LiVo: Sing a Song with a Vowel Keyboard

Kazuhiko Yamamoto
The University of Tokyo

Takeo Igarashi*
The University of Tokyo

ABSTRACT

We propose a novel user interface that enables control of a singing voice synthesizer at a live improvisational performance. The user first registers the lyrics of a song with the system before performance, and the system builds a probabilistic model that models the possible jumps within the lyrics. During performance, the user simultaneously inputs the lyrics of a song with the left hand using a vowel keyboard and the melodies with the right hand using a standard musical keyboard. Our system searches for a portion of the registered lyrics whose vowel sequence matches the current user input using the probabilistic model, and sends the matched lyrics to the singing voice synthesizer. The vowel input keys are mapped onto a standard musical keyboard, enabling experienced keyboard players to learn the system from a standard musical score. We examine the feasibility of the system through a series of evaluations and user studies.

Author Keywords

Live Performance, Singing Voice Synthesis, Human Computer Interface, Musical Instrument, Text Entry

ACM Classification

H.5.5 [Information Interfaces and Presentation] Sound and Music Computing, Methodologies and techniques. Signal analysis, synthesis, and processing.

1. INTRODUCTION

The use of a singing voice synthesizer such as VOCALOID [4] has become very popular especially in Japan. However, there is little precedent of live improvisational performance using real-time singing voice synthesis even though there is a huge demand for it. This is mainly because it is very difficult to input song lyrics at a real-time rate; this is the problem we want to address in this paper.

A possible approach is to use automatic fitting of the predefined original lyrics to the melody currently being played using melody matching. However, this approach has two problems. First, players often modify the melody significantly including addition of grace notes and change of order in a live improvisational performance. Second, the same melodies often appear repeatedly in a song with different lyrics, making it very difficult to find the appropriate lyrics from melody alone in improvisational performance.

Another possible approach is to use speech recognition to input lyrics. This allows the user to improvise arbitrary lyrics during performance, but also presents several problems. First, recent popular speech recognition techniques are optimized for recognizing continuous speech as a whole, rather than for recognizing individual characters in a song separately for timed performance. Second, latency is inevitable in speech recognition, but is not acceptable for real-time musical performance. Finally, it is difficult for the player to listen to his or her own performance while vocalizing.

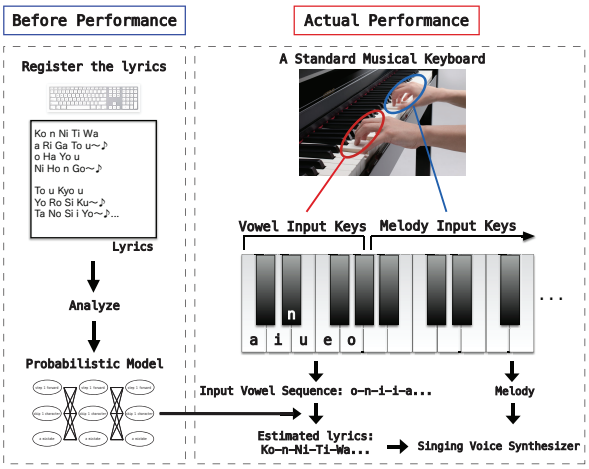

Figure 1: An overview of the proposed system. Our system consists of two steps, lyric registration step and actual performance step. At the lyrics registration step, the user registers the lyrics of the songs, and the system analyzes it. At the actual performance step, the user simultaneously inputs the vowel sequences and melodies using a musical keyboard, and the system estimates the plausible lyrics from them and synthesizes singing voice sounds.

There are a few experimental systems that allow the user to input arbitrary Japanese lyrics during live performance using a combination of vowel and consonant keys [6][8]. However, they require the user to press two keys simultaneously to input a character, making it difficult to play fast songs.

To address these problems, we propose to use a vowel keyboard to input the lyrics during live improvisational performances (Figure 1: right). In our system, the user inputs the lyrics with one hand using a vowel keyboard and the melodies with the other hand using a musical keyboard simultaneously. Our system allows the user to modify the melodies of a song freely and to pick an arbitrary portion of predefined lyrics during a live performance.

Our system is designed for Japanese lyrics. In Japanese, a character consists of a consonant and a vowel (Figure 2: left). Hence, multiple Japanese characters match a given vowel. However, we can identify the most plausible character sequence in the predefined lyrics by finding the corresponding vowel sequence using a probabilistic alignment technique (Figure 1). Specifically, our system automatically finds a portion of the predefined lyrics whose vowel sequence matches well with the vowel sequence being input by the player. We use a Hidden Markov model for alignment.

There are only five vowels in Japanese, "a", "i", "u", "e", and "o". We also use a special character "n", hence we use six keys to input lyrics. This makes it possible to input vowels very rapidly without moving the hand to other locations in contrast to other methods that use many keys to input lyrics. Additionally, by mapping the vowel keyboard onto a traditional musical keyboard, one can represent lyrics as a standard musical score (Figure 2: right), enabling the user to practice the skills more easily.

*The University of Tokyo

NIME'15, May 31-June 3, 2015, Louisiana State Univ., Baton Rouge, LA.

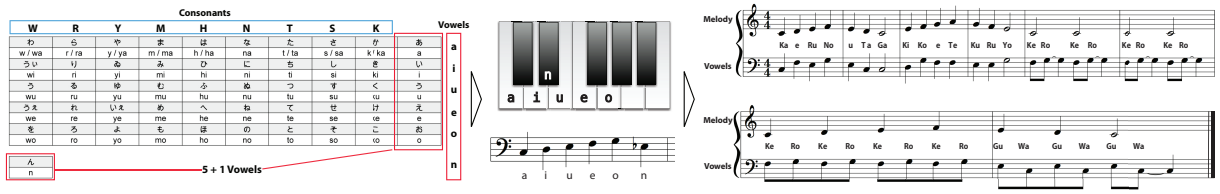

Figure 2: In Japanese, a character consists of a consonant and a vowel (left). For example, "Ka" is the combination of "K" and "a", "Su" is the combination of "S" and "u". There are 5 vowels and a special vowel (n) in Japanese. We mapped these vowels on a piano keyboard (center). Because of this, in our system, the lyrics can be represented as a standard musical score as the left hand part (right).

2. RELATED WORK

Yamamoto et al. [8] used a combination of a dedicated special keyboard to input lyrics used by the left hand and a standard musical keyboard used by the right hand for improvisational performance. The lyrics keyboard is designed for Japanese, consisting of ten consonant keys and five vowel keys placed to fit the left hand. The user inputs a character by pressing a combination of a consonant key and a vowel key. However, it is very difficult for a typical player to press correct multiple keys simultaneously during a live performance.

Formant Bros. [6] assigned lyrics input keys to a common musical keyboard. A character can be input using the triplet three-key combination of a pitch key, a consonant key, and a vowel key. The benefit of this approach is that it enables description of lyrics and melodies as a standard musical score, making the method easy to learn. However, the consecutive triplet chord input is very difficult even for professional pianists. Thus, the approach remained at the level of playing very slow nursery rhymes, in contrast to regular songs played at a realistic speed (we assume the range of tempo of regular songs is about 50∼200 BPM [beats per minute]).

HANAUTAU [7] uses pitch detection from the user's voice inputted by microphone for melody and lyrics typed with both hands using a common QWERTY keyboard. However, using a QWERTY keyboard does not provide input at a speed sufficient to play common music adequately.

A case has been made to use a Flick text input method [5] for live performances using real-time singing voice synthesis to input lyrics. Although the Flick text input method is a very fast text entry method, , it still can't achieve sufficient input speed for singing a song. Additionally, because it requires two-step control (push and slide), it is difficult to adjust the timing to the music using that method.

DiVA [1] uses CyberGlove and several sensors and measured the hands gestures to control the lyrics. The gestures are trained and trigger a neural network with a given gestural language that associates one posture for each phoneme of English. However, the gestural control is difficult for fast songs.

Cantor Digitalis [2] has been used in several musical improvisations using singing voice synthesizer by multi-touch tablet. Their alphabet control is limited in only a few vowels (formants) and can't output the most characters including consonants as a language. Then, their system is inadequate for performing the lyrics of common songs. We address this issue.

3. USER INTERFACE

An overview of our system is shown in Figure 1. Our system requires two steps. The first step is lyrics registration before performance. The second step is actual performance. At the lyrics registration step, the user registers the intended lyrics to be used. The user can register multiple lyrics at a time. At the performance step, the user simultaneously inputs vowel sequences using a vowel keyboard and melodies using a musical keyboard. The system estimates the plausible lyrics from the vowel sequences and synthesizes singing voice sounds. Note that the system does not use a melody

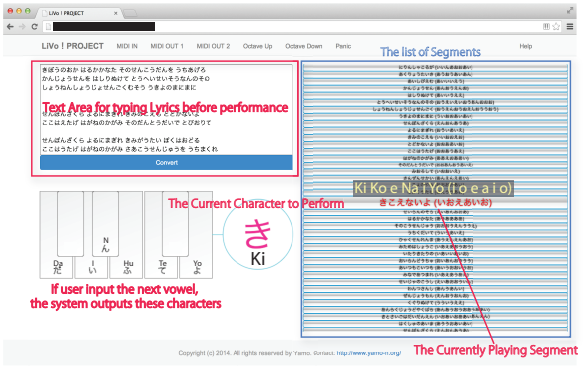

Figure 3: The user interface view of the LiVo system on a web browser. The user enters the lyrics in the top left text area before performance.

sequence for estimation.

The vowel keys are assigned to a portion of a standard musical keyboard. In our prototype, each key mapping is set as "a" to C2, "i" to D2, "u" to E2, "e" to F2, "o" to G2, "n" to D#2, to fit all the keys in a palm (Figure 2: center). This makes it possible to represent a vowel sequence as a standard musical score, making it easier for players to practice the performance. In addition, the musical score for our system requires only monophonic phrases of six vowel keys, which is much simpler than that of Formant Bros. Actually, the musical score for our system is much easier than popular piano scores such as J.S.Bach and Chopin.

Our system doesn't require the user to input the vowel sequences strictly in the order of the original lyrics, because the system estimates the plausible lyrics using a probabilistic model. This allows the user to jump to arbitrary positions in the lyrics including backtracking. Additionally, our system allows the user to make mistakes, freeing the player from paying excessive attention to vowel input. When the user jumps the position of the lyrics, the system would output a few wrong characters until following the user. However, these wrong characters at least have correct vowels, which limits the level of discomfort.

The timing control is complicated because two keys must be pressed in a coordinated way. Our current implementation is as follows. If a pitch key has already been pressed, the system begins a new voice when a new vowel key is pressed. However, if no pitch key has been pressed, the system does not begin a new voice, when a new vowel key is pressed. If a vowel key has already been pressed, the system begins a new voice, when a new pitch key is pressed. If no vowel key is pressed, the system begins a new voice with the last vowel input by the user, when the user presses a new pitch key. We choose this asymmetric scheme because pitch keys serve as the main control, with vowel keys serving as a modifier. Note that previous systems [6] [8] begin a voice only when 3 keys (vowel, consonant, and pitch) are pressed together, creating difficulty in producing fast real-time performance.

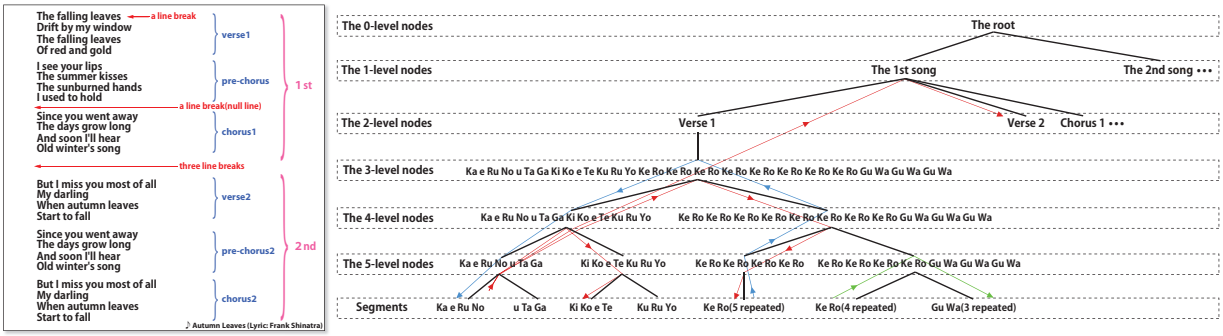

Figure 4: Left: The text format for registering lyrics before performance. We use English here for explanation. The real system only takes Japanese alphabets as input. Right: The hierarchical tree structure of the lyrics segments. The system constructs this structure according to the annotations of the user. The black lines denote the edges. The red, blue and green arrows denote possible jump movements by the user in this tree.

4. TECHNICAL DETAILS OF LYRIC ALIGNMENT

This section explains how we estimate the position that the user wants to perform in the lyrics from the vowel sequences input by the user. In the registration step, the system analyzes the lyrics entered by the user, and constructs a data structure to be used in the performance. In the actual performance step, we estimate the most plausible lyrics using a Hidden Markov Model (HMM) that encodes the behavior of the movements between consecutive characters in the lyrics and jumps during performance. We search the end point of the Viterbi path in this HMM using multi-agent search to find the best matching lyrics for the given vowel sequence. Note that the estimation solely depends on the vowel input, and does not use melody information at all.

4.1 Lyric Registration Step before Performance

Figure 3 shows the user interface view of the system. The user types the lyrics in the the top-left text area and presses the "convert" button to finish. The text format for typing lyrics is shown in Figure 4: left. The system requires the user to annotate rough structures of the song (e.g., repeating, verse, chorus, or several phrases) manually using line breaks. With more line breaks inserted, the system interprets the point as a larger compartmental boundary. After the user completes text entry, the system decomposes the text into morphemes (a sequence of characters) using morphological analysis. We define each delimited morpheme as a "segment".

Commonly, a song has the hierarchical structure that consists of musical phrases. We construct a hierarchical tree structure (called lyrics tree) of the segments using the user's annotations (Figure 4: right). The lyrics tree is constructed by dividing the array of segments recursively from a large structure to a small structure according to the annotations (the smallest structure is a segment). The lyrics tree is used for determining the probabilities of each movement in the lyrics as described in the next subsection.

4.2 The User's Behavior Model for Lyric Movements

We model how the user moves from one character to another character in the lyrics during a performance as an Ergodic Hidden Markov Model (HMM) (Figure 5). In our HMM, an unobserved state corresponds to a position in the lyrics and a transition between unobserved states corresponds to a movement in the lyrics. Each transition generates a specific observable symbol deterministically, which is a vowel of a character. For example, if we have a pre-registered lyrics "Ko-n-Ni-Ti-Wa (segment 1), a-Ri-Ga-To-u (segment 2)", the HMM produces an observable symbol "o" when moving to a state "Ko" or "To".

Each state transition probability of our HMM (probability of a movement in the lyrics) is computed according to the kind of movement. For example, moving to the next character is more likely than a jump to a distant location, and a jump to the head of a sentence is more likely than a jump to the middle of a sentence. The system first enumerates all the possible movements from the current state and sorts them by the likelihood by the kind of movement. For example, in the above example, if you are at "n", possible destinations are "Ni" (next character), "Ti" (skip a character), "n" (stay at same character), "Ko" (previous character), "a" (jump to the head of the next segment) in the order of likelihood. The system then computes the probability of a movement using a monotonically decreasing linear function that takes the position in the sorted list as input and returns probability as an output.

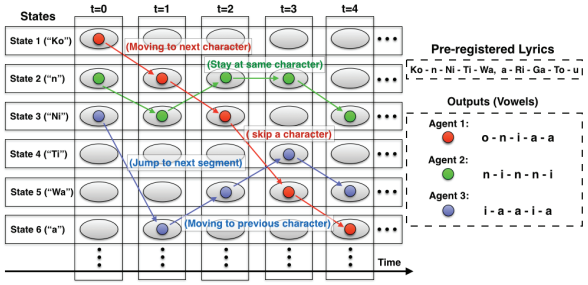

Figure 5: We model the user's movement between two characters in the lyrics during performance as Hidden Markov Model (HMM). A position in the pre-registered lyrics becomes a state.

The likelihood of a movement is computed by traversing the lyrics tree. An example is shown in Figure 4: right. When the current state is one of the character at a leaf node, it can move to the next character within the same leaf node, or move up to parent nodes and then come down to some other leaf node like red, bule and green arrows in Figure 4: right. Each step movement in the tree is associated with a certain cost (e.g. moving to a next character has lower cost than moving to a higher level), and the system accumulates these costs during the traversal.

4.3 Estimation of The Performed Position in Lyrics during Performance

The most plausible position the user wants to perform in the lyrics can be computed as the end point of the Viterbi path that gives the highest accumulated state transition probability (minimum cost) among all the possible paths in the HMM. The Viterbi path is also required to output a vowel sequence that matches with the user inputs. We search this path using a multi-agent search algorithm [3].

The multi-agent search uses multiple interacting intelligent agents for finding the minimum cost path. Each agent is associated with a state in the HMM (a position in the

lyrics), and moves to the next state according to the HMM (color circles in Figure 5). Since multiple destination states exist for a state, the system generates multiple copies of an agent and associates a copy with each destination state. However, if the movement is an impossible one, the system discards the copy. Each agent is scored by the accumulation of the state transition probabilities between the respective pairs of HMM states passed. For example, the score of the agent 1 in Figure 5 is determined by using $\pi_0^1 \times \tau_{12} \times \tau_{23} \times \tau_{35} \times \tau_{56}$ where π_0^1 denotes the initial probability of state 1 and τ_{ij} denotes the state transition probability from state i to state j. If the score of an agent becomes less than a threshold, the system destroys the agent. Additionally, if multiple agents are reached at the same position in the lyrics, the system retains only the agent with the highest score. This procedure corresponds to a the pruning process in dynamic programming. Finally, we obtain the optimum position by selecting an agent has the maximum score at the current time.

5. EVALUATION

5.1 Playability

We examined the playability of our system by a professional pianist. We selected a famous Japanese song "SenbonSakura" (Author: KurousaP) for performance. This song is one of the Japanese songs with the fastest tempo, and contains very fast movements between the characters in the lyrics. We picked this song as a stress test for the system. The time for practice was one week, one hour per day.

A portion of the actual performance scene for this song after practice is shown in the supplemental video. The total length of the performance was four minutes, five seconds. The video shows that this song can be performed at the original tempo. The pianist adds several musical expressions including ad-lib modification of the melody. These expressions can't be performed using any existing methods. Additionally, the system outputted plausible lyrics, even though the pianist often made mistakes and improvisational changes.

In addition, we recruited five participants and played the recorded sound of this performance for them. We requested them to report the number of times they felt an unnatural singing voice sound. No one reported this more than three times.

5.2 Workshop

We held a workshop with ten amateur pianists, all of whom have experimented with piano or other musical keyboards for more than five years. We gave the players a musical score for our system and asked them to practice. We selected a standard jazz song "Autumn Leaves" (Music:Joseph Kosma, Lyrics:Junko Akiyama) with the tempo 120[BPM] for practice. All participants had already known this song before the workshop. We printed the score at this study. The user checked the vocal singing sound only by hearing. The time to practice was three days, 30 minutes per day for all participants.

After the practice, we requested the participants to play the song to an accompaniment for more than six choruses repeatedly, while modifying the melodies and jumping freely ad-lib. As described in §Introduction, most existing approaches, such as melody fitting, can't be used for this scenario. We included this actual scene in the supplemental video. The video shows each pianist playing the song while modifying the melody significantly ad-lib. The participants re-mixed the pieces of the predefined lyrics flexibly. This new type of musical expression is enabled by our system. Note that all participants we employed already had a skill for improvising music by piano before the experiment. So, special trainings for improvisation were not required, even including the control of lyrics.

After the workshop, We conducted individual interviews. At the interviews, we asked three questions. First, about the difficulty of the system: easy, neutral, difficult, impossible. If difficult or impossible, we were going to ask the reason. But, all participants answered it's easy. Second, about the playability. We asked whether the user has any dissatisfactions for expressing the musical phrases they want. Finally, we requested free comments.

Some participants mentioned that they feel discomfort at first because they already have substantial experience playing piano and can read musical scores, but the vowel keys we assigned on the musical keyboard don't correspond to the heard pitches of the synthesized sound. However, they also mentioned that the sense of discomfort decreased gradually as they adjusted to our system.

Nearly all participants said that the system allow input mistakes to some extent and it was a very nice feature. They added that practice for our system was substantially easier than they expected, because the lyrics are represented as a standard musical score. We consider these to be significant merits of our system.

6. CONCLUSION

In this paper, we proposed a practical user interface that enables the use of real-time singing voice synthesizer at an improvisational live performance by inputting the lyrics and melodies of songs simultaneously using a standard musical keyboard. The proposed system allows arbitrary movements within the lyrics including jumping, backtracking, and mistakes by estimating plausible lyrics from vowel sequences using a probabilistic model. Additionally, the lyrics for our system can be described using a standard musical score making it easier to learn. As a result, we achieved very flexible control of lyrics and melody using our system at real-time rate that enabled live improvisational performance of distinctive musical expressions.

7. REFERENCES

[1] d'Alessandro, Nicolas, Wang, J., Pritchard,R., Fels, S. Bringing Bio-Mechanical Modelling of the OPAL Complex as a Mapping Layer for Performative Voice Synthesis, *9th International Seminar on Speech Production (ISSP)*, p.111-118, 2011.

[2] Feugere, L., d'Alessandro, C., Doval, B. Performative voice synthesis for edutainment in acoustic phonetics and singing: a case study using the Cantor Digitalis, *5th International ICST Conference*, p.169-178, 2013.

[3] Goto, M., Muraoka, Y. Beat Tracking based on Multiple-agent Architecture - A Real-time Beat Tracking System for Audio Signals, *The Second International Conference on Multiagent Systems*, p.103-110, 1996.

[4] Kenmochi, H., Ohshita, H. YAMAHA Corporation. VOCALOID - Commercial Singing Synthesizer Based on Sample Concatenation, *INTERSPEECH*, p.4009-4010, 2007.

[5] Mikumin P. Flick input for realtime singing synthesizer, http://www.nicovideo.jp/watch/sm17357529, 2013.

[6] Miwa, M., Sakonda, N. BROTHERS' Button to Phoneme Transfer Standard for International Language, *Departmental Bulletin Paper for Nagoya University of Arts and Science*, Vol.6, p.21-33, 2013.

[7] Takemoto, T., Baba, T., Katayori, H. A Real-Time Singing Generator Using Typing and Humming "Hanautau" Based on an Agile Software Development, *Interaction, Information Processing Society of Japan*, 2014.

[8] Yamamoto, K., Kagami, S., Hamano, K., Kashiwase, K. The Development of a Text Input Interface for Realtime Japanese Vocal Keyboard, *Journal of Information Processing, Vol.21, No.2*, p.274-282, 2013.