Proceedings of the 5th International Conference on Music Information Retrieval (ISMIR 2004),
pp.403-408, October 2004.

SPEECH-RECOGNITION INTERFACES FOR MUSIC INFORMATION RETRIEVAL: "SPEECH COMPLETION" AND "SPEECH SPOTTER"

Masataka Goto[†], *Katunobu Itou*[‡], *Koji Kitayama*[††], *and Tetsunori Kobayashi*[††]
[†] National Institute of Advanced Industrial Science and Technology (AIST), Japan
[‡] Nagoya University, Japan [††] Waseda University, Japan

ABSTRACT

This paper describes music information retrieval (MIR) systems featuring automatic speech recognition. Although various interfaces for MIR have been proposed, speech-recognition interfaces suitable for retrieving musical pieces have not been studied. We propose two different speech-recognition interfaces for MIR, *speech completion* and *speech spotter*, and describe two MIR-based hands-free jukebox systems that enable a user to retrieve and play back a musical piece by saying its title or the artist's name. The first is a music-retrieval system with the *speech-completion* interface that is suitable for music stores and car-driving situations. When a user can remember only part of the name of a musical piece or an artist and utters only a remembered fragment, the system helps the user recall and enter the name by *completing* the fragment. The second is a background-music playback system with the *speech-spotter* interface that can enrich human-human conversation. When a user is talking to another person, the system allows the user to enter voice commands for music-playback control by *spotting* a special voice-command utterance in face-to-face or telephone conversations. Our experimental results from use of these systems have demonstrated the effectiveness of the *speech-completion* and *speech-spotter* interfaces.

Keywords: speech recognition, MIR interface, hands-free MIR, jukebox, title and artist search
Video demonstration: http://staff.aist.go.jp/m.goto/ISMIR2004/

1. INTRODUCTION

The purpose of this study is to build a music-retrieval system with a speech-recognition interface that facilitates both identification of a musical piece and music playback in everyday life. We think a speech-recognition interface is well-suited to music information retrieval (MIR), especially retrieval of a musical piece by entering its title or the artist's name. At home or in a car, for example, an MIR-based jukebox system with a speech-recognition interface would allow users to change background music just by saying the name of a musical piece or an artist. At music-listening stations in music stores or on *karaoke* machines, a speech-recognition interface could also help users find musical pieces they have been looking for without using any input device other than a microphone.

Most previous MIR research, however, has not explored how speech recognition can be used for retrieving music information, although various MIR interfaces using text, symbols, MIDI, or audio signals have been proposed. To retrieve a musical piece, a typical approach is to use a text query related to bibliographic information. This approach requires the use of hand-operated input devices, such as a computer keyboard, mouse, or stylus pen. Another approach is to use a melody-related query given through symbols, MIDI, or audio signals. In particular, music retrieval through a sung melody is called query by humming (QBH), and this approach is considered promising because it requires only a microphone and can easily be used by a novice. However, even though this approach uses a microphone, speech recognition of the names of musical pieces and artists has not been considered.

In this paper, we describe two original speech-recognition interfaces, *speech completion* and *speech spotter*, which are suitable for MIR. Using these interfaces, we have built two MIR-based jukebox systems that allow users to find and play back a musical piece by saying its title or the artist's name.

- Music-retrieval system with the *speech-completion* interface
  This system enables a user to retrieve a musical piece or a list of musical pieces by an artist even if the user can remember only part of the name of the piece or artist. A user cannot similarly enter an incomplete name into current speech recognizers because these recognizers tacitly force the user to utter the entire name carefully and precisely. Our system, on the other hand, allows a user to enter the name by uttering a remembered fragment of it: the system can *complete* the rest (i.e., fill in the missing part) of the partially uttered fragment.
- Music playback system with the *speech-spotter* interface
  This system enables a user to listen to background music by uttering the name of a musical piece or artist while talking to another person. It has been difficult to use current speech recognizers in the midst of human-human conversation because it is difficult to judge, from microphone input only, whether a user is speaking to another person or a speech recognizer. Our sys-

tem, on the other hand, allows a user to enter voice commands for music-playback control in the midst of face-to-face or telephone conversations: the system can *spot* (identify) a special voice-command utterance in human-human conversations without otherwise interrupting the conversations.

In the following sections, we explain both of these speech-capable music jukebox systems and describe their implementation. We then show experimental results which have demonstrated the effectiveness of our speech-recognition interfaces.

2. MUSIC-RETRIEVAL SYSTEM WITH THE SPEECH-COMPLETION INTERFACE

In this system, a user can retrieve a musical piece by uttering a fragment of its title or the artist's name with an intentional *filled pause* (the lengthening of a vowel during hesitation). In human-human conversation, when a speaker cannot remember the entire name of a piece or an artist and hesitates while uttering the name, a listener will sometimes help the speaker recall it: the listener suggests options by *completing* the partially uttered fragment (i.e., by filling in the rest of it). For example, when a speaker cannot remember the last part of a Japanese phrase *"maikeru jakuson"* (in English, "Michael Jackson") [1] and stumbles, saying *"maikeru–"* [2] (in English, "Michael, uh..." or "Michael–") with a filled pause *"ru–"* ("uh..." or "l–"), [3] a listener can help the speaker by asking whether the speaker intends to say *"maikeru jakuson"* ("Michael Jackson"). Our system can provide this completion assistance to users.

The concept of completing a fragment is widely used in text-based interfaces. Several text editors (e.g., Emacs) and UNIX shells (e.g., tcsh and bash), for example, provide functions for completing the names of files and commands. These functions fill in the rest of a partially typed fragment when a completion-trigger key (typically the Tab key) is pressed. Completion functions for pen-based interfaces have also been proposed [1]. However, even though completion is so convenient that it often becomes indispensable to users, an effective completion function for speech input interfaces has not been developed because there has been no effective way to trigger the function during natural speech input and current speech recognizers have difficulty recognizing a partially uttered fragment.

We therefore propose the *speech-completion* interface which completes the names of musical pieces and artists and displays completion candidates so that a user can select the correct one (Figures 1, 2, and 3). The most important point is that we use an intentional filled pause (a vowel-lengthening hesitation like "er...") to trigger this speech-completion function. To prevent this kind of completion assistance becoming annoying, it should be invoked only when a user wants to obtain completion candidates. Because the filled pause is a typical hesitation phenomenon that indicates a user is having trouble thinking of or recalling a subsequent word [2], its use is natural [4] and makes this speech-completion function effective and practical.

This interface provides three benefits:

1. A user can more easily recall poorly remembered names.
2. Less labor is needed to input a long name. For example, you can enter a song title "Supercalifragilisticexpialidocious" (a song from "Mary Poppins") by uttering "Supercalifra–, No.1".
3. The user is not forced to utter the entire name carefully and precisely, as is required by most current speech-recognition systems.

2.1. Search methods

Our music-retrieval system provides two speech-based search methods for a music database: a method of specifying the musical-piece title and a method of specifying the artist's name. For the latter method, the system shows, on the screen, a numbered list of titles for the specified artist in the music database, and a user can select an appropriate title by uttering either the title or its number (Figure 3). After the musical piece is identified by either method, the system plays back its sound file in our current implementation. This interface can also be used for playing back the appropriate standard MIDI file (SMF) or *karaoke* track, making a play list, or editing and downloading music files.

When a user enters the names of musical pieces or artists, the system allows the user to complete speech in either a forward or backward direction:

1. *Forward speech completion* (Figure 1)
   A user who does not remember the last part of a name can invoke this completion by uttering the first part while intentionally lengthening its last syllable (making a filled pause). Here, the user can insert a filled pause at an arbitrary position *(even within a word)*. The user then gets a numbered list of completion candidates whose beginnings acoustically resemble the uttered fragment.
2. *Backward speech completion* (Figure 2)
   A user who does not remember the first part of a name can invoke this completion by uttering the last part after intentionally lengthening the last syllable of a predefined special keyword — called the *wildcard keyword* (In the current implementation, we use the Japanese wildcard *"nantoka"* (in English, "something"), and a user can utter, for example, *"nantoka– jakuson"* ("something– Jackson") [5]). The user then gets a numbered list of completion candidates whose end-

[1] When a foreign name like "Michael Jackson" is written or pronounced in Japanese, it is regularized to conform to the Japanese style: "*maikeru jakuson*."

[2] In this paper, underlining indicates that the underlined syllable is prolonged (a filled pause is uttered).

[3] In Japanese, vowel-lengthening hesitations like "maikeru–" (sounding like "Michael–" in English) are very common, while inserted-filler hesitations like "Michael, uh..." are usually used in English.

[4] This is especially true for Japanese, a moraic language in which every mora ends with a vowel that can be lengthened. In fact, speakers typically use filled pauses to gain time to recall a word or to wait for a listener to help with word choice.

[5] This form of expression is very natural in Japanese.

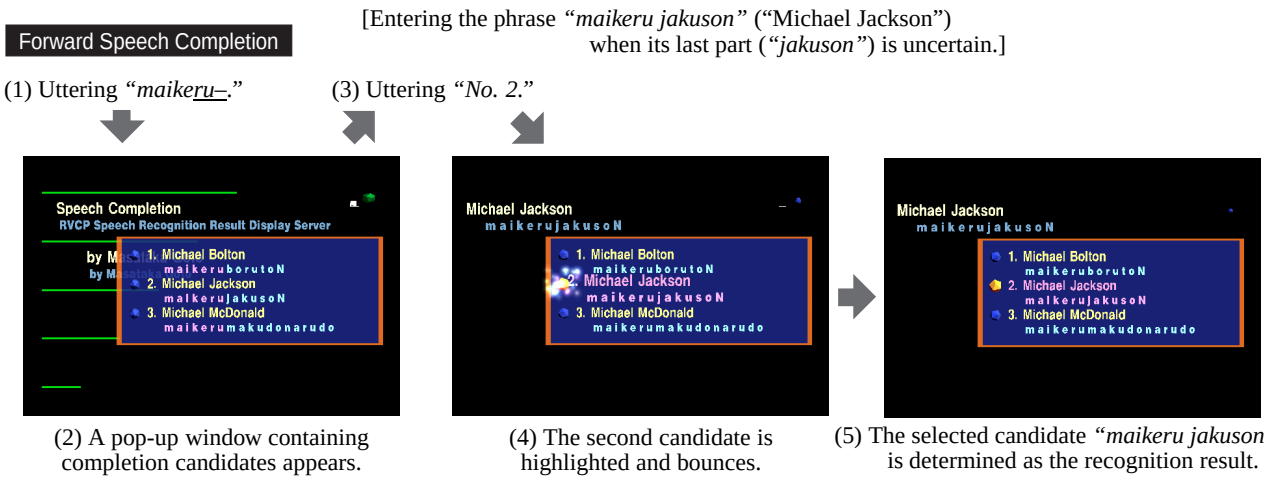

Figure 1. Screen snapshots of forward speech completion.

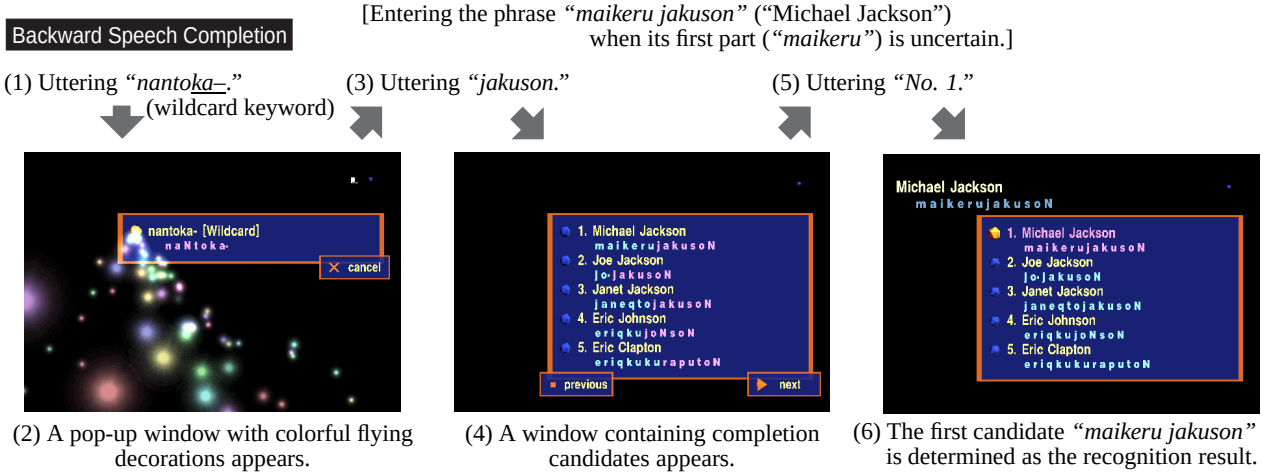

Figure 2. Screen snapshots of backward speech completion.

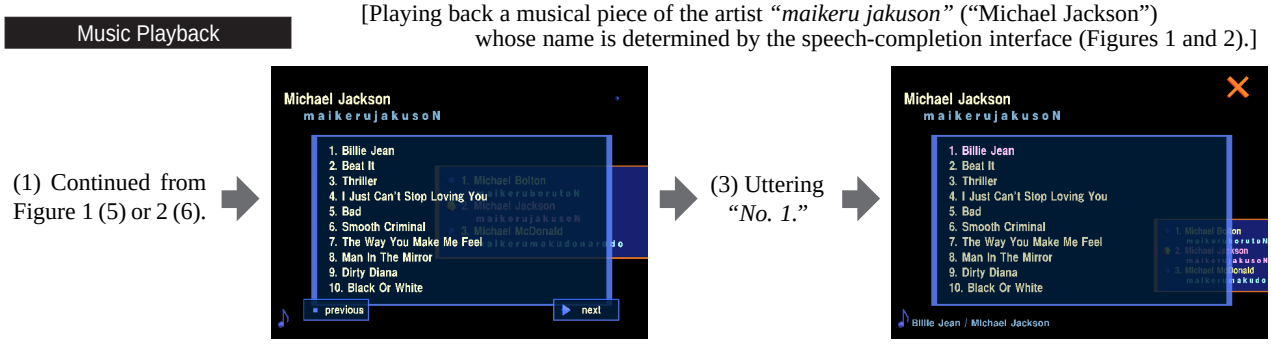

Figure 3. Screen snapshots of music playback.

ings acoustically resemble the uttered last part. Completion candidates are generated by replacing the wildcard keyword (filling in the first part) as if a wildcard search was done.

The user can see other candidates by uttering the turning-the-page phrases, "next candidates" and "previous candidates," displayed whenever there are too many candidates to fit onto the screen. If all the candidates are inappropriate or the user wants to enter another name, the user can simply ignore the displayed candidates and proceed with the next utterance. When the user selects one of the candidates by saying (reading out) either its number, the rest of the name, or the entire name, that name is highlighted and used for the music playback (Figure 3).

2.2. Implementation

Figure 4 shows a block diagram of the method for implementing the speech-completion interface: the two main processes are the *filled-pause detector* (Section 2.2.1) and *speech recognizer* (Section 2.2.2). All the names of artists, musical pieces, and the corresponding sound files

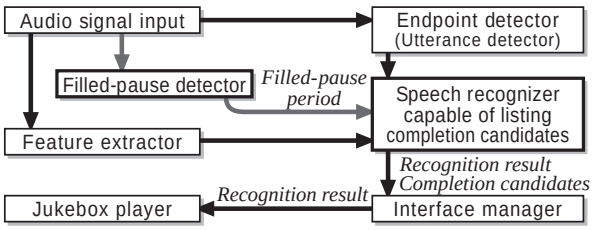

Figure 4. Method for implementing the speech-completion interface.

are stored on an SQL (Structured Query Language) database server. Each name of the artists and their pieces is also registered as a single word in the system vocabulary of the speech recognizer. Note that we cannot convert an uttered fragment of a vocabulary word into text by using an up-to-date HMM-based speech recognizer [6] because the recognizer only accepts a combination of vocabulary words: it is therefore necessary to extend the speech recognizer to deal with word fragments.

2.2.1. Filled-pause detector

To detect filled pauses in real time, we use a robust filled-pause detection method [3]. This is a bottom-up method that can detect a lengthened vowel in any word through a sophisticated signal-processing technique. It determines the beginning and end of each filled pause by finding two acoustical features of filled pauses — small fundamental frequency (voice pitch) transitions and small spectral envelope deformations.

2.2.2. Speech recognizer capable of listing completion candidates

We extended a typical speech recognizer to provide a list of completion candidates whenever a filled pause was detected [4, 5]. Because single phonemes cannot be recognized accurately enough, up-to-date speech recognizers do not determine a word's phoneme sequence phoneme by phoneme. Instead, they choose the maximum likelihood (ML) hypothesis while pursuing multiple hypotheses on a vocabulary tree where all vocabulary words (i.e., names of artists and musical pieces) are stored. When the beginning of a filled pause is detected, the recognizer determines which completion method is to be invoked (forward or backward) on the basis of whether the wildcard keyword is the ML hypothesis at that moment.

In forward speech completion, completion candidates are obtained by deriving from the vocabulary tree those words that share the prefix corresponding to each incomplete word hypothesis for the uttered fragment: the candidates are obtained by tracing from the top 15 hypotheses to the leaves on the tree.

In backward speech completion, it is necessary to obtain completion candidates by recognizing a last-part fragment uttered after the wildcard keyword, which is not registered as a vocabulary word. We therefore introduced an *entry node table* that lists the roots (notes) from which the speech recognizer starts searching. Just after the wildcard keyword, every syllable in the middle of all the vocabulary words is temporarily added to the table. Then, after the last-part fragment is uttered, the hypotheses that have reached leaves are considered completion candidates.

[6] If this text conversion was possible, the problem we are solving would be similar to text-based completion and much easier. However, there is no such thing as a universal speech (phonetic) typewriter.

Speaker A: Let's change the background music.
Speaker B: How about Michael Jackson?
Speaker A: Uhm..., okay. I like his hit song from 1991.
Speaker B: Yeah, the title is "Black or White."
Speaker A: Uhm..., Black or White. *(with a high pitch)*
The system then plays the song "Black or White."

Figure 5. An example of using the music playback system with the speech-spotter interface.

3. MUSIC PLAYBACK SYSTEM WITH THE SPEECH-SPOTTER INTERFACE

In this system, a user can listen to background music by saying its title or the artist's name while talking to another person, as illustrated in Figure 5. For this system to be practical, it must be able to monitor human-human conversations without disturbing them and provide music playback only when asked for it. We think such on-demand music playback assistance in human-human conversation is useful and convenient because it does not require the use of any input device other than a microphone.

It has been difficult, however, to achieve a practical means of providing such assistance by using only microphone input. Previous approaches using only speech information detected keywords in speech signals by means of word-spotting technology [6, 7]. These techniques, though, are poor at judging, without the context being restricted in advance, whether the detected keywords are intended to be *command utterances* (voice commands for music-playback control) for a computer system or *conversational utterances* for a conversational partner. Although there have been other spotting approaches which required that an utterance intended for the system be preceded by a keyword, such as *Computer*, *Casper*, or *Maxwell*, this restricted the usual behavior of a user: the user was forced to avoid use of the keyword in human-human conversation in front of the microphone. Other previous speech-interface systems have had to use other input devices such as a button or a camera [8, 9]. In short, no previous approach allowed a system to identify a *command utterance* in conversation without the context being restricted or some other device being used.

We therefore developed the *speech-spotter* interface which enables a user to request music playback only when desired while talking to another person. This interface regards a user utterance as a *command utterance* only when it is intentionally uttered with a high pitch just after a filled pause such as "er..." or "uh...". In other words, a computer system accepts this specially designed unnatu-

ral utterance only and ignores other normal utterances in human-human conversation. For example, when a user enters the title of a musical piece by saying "Er... *(a filled pause)*, Black or White *(an utterance with a high pitch)*",[7] the system plays back the corresponding sound file. For this speech-spotter utterance, we use the unnaturalness of nonverbal speech information — in this case an intentional filled pause and a subsequent high-pitch utterance — because this combination is not normally uttered in (Japanese) human-human conversation but nevertheless can be easily uttered.

This interface provides three benefits:

1. In human-human conversation, speech-based assistance can immediately be used whenever needed.
2. A hands-free interface system with only a microphone is achieved. A user is free regarding body movement and can use the system even during a telephone conversation.
3. A user can feel free to use any words in conversation with another person. The user does not have to carefully avoid saying anything that the system will accept as input.

3.1. Search methods

Our music playback system supports the following search methods:

- *Specifying the musical-piece title*
  When the title of a musical piece is uttered, such as "Er..., Without You", the system plays back the corresponding sound file. It also either shows the title on the screen, has a speech synthesizer read out the title, or both.
- *Specifying the artist's name*
  When the name of an artist is uttered, such as "Uhm..., Mariah Carey", the system shows a numbered list of musical-piece titles for that artist on the screen or has the speech synthesizer read out the list. After the user selects a musical piece by saying the speech-spotter utterance of either the title or its number, the system plays back the piece. It also highlights the selected title or reads out the title.

The system allows the user to say speech-spotter utterances at any time by overlapping and interrupting the speech synthesis or music playback. The user, for example, can stop the music playback by saying "Uh..., stop", or change the current piece by saying another title.

This system is useful not only when a user would like to enjoy background music, but also when a user would like to talk about music in telephone conversations while listening to it; note that the system does not disturb such conversations. In particular, this is very effective for people who like to listen to music in everyday life because it makes it much easier for them to share background music and discuss it during playback on the telephone. The system can also be used to change background music in an actual room where people are talking.

[7] In this paper, overlining indicates that the pitch of the underlined words is intentionally raised by a user.

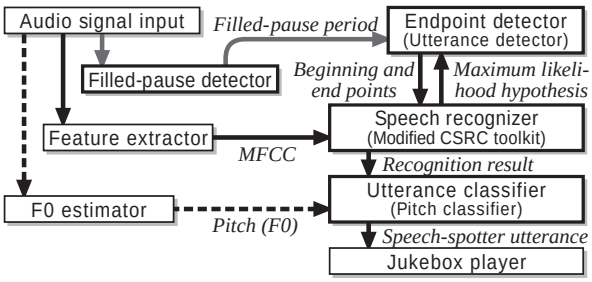

Figure 6. Method for implementing the speech-spotter interface.

3.2. Implementation

Figure 6 shows a block diagram of the method for implementing the speech-spotter interface. The four main processes are the *filled-pause detector* (Section 2.2.1), *endpoint detector* (Section 3.2.1), *speech recognizer* (Section 3.2.2), and *utterance classifier* (Section 3.2.3). Like the implementation of the speech-completion interface, all the names are stored on the SQL database server and each name is registered as a single word in the system vocabulary.

Speech-spotter utterances can be detected through the following four steps:

1. Each filled pause is detected by the *filled-pause detector* described in Section 2.2.1.
2. When triggered by a detected filled pause, the *endpoint detector* determines the beginning of an utterance.
3. While the content of the utterance is being recognized by the *speech recognizer*, the end of the utterance is automatically determined by the *endpoint detector*.
4. The average pitch of the utterance whose beginning and end points were determined above is judged to be high or normal by the *utterance classifier*.

3.2.1. Determining the beginning of an utterance (endpoint detector)

Whenever a filled pause is detected, the beginning of the subsequent utterance is determined as being 130 ms before the end of the filled pause — i.e., as being in the middle of the filled pause. Every lengthened vowel (and subsequent consonant if necessary) should be inserted at the beginning of the grammar.

3.2.2. Determining the end of an utterance (endpoint detector and speech recognizer)

After the HMM-based *speech recognizer* starts decoding the current utterance, the *endpoint detector* checks the ML hypothesis (intermediate speech-recognition result) for every frame. In the frame-synchronous Viterbi beam search, if the ML hypothesis stays at a unique node that is not shared by other words in a tree dictionary or a silence node that corresponds to the silence at the end of a sentence, its frame is considered the end of the utterance and the *speech recognizer* stops decoding [10].

3.2.3. Judging the voice pitch (utterance classifier)

On the basis of a speaker-independent pitch-classification method using a threshold relative to the *base fundamental frequency (base F0)* [11], the *utterance classifier* filters out normal-pitch utterances to obtain high-pitch speech-spotter utterances. The base F0 is a unique pitch reference that corresponds to the pitch of the speaker's natural voice and can be estimated by averaging the voice pitch during a filled pause. If the *relative pitch value* of an utterance, which is calculated by subtracting the base F0 from the pitch averaged over the utterance, is higher than a threshold, the utterance is judged to be a *speech-spotter utterance*.

4. EXPERIMENTAL RESULTS

We describe the results from evaluating the effectiveness of the system with the speech-completion interface and the system with the speech-spotter interface.

4.1. Evaluation of the speech-completion interface

We tested the system with 45 Japanese subjects (24 male, 21 female). To evaluate whether the subjects preferred to use speech completion after gaining a good command of it, we measured the usage frequencies of speech completion under two conditions: (a) when a subject input a set of name entries from a list by freely using speech completion according to personal preference, and (b) when a subject had to recall and input vaguely remembered entries.

We found that the average usage frequency of speech completion was 74.2% and 80.4%, respectively, for conditions (a) and (b). These results showed that the subjects preferred to use the speech-completion function even when they could choose not to use it. Subjective questionnaire results indicated that the speech-completion interface was helpful and easy to use, and made it easy to recall and input uncertain phrases.

4.2. Evaluation of the speech-spotter interface

We analyzed the detection performance for speech-spotter utterances on a 40-minute corpus consisting of both normal utterances of sentences naturally spoken with spontaneous filled pauses and speech-spotter utterances of 218 names of musicians and songs by 12 Japanese subjects.

We found that the recall and precision rates for detecting speech-spotter utterances were 0.78 and 0.77, respectively. In our experience with the music playback system, users without any training were able to start playback of background music and change it while talking on cellular or normal phones. They felt that the practical performance was much higher than the above rates indicate because visual feedback enabled the users to know how long a vowel should be lengthened during a filled pause. Although melody ringers (cellular phone ring-tones) are widely used, our users had no previous experience of listening to music in the midst of telephone conversation, and appreciated its novelty and usefulness.

5. CONCLUSION

We have described two speech-recognition interfaces suitable for MIR, *speech completion* and *speech spotter*, and demonstrated their usefulness in two different music jukebox systems. The music-retrieval system with the *speech-completion* interface enables a user to listen to a musical piece even if part of its name cannot be recalled. The music playback system with the *speech-spotter* interface enables users to share music playback on the telephone as if they were talking in the same room with background music. As far as we know, this is the first system that people can use to obtain speech-based music information assistance *in the midst of a telephone conversation*.

We believe that practical speech-recognition interfaces for MIR cannot be achieved by simply applying the current automatic speech recognition to MIR: retrieval of musical pieces just by uttering entire titles or artist names is not sufficient. The two interfaces described in this paper can be considered an important first step toward building the ultimate speech-capable MIR interface. It will become more and more important to explore various speech-recognition interfaces for MIR as well as other traditional MIR interfaces.

In the future, we plan to build a unified system where a user can use, from a single microphone input, both of the systems described in this paper as well as a query-by-humming (QBH) system to leverage the potential affinity between speech-recognition interfaces and QBH systems.

6. REFERENCES

[1] T. Masui. An efficient text input method for pen-based computers. *Proc. of CHI'98*, pp. 328–335, 1998.

[2] E. Shriberg. To 'errrr' is human: ecology and acoustics of speech disfluencies. *Journal of the International Phonetic Association*, 31(1):153–169, 2001.

[3] M. Goto, K. Itou, and S. Hayamizu. A real-time filled pause detection system for spontaneous speech recognition. *Proc. of Eurospeech '99*, pp. 227–230, 1999.

[4] M. Goto, K. Itou, T. Akiba, and S. Hayamizu. Speech completion: New speech interface with on-demand completion assistance. *Proc. of HCI International 2001*, volume 1, pp. 198–202, 2001.

[5] M. Goto, K. Itou, and S. Hayamizu. Speech completion: On-demand completion assistance using filled pauses for speech input interfaces. *Proc. of ICSLP 2002*, pp. 1489–1492, 2002.

[6] J. R. Rohlicek, W. Russell, S. Roukos, and H. Gish. Continuous hidden Markov modeling for speaker-independent word spotting. *Proc. of ICASSP 89*, pp. 627–630, 1989.

[7] T. Kawahara, K. Ishizuka, S. Doshita, and C.-H. Lee. Speaking-style dependent lexicalized filler model for key-phrase detection and verification. *Proc. of ICSLP 98*, pp. 3253–3256, 1998.

[8] K. Nagao and A. Takeuchi. Social interaction: Multimodal conversation with social agents. *Proc. of AAAI-94*, volume 1, pp. 22–28, 1994.

[9] Y. Matsusaka *et al.* Multi-person conversation via multi-modal interface — a robot who communicate with multi-user. *Proc. of Eurospeech '99*, pp. 1723–1726, 1999.

[10] K. Kitayama, M. Goto, K. Itou, and T. Kobayashi. Speech starter: Noise-robust endpoint detection by using filled pauses. *Proc. of Eurospeech 2003*, pp. 1237–1240, 2003.

[11] M. Goto, Y. Omoto, K. Itou, and T. Kobayashi. Speech shift: Direct speech-input-mode switching through intentional control of voice pitch. *Proc. of Eurospeech 2003*, pp. 1201–1204, 2003.