VOCALISTENER: A SINGING-TO-SINGING SYNTHESIS SYSTEM BASED ON ITERATIVE PARAMETER ESTIMATION

Tomoyasu Nakano **Masataka Goto**
National Institute of Advanced Industrial Science and Technology (AIST), Japan
{t.nakano, m.goto} [at] aist.go.jp

ABSTRACT

This paper presents a singing synthesis system, *VocaListener*, that automatically estimates parameters for singing synthesis from a user's singing voice with the help of song lyrics. Although there is a method to estimate singing synthesis parameters of pitch (F_0) and dynamics (power) from a singing voice, it does not adapt to different singing synthesis conditions (*e.g.*, different singing synthesis systems and their singer databases) or singing skill/style modifications. To deal with different conditions, VocaListener repeatedly updates singing synthesis parameters so that the synthesized singing can more closely mimic the user's singing. Moreover, VocaListener has functions to help modify the user's singing by correcting off-pitch phrases or changing vibrato. In an experimental evaluation under two different singing synthesis conditions, our system achieved synthesized singing that closely mimicked the user's singing.

1 INTRODUCTION

Many end users have started to use commercial singing synthesis systems to produce music and the number of listeners who enjoy synthesized singing is increasing. In fact, over one hundred thousand copies of popular software packages based on Vocaloid2 [1] have been sold and various compact discs that include synthesized vocal tracks have appeared on popular music charts in Japan. Singing synthesis systems are used not only for creating original vocal tracks, but also for enjoying collaborative creations and communications via content-sharing services on the Web [2, 3]. In light of the growing importance of singing synthesis, the aim of this study is to develop a system that helps a user synthesize natural and expressive singing voices more easily and efficiently. Moreover, by synthesizing high-quality human-like singing voices, we aim at discovering the mechanism of human singing voice production and perception.

Much work has been done on singing synthesis. The most popular approach for singing synthesis is *lyrics-to-singing (text-to-singing) synthesis* where a user provides note-level score information of the melody with its lyrics to synthesize a singing voice [1, 4, 5]. To improve naturalness and provide original expressions, some systems [1] enable a user to adjust singing synthesis parameters such as pitch (F_0) and dynamics (power). The manual parameter adjustment, however, is not easy and requires considerable time and effort. Another approach is *speech-to-singing synthesis* where a speaking voice reading the lyrics of a song is converted into a singing voice by controlling acoustic features [6]. This approach is interesting because a user can synthesize singing voices having the user's voice timbre, but various voice timbres cannot be used.

In this paper, we propose a new system named *VocaListener* that can estimate singing synthesis parameters (pitch and dynamics) by mimicking a user's singing voice. Since a natural voice is provided by the user, the synthesized singing voice mimicking it can be human-like and natural without time-consuming manual adjustment. We named this approach *singing-to-singing synthesis*. Janer *et al.* [7] tried a similar approach and succeeded to some extent. Their method analyzes acoustic feature values of the input user's singing and directly converts those values into the synthesis parameters. But their method is not robust with respect to different singing synthesis conditions. For example, even if we specify the same parameters, the synthesized results always differ when we change to another singing synthesis system or a different system's singer database because of the results' nonlinearity. The ability to mimic a user's singing is therefore limited.

To overcome such limitations on robustness, VocaListener iteratively estimates singing synthesis parameters so that after a certain number of iterations the synthesized singing can become more similar to the user's singing in terms of pitch and dynamics. In short, VocaListener can synthesize a singing voice while listening to its own generated voice through an original feedback-loop mechanism. Figure 1 shows examples of synthesized voices under two different conditions (different singer databases). With the previous approach [7], there were differences in pitch (F_0) and dynamics (power). On the other hand, such differences are minimal with VocaListener.

Moreover, VocaListener supports a highly-accurate lyrics-to-singing synchronization function. Given the user's singing and the corresponding lyrics without any score information, VocaListener synchronizes them automatically to determine each musical note that corresponds to a phoneme of the lyrics. We therefore developed an originally-adapted/trained acoustic model for singing syn-

SMC 2009, July 23-25, Porto, Portugal

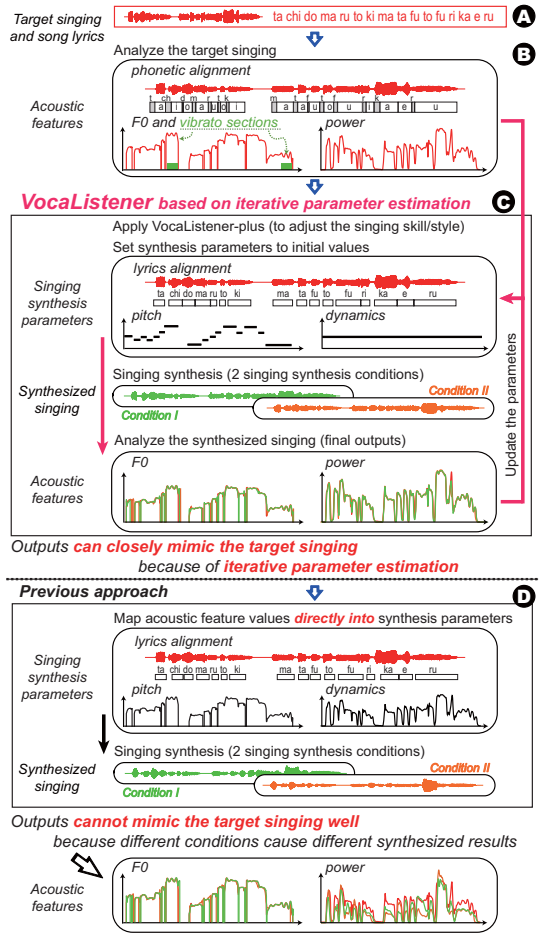

Figure 1. Overview of *VocaListener* and problems of a previous approach by Janer *et al.* [7].

chronization. Although synchronization errors with this model are rare, we also provide an interface that lets a user easily correct such errors just by pointing them out. In addition, VocaListener also supports a function to improve synthesized singing as if the user's singing skill were improved.

2 PARAMETER ESTIMATION SYSTEM FOR SINGING SYNTHESIS: VOCALISTENER

VocaListener consists of three components, the *VocaListener-front-end* for singing analysis and synthesis, the *VocaListener-core* to estimate the parameters for singing synthesis, and the *VocaListener-plus* to adjust the singing skill/style of the synthesized singing.

Figure 1 shows an overview of the VocaListener system. The user's singing voice (i.e., *target singing*) and the lyrics[1] are taken as the system input (Ⓐ). Using this input, the system automatically synchronizes the lyrics with the target singing to generate note-level score information, estimates the fundamental frequency (F_0) and the power of the target singing, and detects vibrato sections that are used just for the *VocaListener-plus* (Ⓑ). Errors in the lyrics synchronization can be manually corrected through simple interaction. The system then iteratively estimates the parameters through the VocaListener-core, and synthesizes the singing voice (Ⓒ). The user can also adjust the singing skill/style (*e.g.*, vibrato extent and F_0 contour) through the VocaListener-plus.

[1] In our current implementation, Japanese lyrics spelled in a mixture of Japanese phonetic characters and Chinese characters are mainly supported. English lyrics can also be easily supported because the underlying ideas of VocaListener are universal and language-independent.

2.1 VocaListener-front-end: analysis and synthesis

The VocaListener-front-end consists of singing analysis and singing synthesis. Throughout this paper, singing samples are monaural recordings of solo vocal digitized at 16 bit / 44.1 kHz.

2.1.1 singing analysis

The system estimates the fundamental frequency (F_0), the power, and the onset time and duration of each musical note. Since the analysis frame is shifted by 441 samples (10 ms), the discrete time step (1 *frame-time*) is 10 ms. This paper uses time t for the time measured in frame-time units.

In VocaListener, these features are estimated as follows:

Fundamental frequency: $F_0(t)$ is estimated using SWIPE [8]. Hereafter, unless otherwise stated, $F_0(t)$ are log-scale frequency values (real numbers) in relation to the MIDI note number (a semitone is 1, and middle C corresponds to 60).

Power: $Pow(t)$ is estimated by applying a Hanning window whose length is 2048 samples (about 46 ms).

Onset time and duration: To estimate the onset time and duration of each musical note, the system synchronizes the phoneme-level pronunciation of the lyrics with the target singing. This synchronization is called *phonetic alignment* and is estimated through Viterbi alignment with a phoneme-level hidden Markov model (monophone HMM). The pronunciation is estimated by using a Japanese language morphological analyzer [9].

2.1.2 singing synthesis

In our current implementation, the system estimates parameters for commercial singing synthesis software based on Yamaha's Vocaloid or Vocaloid2 technology [1]. For example, we use software named Hatsune Miku (referred to as CV01) and Kagamine Rin (referred to as CV02) [10] for synthesizing Japanese female singing. Since all parameters are estimated every 10 ms, they are linearly interpolated at every 1 ms to improve the synthesized quality, and are fed via a VSTi plug-in (Vocaloid Playback VST Instrument).

2.2 VocaListener-plus: adjusting singing skill/style

To extend the flexibility, the VocaListener-plus provides functions, *pitch change* and *style modification*, which can

modify the value of the estimated acoustic features of the target singing. The user can select whether to use these functions based on personal preference. Figure 2 shows an example of using these functions.

2.2.1 Pitch change

We propose *pitch transposition* and *off-pitch correction* to overcome the limitations of the user's singing skill and pitch range. The pitch transposition function changes the target $F_0(t)$ just by adding an offset value for transposition during the whole section or a partial section.

The off-pitch correction function automatically corrects off-pitch phrases by adjusting the target $F_0(t)$ according to an offset of F_d ($0 \leq F_d < 1$) estimated for each voiced section. The off-pitch amount F_d is estimated by fitting a semitone-width grid to $F_0(t)$. The grid is defined as a comb-filter-like function where Gaussian distributions are aligned at one semitone intervals. Just for this fitting, $F_0(t)$ is temporarily smoothed by using an FIR lowpass filter with a 3-Hz cutoff frequency[2] to suppress F_0 fluctuations (overshoot, vibrato, preparation, and fine fluctuation) of the singing voice [11, 12]. Last, the most fitted offset F_d is used to adjust $F_0(t)$ to its nearest correct pitch.

2.2.2 Style modification

In this paper, *vibrato adjustment* and *singing smoothing* are proposed to emphasize or suppress the F_0 fluctuations. Since the F_0 fluctuations are important factors to characterize human singing [11, 12], a user can change the impression of singing. The $F_0(t)$ and $Pow(t)$ of the target singing are adjusted by interpolating or extrapolating between the original values ($F_0(t)$ and $Pow(t)$) and their smoothed values obtained by using an FIR lowpass filter. A user can separately adjust vibrato sections and other sections. The vibrato sections are detected by using the vibrato detection method [13].

2.3 VocaListener-core: estimating the parameters

Figure 3 shows the estimation process for *VocaListener-core*. After acoustic features of the target singing (modified by VocaListener-plus, if necessary) are estimated, these features are converted into synthesis parameters that are then fed to the singing synthesis software. The synthesized singing is then analyzed and compared with the target singing. Until the synthesized singing is sufficiently close to the target singing, the system repeats the parameter update and its synthesis.

2.3.1 Parameters for singing synthesis

The system estimates parameters for *pitch*, *dynamics*, and *lyrics alignment* (Table 1). The pitch parameters consist of *MIDI note number (Note#)*[3], *pitch bend (PIT)*, and *pitch bend sensitivity (PBS)*, and the dynamics parameter is *dynamics (DYN)*. For the pitch (F_0), the fractional portion (PIT) is separated from the integer portion (Note#). PIT represents a relative decimal deviation from the corresponding integer note number (Note#), and PBS specifies the range (magnitude) of its deviation. The results of the lyrics alignment are represented by the note onset (onset time) and its duration.

These MIDI-based parameters can be considered typical and common, not specific to the Vocaloid software. A set of these parameters, PIT, PBS, and DYN, are iteratively estimated after being initialized to 0, 1, and 64, respectively.

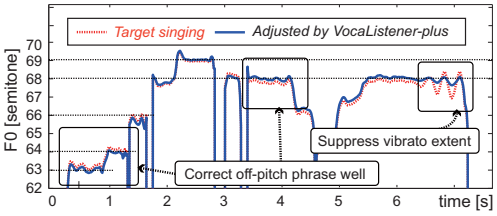

Figure 2. Example of $F_0(t)$ adjusted by VocaListener-plus.

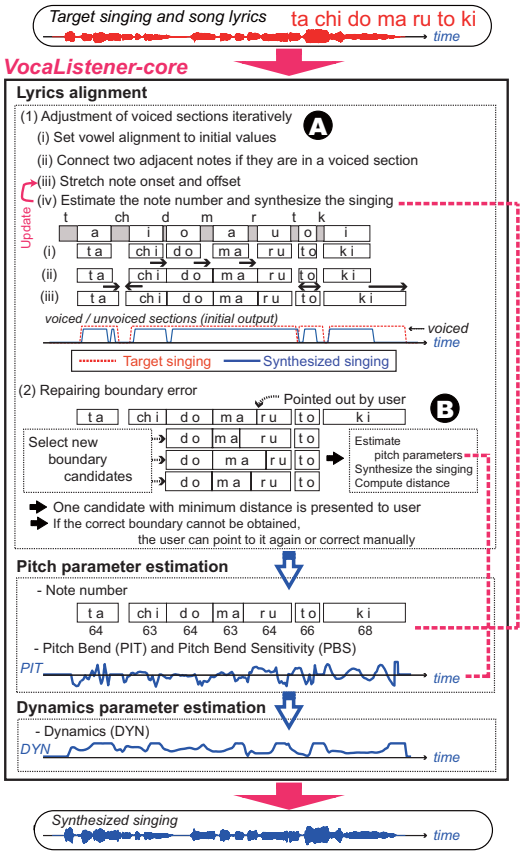

Figure 3. Overview of the parameter estimation procedure, *VocaListener-core*.

[2] We avoid unnatural smoothing by ignoring silent sections and leaps of F_0 transitions wider than a 1.8-semitone threshold.

[3] For synthesis, each mora of Japanese pronunciation is mapped into a musical note, where the mora representation can be classified into three types: "V", "CV", and "N". "V" denotes vowel (*a, i, ...*), "C" denotes consonant (*t, ch, ...*), and "N" denotes syllabic nasal (*n*).

Table 1. Relation between singing synthesis parameters and acoustic features.

Acoustic features	Synthesis parameters	
F_0	Pitch	Note#, PIT, and PBS
Power	Dynamics	DYN
Phonetic alignment	Lyrics alignment	Note onset Note duration

2.3.2 *Lyrics alignment estimation with error repairing*

Even if the same note onset and its duration (lyrics alignment) are given to different singing synthesis systems (such as Vocaloid and Vocaloid2) or different singer databases (such as CV01 and CV02), the note onset and note duration often differ in the synthesized singing because of their nonlinearity (caused by their internal waveform concatenation mechanism). We therefore have to adjust (update) the lyrics alignment iteratively so that each voiced section of the synthesized singing can be the same as the original voiced section of the target singing. As shown in Figure 3Ⓐ, the last two steps (iii) and (iv) in the following four steps are repeated:

Step (i) Given the phonetic alignment of the automatic synchronization, the note onset and duration are initialized by using its vowel.

Step (ii) If two adjacent notes are not connected but their sections are judged to be a single voiced section, the duration of the former note is extended to the onset of the latter note so that they can be connected. This eliminates a small gap and improves the naturalness of the synthesized singing.

Step (iii) By comparing voiced sections of the target and synthesized singing, the note onset and duration are adjusted so that they become closer to those of the target.

Step (iv) Given the new alignment, the note number (Note#) is estimated again and the singing is synthesized.

Although the automatic synchronization of song lyrics with the target singing is accurate in general, there are sometimes a few boundary errors that degrade the synthesized quality. We therefore propose an interface that lets a user correct each error just by pointing it out without manually adjusting (specifying) the boundary. As shown in Figure 3Ⓑ, other boundary candidates are shown on a screen so that the user can simply choose the correct one by listening to each one. Even if it is difficult for a user to specify the correct boundary from scratch, it is easy to choose the correct candidate interactively. To generate candidates, the system computes timbre fluctuation values of the target singing by using ΔMFCCs, and several candidates with high fluctuation values are selected. The system then synthesizes each candidate and compares it with the target singing by using MFCCs. The candidates are sorted and presented to the user in the order of similarity to the target singing. If none of the candidates are correct, the user can correct manually at the frame level.

2.3.3 *Pitch parameter estimation*

Given the results of lyrics alignment, the pitch parameters are iteratively estimated so that the synthesized F_0 can become closer to the target F_0. After the note number of each

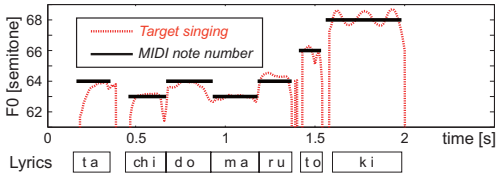

Figure 4. F_0 of the target singing and estimated note numbers.

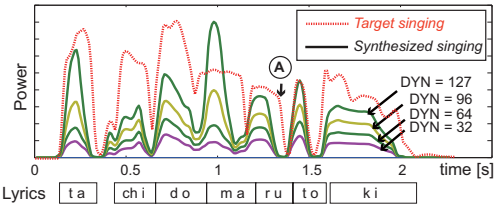

Figure 5. Power of the target singing and power of the singing synthesized with four different dynamics.

note is estimated, PIT and PBS are repeatedly updated by minimizing a distance between the target F_0 and the synthesized F_0.

The note number $Note\#$ for each note is estimated by

$$\text{Note\#} = \underset{n}{\text{argmax}} \left(\sum_t \exp\left\{ -\frac{(n - F_0(t))^2}{2\sigma^2} \right\} \right), \quad (1)$$

where n denotes a note number candidate, is set to 0.33, and t is 0 at the note onset and continues for its duration. Figure 4 shows an example of F_0 and its estimated note numbers.

The PIT and PBS are then estimated by repeating the following steps, where i is the number of updates (iterations), $F0_{\text{org}}(t)$ denotes F_0 of the target singing, and PIT and PBS are represented by $PIT^{(i)}(t)$ and $PBS^{(i)}(t)$:

Step 1) Obtain synthesized singing from the current parameters.

Step 2) Estimate $F0^{(i)}_{\text{syn}}(t)$ that denotes F_0 of the synthesized singing.

Step 3) Update $Pb^{(i)}(t)$ by

$$Pb^{(i+1)}(t) = Pb^{(i)}(t) + \left(F0_{\text{org}}(t) - F0^{(i)}_{\text{syn}}(t) \right), \quad (2)$$

where $Pb^{(i)}(t)$ is a log-scale frequency computed from $PIT^{(i)}(t)$ and $PBS^{(i)}(t)$.

Step 4) Obtain the updated $PIT^{(i+1)}(t)$ and $PBS^{(i+1)}(t)$ from $Pb^{(n+1)}(t)$ after minimizing $PBS^{(i+1)}(t)$. Since a smaller PBS gives better resolution of the synthesized F_0, PBS should be minimized at every iteration as long as PIT can represent the correct relative deviation.

2.3.4 *Dynamics parameter estimation*

Given the results of lyrics alignment and the pitch parameters, the dynamics parameter is iteratively estimated so that the synthesized power can be closer to the target power. Figure 5 shows the power of the target singing before normalization and the power of the singing synthesized with four different dynamics. Since the power of the target singing depends on recording conditions, it is important to mimic the relative power after normalization that is determined so

Table 2. Dataset for experiments A and B and synthesis conditions. All of the song samples were sung by female singers.

Exp. No.	Song No.	Excerpted section	Length [s]	Synthesis conditions
A	No.07	intro–verse–chorus	103	CV01
A	No.16	intro–verse–chorus	100	CV02
B	No.07	verse A	6.0	CV01, CV02
B	No.16	verse A	7.0	CV01, CV02
B	No.54	verse A	8.9	CV01, CV02
B	No.55	verse A	6.5	CV01, CV02

that the normalized target power can be covered by the synthesized power with DYN = 127 (maximum value). However, because there are cases where the target power exceeds the limit of synthesis capability (*e.g.*, Fig.5Ⓐ), the synthesized power cannot perfectly mimic the target. As a compromise, the normalization factor α is determined by minimizing an error defined as a square error between $\alpha Pow_{\rm org}(t)$ and $Pow_{\rm syn}^{\rm DYN=64}(t)$, where $Pow_{\rm syn}^{\rm DYN=64}(t)$ denotes the synthesized power with DYN = 64.

The DYN is then estimated by repeating the following steps, where $Pow_{\rm org}(t)$ denotes the power of the target singing:

Step 1) Obtain synthesized singing from the current parameters.

Step 2) Estimate $Pow_{\rm syn}^{(i)}(t)$ that denotes the power of the synthesized singing.

Step 3) Update $Db^{(i)}(t)$ by

$$Db^{(i+1)}(t) = Db^{(i)}(t) + \left(\alpha Pow_{\rm org}(t) - Pow_{\rm syn}^{(i)}(t)\right), \quad (3)$$

where $Db^{(i)}(t)$ is the actual power given by the current DYN.

Step 4) Obtain the updated DYN from $Db^{(i+1)}(t)$ by using the relationship between the DYN and the actual power values. Before these iteration steps, this relationship should be investigated once by synthesizing the current singing with five DYN values ($= 0, 32, 64, 96, 127$). The relationship for each of the other DYN values is linearly interpolated.

3 EXPERIMENTAL EVALUATIONS

The VocaListener was tested in two experiments. Experiment A evaluated the number of times manual corrections had to be made, and experiment B evaluated the performance of the iterative estimation under different conditions.

In these experiments, two singer databases, CV01 and CV02, were used with the default software settings except for the note-level properties of "No Vibrato" and "0% Bend Depth". Unaccompanied song samples (solo vocal) were taken from the RWC Music Database (Popular Music [14]), and were used as the target singing as shown in Table 2.

For the automatic synchronization of the song lyrics in experiment A, a speaker-independent HMM provided by CSRC [15] for speech recognition was used as the basic acoustic model for MFCCs, ΔMFCCs, and Δpower. The HMM was adapted with singing voice samples by applying MLLR-MAP [16]. As in cross validation where one song sample is evaluated as the test data and the other samples are used as the training data, we excluded the same singer from the HMM adaptation data.

3.1 Experiment A: interactive error repairing for lyrics alignment

To evaluate the lyrics alignment, experiment A used two female songs that were over 100 s in length. Table 3 shows the number of boundary errors that had to be repaired (pointed out) and the number of repairs needed to correct those errors[4]. For example, among 128 musical notes for song No.16, there were only three boundary errors that should be manually pointed out on our interface, and two of these were pointed out twice. In other words, one error was corrected by choosing the first candidate, and the other two errors were corrected by choosing the second candidate. In our experience with many songs, errors tend to occur around /w/ or /r/ (semivowel, liquid) and /m/ or /n/ (nasal sound).

3.2 Experiment B: iterative estimation experiment

Experiment B used four song excerpts sung by four female singers. As shown in Table 2, each song was tested with two conditions — i.e., two singer databases, CV01 and CV02. Since the experiment focused on the performance of the iterative estimation for the pitch and dynamics, we used the hand-labeled lyrics alignment here. The results were evaluated by the *mean error value* defined by

$$\mathrm{err}_{\rm f0}^{(i)} = \frac{1}{T_f}\sum_t \left| F0_{\rm org}(t) - F0_{\rm syn}^{(i)}(t) \right|, \quad (4)$$

$$\mathrm{err}_{\rm pow}^{(i)} = \frac{1}{T_p}\sum_t \left| 20\log\left(\alpha Pow_{\rm org}(t)\right) - 20\log\left(Pow_{\rm syn}^{(i)}(t)\right) \right|, \quad (5)$$

where T_f denotes the number of voiced frames, and T_p denotes the number of nonzero power frames.

Table 4 shows the mean error values after each iteration for song No.07, where the "$\times n$" column denotes the number of iterations before synthesis and the "$\times 0$" column denotes initial synthesis without any iteration. Starting from large errors of initial synthesis ("$\times 0$"), the mean error values were monotonically decreased after each iteration and the synthesized singing after the fourth iteration ("$\times 4$") was most similar to the target singing. The results for the other songs also showed similar improvement as shown in Table 5. The "Previous approach" column in Tables 4 and 5 denotes the results of mapping acoustic feature values directly into synthesis parameters (almost equivalent to [7]). The mean error values after the fourth iteration were much smaller than the previous approach. In fact, when we listened to those synthesized results, the synthesized results after the fourth iteration ("$\times 4$") were clearly better than the synthesized results without any iteration ("$\times 0$" and "Previous approach").

3.3 Discussion

The results of experiment A show that our automatic synchronization (lyrics alignment) worked well. Even if there were a few boundary errors (eight errors among 166 notes in No.07 and three errors among 128 notes in No.16), they

[4] This table does not show another type of error where the global phrase boundary was wrong. There were two such errors in No.16 and they could also be corrected through simple interaction (just by moving roughly).

Table 3. Number of boundary errors and number of repairs for correcting (pointing out) errors in experiment A.

Song No.	Synthesis conditions	Number of notes	Number of boundary errors after each repair			
			×0	×1	×2	×3
No.07	CV01	166	8	5	2	0
No.16	CV02	128	3	2	0	—

Table 4. Mean error values after each iteration for song No.07 in experiment B.

Parameters	Synthesis conditions	Mean error values ($\mathrm{err}_{\mathrm{f0}}^{(i)}$ [semitone] and $\mathrm{err}_{\mathrm{pow}}^{(i)}$ [dB])					
		Previous approach	VocaListener				
			×0	×1	×2	×3	×4
Pitch	CV01	0.217	0.386	0.091	0.058	0.042	0.034
Pitch	CV02	0.198	0.352	0.074	0.041	0.029	0.024
Dynamics	CV01	13.65	11.22	4.128	3.617	3.472	3.414
Dynamics	CV02	14.17	15.26	6.944	6.382	6.245	6.171

Table 5. Minimum and maximum error values for all four songs in experiment B.

Parameters	Mean error values (min−max)		
	Previous approach	VocaListener	
		×0	×4
Pitch	0.168−0.369	0.352−1.029	0.019−0.107
Dynamics	9.545−15.45	10.46−19.04	1.676−6.560

could be easily corrected by choosing from the top three candidates. We thus confirmed that our interface for correcting boundary errors was easy-to-use and efficient. Moreover, we recently developed an original acoustic model that was trained from scratch with singing voices including a wide range of vocal timbres and singing styles. Although we did not use this high-performance model in the above experiments, our preliminary evaluation results suggest that more accurate synchronization can be achieved.

The results of experiment B show that iterative updates were an effective way to mimic the target singing under various conditions. In addition, we tried to estimate the parameters for CV01/CV02 using song samples synthesized with CV01 as the target singing, and confirmed that the estimated parameters for CV01 were almost same with the original parameters and the synthesized singing with CV01/CV02 sufficiently mimicked the target singing. VocaListener can thus be used not only for mimicking singing by human, but also for re-estimating the parameters under different synthesis conditions without time-consuming manual adjustment.

4 CONCLUSION

We have described a singing-to-singing synthesis system, VocaListener, that automatically estimates parameters for singing synthesis by mimicking a user's singing. The experimental results indicate that the system effectively mimics target singing with error values decreasing with the number of iterative updates. Although Japanese lyrics are currently supported in our implementation, our approach can be utilized for any other language.

In our experience of synthesizing various songs with VocaListener using seven different singer databases on two different singing synthesis systems (Vocaloid and Vocaloid2), we found the synthesized quality was high and stable[5] . One benefit of VocaListener is that a user does not need to perform time-consuming manual adjustment even if the singer database changes. Before VocaListener, this problem was widely recognized and many users had to repeatedly adjust parameters. With VocaListener, once a user synthesizes a song based on the target singing (even synthesized singing the user has adjusted in the past), its vocal timbre can be easily changed just by switching a singer database on our interface. Since this ability is very useful for end users, we name this meta-framework a *Meta-Singing Synthesis System*. We hope that a future singing synthesis framework will support this promising idea, thus expediting wider use of singing synthesis systems to produce music.

[5] A demonstration video including examples of synthesized singing is available at http://staff.aist.go.jp/t.nakano/VocaListener/.

5 ACKNOWLEDGEMENTS

We thank Jun Ogata (AIST), Takeshi Saitou (CREST/AIST), and Hiromasa Fujihara (AIST) for their valuable discussions. This research was supported in part by CrestMuse, CREST, JST.

6 REFERENCES

[1] Kenmochi, H. *et al.* "VOCALOID – Commercial Singing Synthesizer based on Sample Concatenation," *Proc. INTERSPEECH 2007*, pp.4011–4010, 2007.

[2] Hamasaki, M. *et al.* "Network Analysis of Massively Collaborative Creation of Multimedia Contents: Case Study of Hatsune Miku Videos on Nico Nico Douga," *Proc. uxTV' 08*, pp.165–168, 2008.

[3] Cabinet Office, Government of Japan. "Virtual Idol," *Highlighting JAPAN through images*, Vol.2, No.11, pp.24–25, 2009. http://www.gov-online.go.jp/pdf/hlj_img/vol_0020et/24-25.pdf

[4] Bonada, J. *et al.* "Synthesis of the Singing Voice by Performance Sampling and Spectral Models," *IEEE Signal Processing Magazine*, Vol.24, Iss.2, pp.67–79, 2007.

[5] Saino K. *et al.* "HMM-based singing voice synthesis system, " *Proc. ICSLP06*, pp.1141–1144, 2006.

[6] Saitou, T. *et al.* "Speech-To-Singing Synthesis: Converting Speaking Voices to Singing Voices by Controlling Acoustic Features Unique to Singing Voices," *Proc. WASPAA2007*, pp.215–218, 2007.

[7] Janer, J. *et al.*: "Performance-Driven Control for Sample-Based Singing Voice Synthesis," *Proc. DAFx-06*, pp.42–44, 2006.

[8] Camacho, A. "SWIPE: A Sawtooth Waveform Inspired Pitch Estimator for Speech and Music," Ph.D. Thesis, University of Florida, 116 p., 2007.

[9] Kudo, T. "MeCab: Yet Another Part-of-Speech and Morphological Analyzer". http://mecab.sourceforge.net/

[10] Crypton Future Media. "What is the HATSUNE MIKU movement?," http://www.crypton.co.jp/download/pdf/info_miku_e.pdf

[11] Saitou, T. *et al.* "Development of an F0 control Model Based on F0 Dynamic Characteristics for Singing-Voice Synthesis," *Speech Communication*, Vol.46, pp.405-417, 2005.

[12] Mori, H. *et al.* "F_0 Dynamics in Singing: Evidence from the Data of a Baritone Singer," *IEICE Trans. Inf. & Syst.*, Vol.E87-D, No.5, pp.1086–1092, 2004.

[13] Nakano, T. *et al.* "An Automatic Singing Skill Evaluation Method for Unknown Melodies Using Pitch Interval Accuracy and Vibrato Features," *Proc. ICSLP 2006*, pp.1706–1709, 2006.

[14] Goto, M. *et al.* "RWC Music Database: Popular, Classical, and Jazz Music Databases," *Proc. ISMIR 2002*, pp.287–288, 2002.

[15] Lee, A. *et al.* "Continuous Speech Recognition Consortium – An Open Repository for CSR Tools and Models –," *Proc. LREC2002*, pp.1438–1441, 2002.

[16] Digalakis, V.V. *et al.*: "Speaker Adaptation Using Combined Transformation and Bayesian Methods," *IEEE Transactions on Speech and Audio Processing*, Vol.4, No.4, pp.294–300, 1996.