Speech Starter: Noise-Robust Endpoint Detection by Using Filled Pauses

Koji Kitayama†*, Masataka Goto*‡*, Katunobu Itou*‡ *and Tetsunori Kobayashi*†

†Dept. EECE, Waseda Univ.,
3-4-1 Okubo, room 55N-509, Shinjuku-ku, Tokyo 169-8555, JAPAN.
‡ National Institute of Advanced Industrial Science and Technology (AIST)
1-1-1 Umezono, Tsukuba, Ibaraki 305-8568, JAPAN.

† {kitayama,koba}@tk.elec.waseda.ac.jp, ‡ {m.goto@,itou@ni.}aist.go.jp

Abstract

In this paper we propose a speech interface function, called *speech starter*, that enables noise-robust endpoint (utterance) detection for speech recognition. When current speech recognizers are used in a noisy environment, a typical recognition error is caused by incorrect endpoints because their automatic detection is likely to be disturbed by non-stationary noises. The speech starter function enables a user to specify the beginning of each utterance by uttering a filler with a filled pause, which is used as a trigger to start speech-recognition processes. Since filled pauses can be detected robustly in a noisy environment, practical endpoint detection is achieved. Speech starter also offers the advantage of providing a hands-free speech interface and it is user-friendly because a speaker tends to utter filled pauses (e.g., "er...") at the beginning of utterances when hesitating in human-human communication. Experimental results from a 10-dB-SNR noisy environment show that the recognition error rate with speech starter was lower than with conventional endpoint-detection methods.

1. Introduction

Noise-robust speech endpoint detection is important to achieve practical speech recognition in a noisy real-world environment. Most current speech recognizers recognize utterances after their endpoints are detected by using signal processing techniques. Typical automatic endpoint-detection methods are based on two acoustical features: zero crossing rates and short time energy [1]. While those methods are useful in a silent environment, they are not robust in noisy environments because the acoustical features are likely to be disturbed by noises. The incorrectly detected endpoints cause serious recognition errors: given wrong utterance periods, the recognizers cannot obtain appropriate recognition results.

A typical approach to solving this endpoint-detection problem is to use a button: if a user presses the button while speaking, an utterance is accepted by a speech recognizer. This approach, however, is disadvantageous in that we cannot build a hands-free interface and it is still not robust in a noisy environment when a user may press or release the button too early or too late.

Other major research approaches have improved the use of acoustical features for detecting endpoints [2, 3, 4], but there is still room for improvement in performance with regard to non-stationary burst noises. While a continuous speech recognition method that does not need explicit endpoint detection was proposed [5], it is difficult for a user to anticipate or control which utterances can be accepted by the speech recognizer. Although a method of using a speaker's facial motions has succeeded in detecting endpoints in a noisy environment [6], it cannot be used unless a camera is available.

We propose a new speech interface function, called *speech starter*, which solves this problem by making full use of nonverbal speech information, a *filled pause* (the lengthening of a vowel during hesitation). The filled pause is a natural hesitation that indicates a speaker is having trouble preparing (thinking of) a subsequent utterance. Speakers sometimes utter fillers with a filled pause, such as "er..." or "uh...", at the beginning of an utterance[1]. We therefore use the filled pause as a trigger to start speech recognition: each filled pause is regarded as the beginning of an utterance by a speech recognizer. Since filled pauses can be detected robustly in a noisy environment [7], the speech-starter function achieves robust endpoint detection on a hands-free speech-input interface. The most important point is that we use *intentional* filled pauses for the speech-starter function: a user must utter a filled pause at the beginning of each utterance.

In the following sections, we explain the basic concept and design of speech starter and then describe the implementation of a speech interface system with the speech-starter function. Finally, we show experimental results from various noisy environments that demonstrate the robustness of speech starter.

2. Speech starter

Speech starter is an endpoint-control interface function that enables a user to explicitly specify the beginning of an utterance without using any other device — i.e., by using only voice. The rule of using this function is that a user must always utter a filled pause at the beginning of each utterance: the user can use an arbitrary filler as long as it contains a filled pause. For example, if a user wants to enter a word "Michael Jackson," the user must say "er... Michael Jackson." The speech-starter function provides three benefits:

1. Noise-robust endpoint detection
   Because the short time energy of vowels tends to be high, filled pauses, the lengthened vowels, are likely to have high energy in speech signals and can be detected robustly in a noisy environment. This enables a speech recognizer to decode an utterance of noisy speech by starting from the appropriate beginning of the utterance. Since the recognizer does not start until a filled pause is detected, it can also reject non-stationary burst noises:

[1] This is especially true for Japanese speakers: when speakers hesitate and think of how to start, they tend to begin their utterances with Japanese fillers with a filled pause, such as /e-/, /ano-/, and /n-/, especially in a formal but unprepared speech presentation.

those noises do not contain lengthened vowels (filled pauses) in general.

2. User-friendly interface

   Although a user intentionally utters the filled pause for the speech-starter function, it sounds natural and a user can do it without any training because a speaker sometimes does the same thing while hesitating in human-human communication.

3. Microphone only

   The speech-starter function does not require other devices such as buttons or cameras: only a microphone is needed to trigger the start of the speech-recognition process.

In addition to a typical speech recognizer, the speech-starter function requires the following three processes:

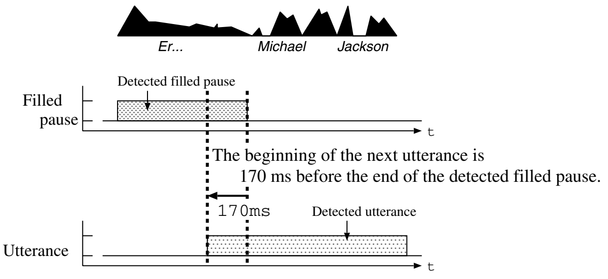
1. Detection of filled pause in real time
2. Determination of the endpoint (the beginning of an utterance) at which the speech recognizer starts decoding
3. Determination of the endpoint (the end of the utterance) at which the speech recognizer stops decoding

2.1. Detecting a filled pause

To detect filled pauses in real time, we use a robust filled-pause detection method [7]. This is a bottom-up method that can detect a lengthened vowel in any filler 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 transitions and small spectral envelope deformations.

2.2. Determining the beginning of an utterance

When a filled pause is detected, the beginning of the next utterance is determined by using the end of the filled pause. Figure 1 shows how the beginning of an utterance is determined. In our current implementation, the beginning of the utterance is determined as being 170 ms before the end of the filled pause. Since the timing of the beginning of the utterance is during the filled pause, a speech recognizer can start decoding at a stable vowel.



Detected filled pause

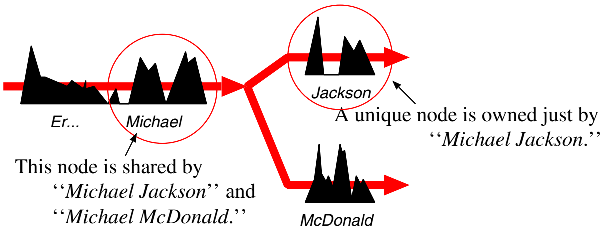
The beginning of the next utterance is 170 ms before the end of the detected filled pause.

Figure 1: The beginning of an utterance.

2.3. Determining the end of an utterance

After the speech recognizer starts decoding, the end of the current utterance must be determined. We determine it by using an intermediate speech-recognition result, the maximum likelihood hypothesis in speech recognition. If the maximum likelihood hypothesis stays at one of the following two types of nodes during the frame-synchronous onepass Viterbi beam search, its frame is considered the end of the utterance and the speech recognizer stops decoding.

1. The maximum likelihood hypothesis stays at a unique node [8] that is not shared by other words in a tree dictionary (Fig. 2). In other words, it stays at a node that is owned by a single word.
2. The maximum likelihood hypothesis stays at a silence node that corresponds to the silence at the end of a sentence (Fig. 3).



A unique node is owned just by *"Michael Jackson."*

This node is shared by *"Michael Jackson"* and *"Michael McDonald."*

Figure 2: Determining the end of an utterance: a speech recognizer stops decoding after staying at a unique node for a while.

A silence node corresponds to the silence at the end of a sentence.

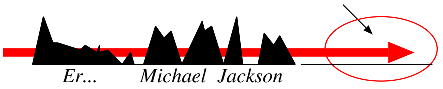


Figure 3: Determining the end of an utterance: a speech recognizer stops decoding after staying at a silence node for a while.

3. Implementation

Figure 4 shows the architecture of our speech-starter system. Each of the boxes in the figure represents a different process. These can be distributed over a LAN (Ethernet) and connected by using a network protocol called *RVCP (remote voice control protocol)* [9, 10]. The speech recognizer is implemented by modifying the CSRC (continuous speech recognition consortium) Japanese dictation toolkit [11] (*julian 3.3beta* speech recognition engine). At each frame, the speech recognizer sends word hypotheses to the endpoint detector.

The input audio signal is analyzed by both the filled-pause detector using Goto's method [7] and the feature extractor obtaining the MFCC features. When the endpoint detector receives the end of each filled pause, it determines the beginning of an utterance and sends it to the speech recognizer. The speech recognizer receives the MFCC features, decodes them after receiving the beginning of the utterance, and sends word hypotheses to the endpoint detector. The endpoint detector receives the hypotheses and judges whether one of the conditions for the end of the utterance is satisfied; if it is satisfied, the end of the utterance is determined. Finally, the graphics manager displays the speech recognition results.

4. Experiments on isolated word recognition

To evaluate the effectiveness of the speech-starter function for robust speech recognition in a noisy environment, we compared the following three endpoint detection methods under various noisy environments:

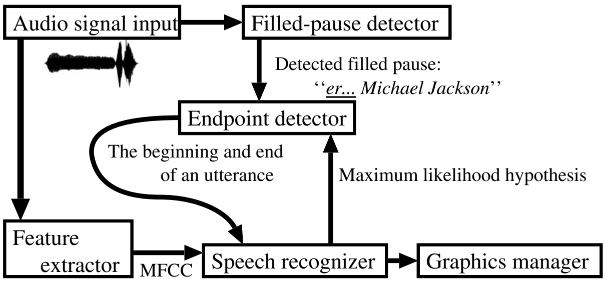


Figure 4: System architecture.

1. Speech starter
2. Method of using zero crossing rates and short time energy
3. Short pause segmentation method [12] that *julian* [11] supports for decoding without needing explicit endpoint detection.

For the experiments, we used a system vocabulary comprising 521 entries (names of 179 Japanese musicians and 342 of their songs), which were collected from Japanese hit charts during fiscal 2000 [9, 10].

4.1. Experimental setup

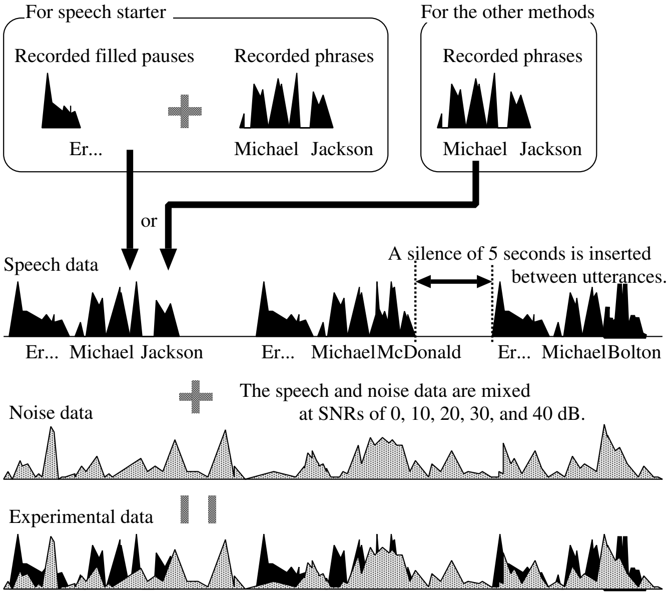


Figure 5: How the speech-shift function was tested in a noisy environment.

Figure 5 shows how we prepared the audio data used to evaluate the three methods. We recorded 179 Japanese phrases and 11 Japanese filled pauses (/e-/) spoken by a Japanese male speaker. For the speech starter, we prepared utterances by concatenating a recording of a filled pause and a recording of a phrase. To evaluate the performance of isolated utterances, we inserted a silence (with stationary background noise) of five seconds between utterances.

The speech data were mixed with seven types of real-world noises[13] (in a running car [1500cc class], an event hall [in a booth], an event hall [aisle], at a crossroads, in a train [old railroad line], a computer room [workstation], and an elevator hall [department store]) at five different SNRs (0, 10, 20, 30, and 40 dB). The SNR was calculated by using the average energy of signals and noises during hand-labeled correct utterances. The noises we used for these tests were non-stationary: it was difficult to recognize words at low SNRs.

The acoustic features were 12 MFCCs, 12 ΔMFCCs, and 1 Δpower. Cepstrum mean normalization (CMN) was not used. The acoustic models were trained with 20414 sentences from the ASJ speech database of phonetically balanced sentences (ASJ-PB) and newspaper article sentences (ASJ-JNAS) [14].

Figure 6 shows the grammar for speech starter, and Fig. 7 shows the grammar for the other two methods. For speech starter, the grammar begins with the filled-pause node that corresponds to the middle of 14 Japanese fillers, such as "e-", "n-" and "u-." For the other methods, the grammar begins with the silence node.

The threshold needed by the method using zero crossing rates and short time energy was determined to maximize the endpoint detection performance for another learning data set. The data set was a 20dB-SNR mixture of speech data and noise, "an event hall [in a booth]"; we inserted a silence (with stationary background noise) of three seconds between utterances.

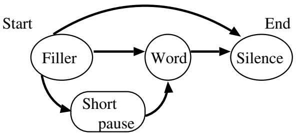


Figure 6: Grammar for speech starter.

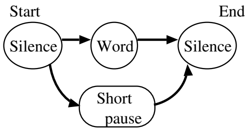


Figure 7: Grammar for other methods.

4.2. Evaluation measure

We compared the system output (utterances and their recognition results) with the hand-labeled correct words (utterances). The degree of matching between the recognized and correct words was evaluated by using the F-measure, which is the harmonic mean of the recall rate (R) and the precision rate (P):

$$\text{F-measure} = \frac{2RP}{R+P} \tag{1}$$

$$R = \frac{\text{the number of words recognized correctly}}{\text{the number of correct words (179 words)}} \tag{2}$$

$$P = \frac{\text{the number of words recognized correctly}}{\text{the number of detected utterances}} \tag{3}$$

We judged that the system output was correct if the recognition results (words) were correct, but we ignored mistakes regarding the type of filler; for example, even if "er..." was recognized as "ah...," we judged it to be correct.

4.3. Experimental results

Figures 8-14 show evaluation results regarding the speech-recognition accuracy in seven noisy environments. Figures 8, 10, 13, and 14 show that speech starter provided the best performance at SNRs of 0 and 10 dB: speech starter provided improved performance in very noisy conditions, including low-frequency background noise and music. In Figs. 8-10, the short pause segmentation method gave many false alarms. Figure 11 shows that the results with speech starter were a bit better than with the other methods for the SNR of 0 and 10 dB. In Fig. 12, the three methods performed almost equally well. These results show that speech starter is robust enough to detect endpoints (utterances) in very noisy environments, especially at SNRs of 0 and 10 dB, and is a practical interface function.

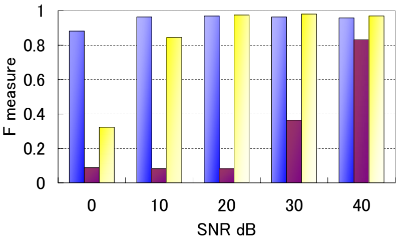


Figure 8: In a running car [1500cc class].

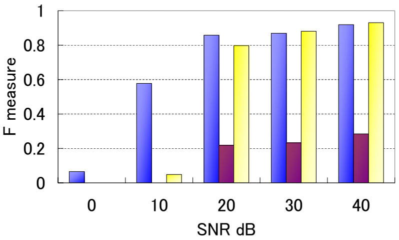


Figure 9: An event hall [in a booth].

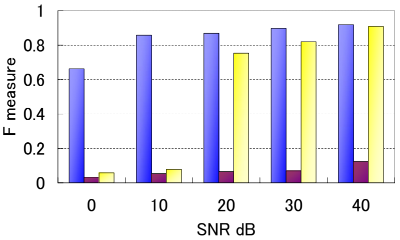


Figure 10: An event hall [aisle].

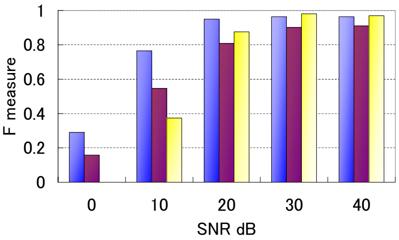


Figure 11: At a crossroads.

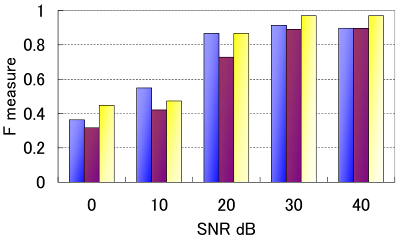


Figure 12: In a train [old railroad line].

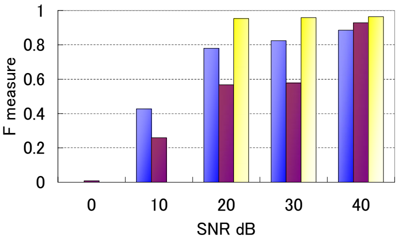


Figure 13: A computer room [workstation].

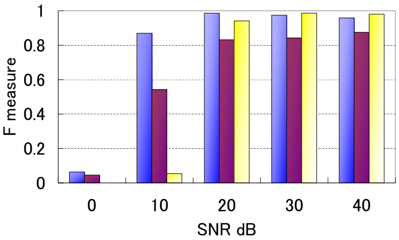


Figure 14: An elevator hall [department store].

🟦 Speech starter
🟪 Short pause segmentation
🟨 Zero crossing rate and energy

5. Conclusion

We have described a speech interface function *"speech starter,"* which uses intentional nonverbal speech information, a filled pause, to enable a user to specify the beginning of an utterance by using only the voice. The idea of making full use of intentional nonverbal information in interface functions originated from research on "speech completion" [9, 10] and "speech shift," which was followed by this research on "speech starter." We regard the speech starter function as a "speech switch" that is a substitute for the mechanical switch of a microphone. Speech starter is useful especially for voice-enabled applications that require hands-free control. In our futer work, we plan to develop practical applications and evaluate the usability of speech starter.

6. References

[1] L. R. Rabiner and M. R. Sambur, "An algorithm for determining the endpoints of isolated utterances," *The Bell System Technical Journal*, vol. 54, no. 2, pp. 297–315, 1975.

[2] S. E. Bou-Ghazale and K. Assaleh, "A robust endpoint detection of speech for noisy environments with application to automatic speech recognition," in *Proc. ICASSP2002*, vol. 4, 2002, pp. 3808–3811.

[3] A. Martin, D. Charlet, and L. Mauuary, "Robust speech / non-speech detection using LDA applied to MFCC," in *Proc. ICASSP2001*, vol. 1, 2001, pp. 237–240.

[4] L. sheng Huang and C. ho Yang, "A novel approach to robust speech endpoint detection in car environments," in *Proc. ICASSP*, vol. 3, 2000, pp. 1751–1754.

[5] O. Segawa, K. Takeda, and F. Itakura, "Continuous speech recognition without end-point detection," in *Proc. ICASSP*, vol. 1, 2001, pp. 245–248.

[6] K. Murai, K. Kumatani, and S. Nakamura, "A robust end point detection by speaker's facial motion," in *International Workshop on Hands-Free Speech Communication*, 2001, pp. 199–202.

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

[8] N. Inoue, M. Nakamura, S. Sakayori, S. Yamamoto, and F. Yato, "Fast speech recognition method by using likelihood comparison of word unique cells," *Trans. IEICE (in Japanese)*, vol. J79-D-II, no. 12, pp. 2110–2116, 1996.

[9] M. Goto, K. Itou, and S. Hayamizu, "Speech completion: On-demand completion assistance using filled pauses for speech input interfaces," in *Proc. ICSLP2002*, 2002, pp. 1489–1492.

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

[11] T. Kawahara, A. Lee, T. Kobayashi, K. Takeda, N. Minematsu, S. Sagayama, K. Itou, A. Ito, M. Yamamoto, A. Yamada, T. Utsuro, and K. Shikano, "Free software toolkit for japanese large vocabulary continuous speech recognition," in *Proc. ICSLP2000*, vol. 4, 2000, pp. 476–479.

[12] A. Lee, T. Kawahara, and K. Shikano, "Julius — an open source real-time large vocabulary recognition engine," in *Proc. Eurospeech2001*, vol. 3, 2001, pp. 1691–1694.

[13] S. Itahashi, "The noise database and japanese common voice data the DAT version," *The journal of the acoustical society of japan (in Japanese)*, vol. 47, no. 2, pp. 951–953, 1991.

[14] K. Itou, M. Yamamoto, K. Takeda, T. Takezawa, T. Matsuoka, T. Kobayashi, K. Shikano, and S. Itahashi, "The design of the newspaper-based japanese large vocabulary continuous speech recognition corpus," in *Proc. ICSLP98*, 1998, pp. 3261–3264.