Development of Speech Input System for Web-based Courseware

Ryuichi Nisimura (nisimura@itakura.nuee.nagoya-u.ac.jp), Shoji Kajita (kajita@media.nagoya-u.ac.jp), Kazuya Takeda (takeda@nuee.nagoya-u.ac.jp) and Fumitada Itakura (itakura@media.nagoya-u.ac.jp)

1 School of Engineering, Nagoya University, Nagoya-City, Japan
2 Center for Information Media Studies, Nagoya University, Nagoya-City, Japan
3 Graduate School of Engineering, Nagoya University, Nagoya-City, Japan
4 Center for Information Media Studies, Nagoya University, Nagoya-City, Japan

Abstract

This paper describes an Web-based speech input and its application system, WebSPEAC system, for digital signal processing on-line courses. The final goal of the development is to facilitate Web-based speech input for students and also the creation of sophisticated signal processing CGI programs for non-technical designers. The WebSPEAC system consists of WebSPEAC server and the client. The server is a CGI program and generates a task specific multipart document in the MIME message format "multipart/mixed" so as to enable students to communicate with the server naturally, i.e., without needless clickings. WebSPEAC client is a helper program executed from Web browser. The client is used to only record user's voice and automatically send it back to WebSPEAC server through UNIX socket connections. In this paper, two examples of on-line course content using WebSPEAC system for a digital signal speech processing are demonstrated in order to show how WebSPEAC system works.

Keywords

speech input, courseware, WebCT, server push, digital signal processing, speech analysis and recognition

Introduction

The recent progress of Web technologies have helped us access and retrieve multimedia contents on the Internet. By just clicking on Web browser, anyone can easily read any text and also play any sounds and videos through the internet. This simple way to access multimedia contents is one of the reasons for the widespread use of Web. However, in the case of sending back multimedia information from a user to a Web server, the method is not always satisfactory. In particular, there is no established way to automatically send back user's speech signal (i.e., voice) or video to a Web server. Needless to say, WebCT (Goldberg, 1997) based on such current Web technologies also has the same problem. This paper addresses this problem, especially for speech.

In online courses related to Digital Signal Processing (DSP) in engineering, it would be better to send back a student's "speech" to the WebCT server, because the speech itself is a very good example of a real world signal. For example, we can display the sound spectrogram, which is a kind of time-frequency pattern, of his speech to teach what temporal and frequency characteristics the signal has. As described above, however, it is troublesome to send back the speech to the WebCT server, as students have to:

1. execute a sound recorder manually on their computers,
2. record their speech using the sound recorder through a microphone,
3. save it in a file,
4. upload the file to WebCT server.

Furthermore, the course designer has to prepare a CGI (Common Gateway Interface) program so as to receive the speech using any file managers, perform spectral analysis on the speech, and return the results to the student's browser, as shown in Figure 1. This CGI programming is not easy for non-technical designers. Thus, it is required to develop a system that facilitates an automatic speech input for students and also the creation of sophisticated CGI for non-technical designers, which can perform some signal processing for the speech signal. To achieve the goal, WebSPEAC uses 1) the server-client mechanism and 2) the multipart documents.

WebSPEAC Server and Client

WebSPEAC server is a set of task specific CGI programs that generate a multipart document in MIME "multipart/mixed" format described later. In this research, the server was developed on Intel-based Linux machines, but implemented in Perl so that the server works on the other platforms. The basic functions of WebSPEAC server are:

- to get student's speech signal from WebSPEAC clients,
- to control the speech acquisition, the execution of a task specific module, and the display the results on student's Web browser, by generating a customized multipart document.

The WebSPEAC client is a helper application of Web browser. The main functions are very simple, i.e.

- to record students speech signal,
- to send it to WebSPEAC server.

In this research, the client was developed in C on the same platform of the server. However, the implementation on the other platforms seems to be not difficult because of the use of simple functions.

Multipart Documents

The multipart documents in the MIME message format "multipart/mixed" is used by HTTP to encapsulate data returned from a server in response to a request. Typically, an HTTP response consists of only a single piece of data.
However, the “multipart/mixed” MIME message has a standard facility for representing many pieces of data in a single HTTP response (Netscape, 1995). The content of multipart document used in WebSPEAC includes:
1. HTML parts to display some messages on user’s browser,
2. system parameters sent to the WebSPEAC client to record user’s speech, as shown in Figure 4. In this research, the multipart document was implemented by HTTP 1.0 (Berners-Lee et al, 1996).

**Communication between WebSPEAC server and the client**

The most important point in communicating between the server and helper programs is not to require needless operations by user such as clicking OK button. In WebSPEAC system using the multipart document, a natural communication between the server and student is performed as shown in Figure 5.
1. Students execute a WebSPEAC server by clicking the link.
2. The server sends a request to WebSPEAC client in order to record student’s utterance.
3. At the same time, the server forks a data receive server to receive the speech data from the browser.
4. The client records student’s speech and sends a request to the server in order to send the speech data.
5. The file name on the server is determined.
6. The client sends the speech data, and the server receives it and saves in the determined file.
7. The data recording server sends the notification of the end to WebSPEAC server, and finishes.

After the speech acquisition, the WebSPEAC server performs some signal processing for the speech signal received, and generates the results in HTML. Finally, the results are displayed on the student’s browser.

**Discussions**

The current implementation of WebSPEAC has the following problems that should be solved in the future development.
1. Netscape only supports the multipart documents based HTTP 1.0. Hence, HTTP 1.1-based multipart documents that requires to support the multipart documents (Fielding et al, 1997) should be used.
2. The way to automatically generate a customized WebSPEAC server is not prepared. For non-technical designers who wants to create their WebSPEAC server, any generator of WebSPEAC servers is essential.
3. The WebSPEAC client has to be developed for all of platforms used. However, it is not difficult to support them since the function of WebSPEAC is very simple.

**Course Content Examples Using WebSPEAC System**

In this section, we show the following two course content using our WebSPEAC system to demonstrate how it works, 1) an Web-based Sonagram Viewer and 2) an Web-based Continuous Speech Recognizer.

**Sonagram Viewer**

The Sonagram Viewer displays the sound spectrogram, which is a kind of time-frequency pattern of a signal. The horizontal and vertical axes show the time in second and the frequency in Hz, respectively. The intensity is determined by the logarithmic magnitude of the frequency components. The sound spectrogram for speech signals is often referred as “voiceprint” (Rabiner and Juang, 1993).

The user’s views of the Sonagram Viewer using WebSPEAC system are shown in Figures 6-9. Figure 6 shows the top page of the system. To start recording, students click the link to the WebSPEAC server for the Sonagram Viewer. The utterance is sent from the WebSPEAC client to the server through the UNIX socket connection, then the server executes a sound spectrogram generator, which outputs the results in GIF format. Finally, the WebSPEAC server sends back the results in HTML to the user’s browser (see Figure 7). The significant point is that students do not require any clickings through this process except for the click at the start.

In analyzing the speech signal by the sound spectrogram generator, the analysis conditions can be changed as shown in Figure 8. By changing the time interval to be analyzed and the pre-emphasis condition, students can see another sound spectrogram (see Figure 9).

**Continuous Speech Recognizer**

To teach how state-of-art speech recognizers work, it is better to be able to use the speech recognizer itself in the on-line course. Thus, we implemented a state-of-art continuous speech recognizer in our WebSPEAC system, using JULIUS distributed by a project of the Information-technology Promotion Agency, Japan (IPA) (Kawahara et al, 1998). JULIUS is a Japanese continuous speech recognizer based on Hidden Markov Model (HMM) that is familiar in the literature of speech recognition research. The word “continuous” means that the recognizer does not require for speakers to pronounce a sentence to be recognized as the concatenation of the isolated words.

Figures 10 and 11 show the user’s view of JULIUS using the WebSPEAC system. JULIUS system has to be executed as a daemon prior to the use of this WebSPEAC system because it takes a few minutes to load the system parameters. At first, by clicking the recording button, the WebSPEAC server for JULIUS is executed (see Figure 10). After recording by the WebSPEAC client, the server receives the speech and re-sends it to the JULIUS daemon. Then, the JULIUS daemon analyzes the speech, extracts a sequence of speech features (MFCC) used in recognition, and outputs the results as shown in Figure 11.

**Summary**

In this paper, we proposed the WebSPEAC system so as to send user’s speech to WebCT automatically, process it and display the results on the Web browser. Furthermore, we implemented the WebSPEAC system on Linux to show how it works well using two examples of the WebSPEAC system, developed for Digital Speech Signal Processing on-line course. However, the WebSPEAC system is still currently under development. Our future work includes:
1. the implementation of WebSPEAC clients for the other platforms,
2. the development of a WebSPEAC contents editor to facilitate the creation of WebSPEAC server for non-technical designer (see Figure 12),
3. the development of WebSPEAC modules and libraries for several tasks used in the WebSPEAC server, such as filtering, coding and so on (see also Figure 12),
4. the HTTP 1.1 (Fielding et al, 1997) compliance for the multipart document generated by the WebSPEAC server for other browsers like Microsoft Internet Explorer.

(The demonstration page of WebSPEAC system described in this paper will be available at http://www.itakura.nuee.nagoya-u.ac.jp/WebSPEAC/)
References

Kawahara, T., Kobayashi, T., Takeda K., Minematsu, N., Itou, K., Yamamoto, M., Yamada, A., Utsuro, T., (Waveform) Figure 2: The WebSPEAC server consists of several application-specific WebSPEAC servers forked by WebCT. The WebSPEAC server is a kind of CGI program, and the function depends on the required task. The WebSPEAC client is a helper of Web browser, and the task is just to record user's voice and send it to the server.

Figure 1: Conventional method to input student's speech, process it and return the results.
1) Students manually execute a sound recorder, utter a word or phrase, and save the speech in a file.
2) The student manually sends the file to the WebCT server using any file managers provided by a CGI program executed by WebCT.
3) The CGI program detects the file upload, then processes the speech by some methods.
4) The results are displayed on the browser.

Figure 3: Proposed WebSPEAC system to input student's speech, process it and return the results. 1) An WebSPEAC server is executed from WebCT and sends a sound recording request to the student's browser. 2) The request is received by the browser, and the WebSPEAC client is automatically executed as the helper. 3) The student utters a word or phrase after a short beep sound. Then, the recording is automatically finished when he stops speaking. 4) The client sends the speech data to the WebSPEAC server, and finishes.

Figure 4: An example of the multipart document used in WebSPEAC. This document consists of three parts in "multipart/mixed" MIME format bounded by "WebSPEAC918810625" in this example. Part I is used to display a message in order to prompt user to speak. In the second part, a WebSPEAC helper is executed from the browser and the parameters in recording is sent to the helper. Part III and the following parts are used to display the results of some signal processing modules in HTML.

Figure 5: The WebSPEAC client (Helper Application)
Sonagram Viewer using WebSPEAC System

This is a demonstration of WebSPEAC System on World Wide Web. Utter a word or phrase after a short beep sound.

*Notice*

Please adjust the input gain of microphone before using this system. Sorry, we have not implemented any error-trapping routines yet. So, you should not stop this system while recording.

START Recording

Figure 5: Communications between WebSPEAC server and the client. For more information, see the text.

Figure 6: The top page of the Web-based sonagram viewer using the WebSPEAC system. By clicking the "START Recording" prompt, WebSPEAC asks the students to utter a word or phrase.

Figure 7: The resultant sound spectrogram.

Figure 8: The variable parameters are the size of sonagram pattern, the time interval for the analysis, the pre-emphasis (a kind of high-pass filtering to change the slope of spectral envelope) and so on.
Speech Recognition System

Say something to the microphone after you hear a beep.

examples

Japanese(KANJi)
テレビ界に入って五年。
特に練習は何もしていない。
あるのは目的のための音楽だけ。
どんなことをした？

Japanese(Romaji)
Terebisuru Sousa wa Nazono Shisensai.
Ananawa Mokutekutoramensen Ongaku Dake.
Danne koto wo shita?

Figure 10: A snapshot of the Web-based speech recognition contents using WebSPEAC system. After a short beep sound, students utter some sentences, and recognize the sentence by JULIUS.

Figure 11: An example of the recognition result. This is the raw output of JULIUS daemon, which shows the recognition process and results.