 1526–1530.

[31] 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 *Proc. 26th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2020, pp. 1979–1989.

[32] H. Zen, K. Tokuda, T. Masuko, T. Kobayasih, and T. Kitamura, "A hidden semi-Markov model-based speech synthesis system," *IEICE Trans. Inf. Syst.*, vol. E90-D, no. 5, pp. 825–834, 2007.

[33] "MusicXML definition." Accessed: 2021. [Online]. Available: http://musicxml.org

[34] K. Tokuda, T. Yoshimura, T. Masuko, T. Kobayashi, and T. Kitamura, "Speech parameter generation algorithms for HMM-based speech synthesis," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process. Proc.*, 2000, vol. 3, pp. 1315–1318.

[35] C. M. Bishop, "Mixture density networks," Neural Comput. Res. Group, Aston Univ., Birmingham, U.K., Tech. Rep. NCRG/94/004, 1994.

[36] K. Nakamura, K. Oura, Y. Nankaku, and K. Tokuda, "HMM-based singing voice synthesis and its application to japanese and english," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, 2014, pp. 265–269.

[37] T. Nakano, M. Goto, and Y. Hiraga, "An automatic singing skill evaluation method for unknown melodies using pitch interval accuracy and vibrato features," in *Proc. Interspeech*, 2006, pp. 1706–1709.

[38] K. Sawada, C. Asai, K. Hashimoto, K. Oura, and K. Tokuda, "The nitech text-to-speech system for the blizzard challenge 2016," in *Proc. Blizzard Challenge 2016 Workshop*, 2016.

[39] M. Morise, F. Yokomori, and K. Ozawa, "WORLD: A vocoder-based high-quality speech synthesis system for real-time applications," *IEICE Trans. Inf. Syst.*, vol. 99, no. 7, pp. 1877–1884, 2016.

[40] K. Tokuda, T. Masuko, N. Miyazaki, and T. Kobayashi, "Hidden Markov models based on multi-space probability distribution for pitch pattern modeling," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 1999, vol. 1, pp. 229–232.

[41] K. Shinoda and T. Watanabe, "Acoustic modeling based on the MDL principle for speech recognition," in *Proc. 5th Eur. Conf. Speech Commun. Technol.*, 1997, pp. 99–102.

[42] K. Saino, H. Zen, Y. Nankaku, A. Lee, and K. Tokuda, "An HMM-based singing voice synthesis system," in *Proc. Interspeech*, 2006, pp. 1141–1144.

[43] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. ICLR*, 2015.

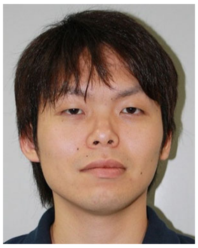
[44] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proc. Int. Conf. Mach. Learn.*, 2015, pp. 448–456.

[45] Y. Hono, "Sinsy demo." Accessed: 2021. [Online]. Available: https://www.sp.nitech.ac.jp/ hono/demos/taslp2021/

[46] K. Saino, K. Oura, M. Tachibana, H. Kenmochi, and K. Tokuda, "Rap-style singing voice synthesis," IPSJ SIG tech. rep., vol. 2012-MUS-94, no. 7, pp. 1–6, 2012.

[47] K. Tokuda, K. Hashimoto, K. Oura, and Y. Nankaku, "Temporal modeling in neural network based statistical parametric speech synthesis," in *Proc. ISCA SSW9*, 2016, pp. 106–111.

Yukiya Hono received the B.E. and M.E. degrees in 2017 and 2019, respectively, both in computer science from the Nagoya Institute of Technology, Nagoya, Japan, where he is currently working toward the Ph.D. degree. From July to August 2019, he was an Intern with Microsoft Development, Japan. He was a Visiting Researcher with the University of Edinburgh, Edinburgh, U.K., from October 2019 to December 2019 and with the University of Sheffield, Sheffield, U.K., from January 2020 to February 2020. His research interests include statistical speech synthesis, singing voice synthesis, and machine learning. He is a Member of the Acoustical Society of Japan (ASJ). He was the recipient of the 18th Student Presentation Award from ASJ, the 2019 Information and Communication Engineers (IEICE) Tokai Section Student Award, and the 2021 IEEE Nagoya Section Excellent Student Award.



Kei Hashimoto (Member, IEEE) received the B.E., M.E., and Ph.D. degrees in computer science, computer science and engineering, and scientific and engineering simulation from the Nagoya Institute of Technology, Nagoya, Japan, in 2006, 2008, and 2011, respectively. From October 2008 to January 2009, he was an Intern Researcher with the National Institute of Information and Communications Technology, Kyoto, Japan. From April 2010 to March 2012, he was a Research Fellow of the Japan Society for the Promotion of Science, Nagoya Institute of Technology. From May 2010 to September 2010, he was a Visiting Researcher with the University of Edinburgh, Edinburgh, U.K., and Cambridge University, Cambridge, U.K. From April 2012 to March 2017, he was a Specially Appointed Assistant Professor with the Nagoya Institute of Technology. From April 2017 to December 2018, he was a Specially Appointed Associate Professor with the Nagoya Institute of Technology, and he is currently an Associate Professor at the same institute. His research interests include statistical speech synthesis and speech recognition. He is a Member of IEICE and the Acoustical Society of Japan.

Keiichiro Oura received the Ph.D. degree in computer science and engineering from the Nagoya Institute of Technology, Nagoya, Japan, in 2010. From April 2010 to March 2017, he was a specially-appointed Assistant Professor with the Nagoya Institute of Technology. From April 2017 to May 2020, he was a specially-appointed Associate Professor with the Nagoya Institute of Technology. He is currently a Project Associate Professor with the Nagoya Institute of Technology and a CEO of the Techno-Speech, Inc. His research interests include statistical speech recognition and synthesis. He was the recipient of the ISCSLP Best Student Paper Award, the IPSJ YAMASHITA SIG Research Award, the ASJ ITAKURA Award, the IPSJ KIYASU Special Industrial Achievement Award, the ASJ AWAYA Prize Young Researcher Award, and the IPSJ Microsoft Faculty Award, in 2008, 2010, 2013, 2013, 2019, 2020, respectively. He is a Member of the Acoustical Society of Japan and the Information Processing Society of Japan.

Yoshihiko Nankaku (Member, IEEE) received the B.E. degree in computer science, and the M.E. and Ph.D. degrees with the Department of Electrical and Electronic Engineering, Nagoya Institute of Technology, Nagoya, Japan, in 1999, 2001, and 2004, respectively. After a year as a Postdoctoral Fellow with the Nagoya Institute of Technology, he became an Associate Professor at the same institute. From May to October 2011, he was a Visiting Researcher with the Department of Engineering, University of Cambridge, Cambridge, U.K. His research interests include statistical machine learning, speech recognition, speech synthesis, image recognition, and multimodal interface. He is a Member of the Institute of Electronics, Information and Communication Engineers, and the Acoustical Society of Japan.

Keiichi Tokuda (Fellow, IEEE) received the B.E. degree in electrical and electronic engineering from the Nagoya Institute of Technology, Nagoya, Japan, in 1984, and the M.E. and Dr. Eng. degrees in information processing from the Tokyo Institute of Technology, Tokyo, Japan, in 1986 and 1989, respectively. From 1989 to 1996, he was a Research Associate with the Department of Electronic and Electric Engineering, Tokyo Institute of Technology. From 1996 to 2004, he was an Associate Professor with the Department of Computer Science, Nagoya Institute of Technology, and he is currently a Professor at the same institute. He is also an Honorary Professor with the University of Edinburgh, Edinburgh, U.K. He was an Invited Researcher with ATR Spoken Language Translation Research Laboratories, Japan, from 2000 to 2013, a Visiting Researcher with Carnegie Mellon University, Pittsburgh, PA, USA, from 2001 to 2002, and with Google from 2013 to 2014. He authored or coauthored more than 80 journal papers and more than 200 conference papers. His research interests include speech coding, speech synthesis and recognition, and statistical machine learning. He was the recipient of six paper awards and three achievement awards. He was a Member of the Speech Technical Committee of the IEEE Signal Processing Society from 2000 to 2003, a Member of ISCA Advisory Council and an Associate Editor for the IEEE Transactions on Audio, Speech and Language Processing, and acts as a organizer and reviewer for many main speech conferences, workshops, and journals. He is an ISCA Fellow.