Sensors and Materials, Vol. 32, No. 10 (2020) 3209–3220
MYU Tokyo

S & M 2333

Voice Recognition and Marking Using Mel-frequency Cepstral Coefficients

Jia-Shing Sheu* and Ching-Wen Chen

Department of Computer Science, National Taipei University of Education,
No. 134, Sec. 2, He-Ping East Road, Da-an District, Taipei 106, Taiwan

(Received March 14, 2020; accepted June 17, 2020)

Keywords: mel-frequency, Hamming window, speech recognition, fast Fourier transform, microphone array

A real-time voice recognition and marking system was developed in this study to automatically identify different voices of speakers. A microphone array was installed for audio reception. Pre-emphasis, framing and Hamming window, fast Fourier transform, mel-frequency, and mel-frequency cepstral coefficients with processing times of 0.001, 0.305, 0.205, 0.049, and 0.546 s, respectively, were used in the system. The total processing time was less than 1.5 s. Unique eigenvalues were obtained for each sound. The results indicated that the proposed system, which is an example of intelligent recording, can be used to automatically record speech in meetings or during classes.

1. Introduction

With the rapid development of technology, smart applications are becoming increasingly popular, although available, fully automatic voice recorders are still not universal. The objective of this study was to develop a fully automatic voice recorder. A microphone array was used to temporarily store the audio data of speakers. These data were used to calculate mel-frequency cepstral coefficients (MFCCs), which in turn were used to calculate the eigenvalues of sound. Unique eigenvalues were extracted for each sound. Speakers were distinguished by comparing the obtained eigenvalues. After comparison, the eigenvalues of each speaker were stored, and the speakers were then marked. Storing the eigenvalues is convenient because each speaker can be marked quickly. After marking the speakers, we listed the text spoken by the speakers using speech-to-text applications available on the Internet. Therefore, traditional recorders can be eliminated in meetings. A recording of a meeting can be obtained through this fully automatic technology. This technology can also record notes during classes, thus dispelling worries of missing notes or losing them. When this technology matures, it can be extended to numerous applications. Section 2 provides a review of the related literature and methodology. The structure of the system is discussed in Sect. 3. The system design and experiments are detailed in Sect. 4. Finally, conclusions are provided in Sect. 5.

*Corresponding author: e-mail: jiashing@tea.ntue.edu.tw
https://doi.org/10.18494/SAM.2020.2860

ISSN 0914-4935

2. Literature Review and Methodology

In speech recognition, the voice of a speaker is used to distinguish the speaker. The human voice consists of sound waves that are generated by vibration in the vocal cords. Waves that can be perceived by living beings are transmitted through air. The main parameters of sound are frequency, amplitude, wavelength, and tone. Frequency refers to the number of periodic vibrations per second, amplitude refers to the loudness of a sound, and tone is the waveform of a sound. The speaker cannot be distinguished by amplitude alone because the amplitude may be the same for different speakers, whereas the frequency and tone of each person's voice are unique. The voice of an individual can be analyzed and identified through spectrograms and sound range analysis.[1,2] Voice recognition mechanisms can be used to not only distinguish human voices but also instruments that emit sound. In the training process, the playing of different instruments is used as the training material input. After preprocessing, power spectral density, which is a feature to store data into a database through feature extraction, is calculated. These data are stored in a database, and features are applied to their linear superposition during the separation process. Waveforms similar to the original waveform can be extracted by this method. The results can be observed for different waveforms.[3]

The fast Fourier transform (FFT) method is used to quickly calculate the discrete Fourier transform (DFT) or inverse transform of a signal. The FFT mainly processes the signal and converts the signal from the original time domain to the frequency domain and vice versa. The FFT can swiftly decompose the DFT matrix into relatively sparse factors and is widely used in various mathematical and engineering calculations.[4–6] The butterfly diagram obtained by the FFT is used for calculation. The result of a smaller DFT can be combined with that of a larger DFT, and vice versa. It is sometimes necessary to decompose a larger DFT into sub-transforms. Butterfly diagrams are used in the signal analysis. A 32- or 64-bit butterfly diagram can be used to causally link bits to each other through a hash algorithm. This can improve the randomness of some large random arrays. Any bit change may change large arrays.[7,8]

A butterfly representation of an FFT with radix 2 is depicted in Fig. 1. The graphical explanation and formula are as follows:

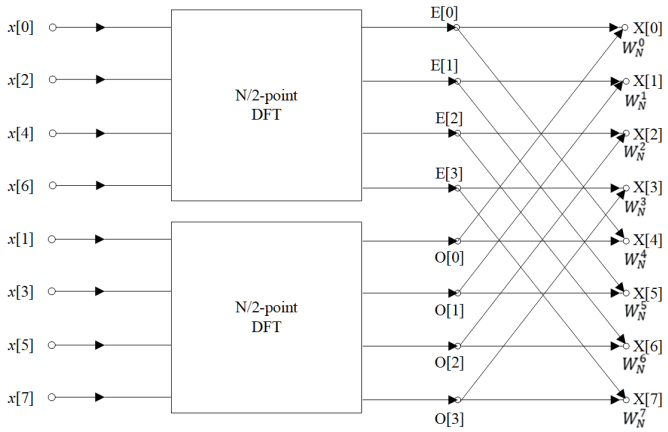

Fig. 1. FFT butterfly.

(1) In the Cooley–Tukey algorithm with a base of 2, the butterfly representation is just a DFT of size 2. Because the size is 2, the input has two corresponding inputs (x_0, x_1) and two outputs (y_0, y_1), as expressed in Eqs. (1) and (2).

$$y_0 = x_0 + x_1 \tag{1}$$

$$y_1 = x_0 - x_1 \tag{2}$$

(2) Relative to the nth root at the beginning of the algorithm, when $n = 2^p$ input bases are 2, the algorithm for the real-time extraction of the FFT $w_n^k = e^{-2\pi ik/n}$ must use Eqs. (3) and (4) in the $O(n\log n)$ butterfly form.

$$y_0 = x_0 + x_1 w_n^k \tag{3}$$

$$y_0 = x_0 - x_1 w_n^k \tag{4}$$

(3) The integer k depends on the conversion of the calculation. The corresponding inverse performs the transformations w and w^{-1}. The inverse butterfly architecture can achieve the same effect, as shown in Eqs. (5) and (6).

$$x_0 = \frac{1}{2}\left(y_0 + y_1\right) \tag{5}$$

$$x_1 = \frac{w_n^{-k}}{2}\left(y_0 - y_1\right) \tag{6}$$

The Hamming window has nonzero values in a certain interval and zero values in other areas. The crux of the Hamming window is $\sin(x)$, which varies from 0 to π. To highlight the frequency component, any signal $f(x)$ can be multiplied by the Hamming window to obtain a better frequency response. Windowing is necessary in signal processing. Our computer can only process signals of a certain length. We need to reduce the original signal $X(t)$ by the sampling time and then change it to $XT(t)$, after which, we process it further. This process is called windowing.[9]

MFCCs are widely used in speech recognition technology, which was proposed in the 1980s. In sound processing, the linear conversion of the logarithmic energy spectrum of the nonlinear mel scale of the sound frequency is performed. The MFCCs are based on this conversion and generated from the cepstrum of each audio segment. The difference between MFCCs and the normal cepstrum is that the frequency band of the linear interval of the MFCCs is lower than that of the normal cepstrum. The frequency spectrum is closer to the human hearing range than the normal cepstrum because the human voice is nonlinear. In this nonlinear state, audio is better represented in more fields. Prior to the technology of MFCCs, speech recognition methods such as linear prediction coefficient (LPC) and linear prediction cepstral coefficient

(LPCC) methods were used; these methods were only effective for identifying the speaker in the case of linear signals. Therefore, LPCs and LPCCs have lower performance characteristics than MFCCs.[10–12]

3. System Architecture

In this study, the eigenvalues of the voice were used to identify and mark the speaker system. First, a microphone array was used to receive sound signals. Then, the eigenvalues of the sound were extracted and stored. After comparing the eigenvalues, the speakers were marked. Figure 2 shows the system architecture obtained using IDEF0.[13] To accomplish speaker identification and marking, the extraction of eigenvalues is necessary. After inputting 5 s of temporary data, pre-emphasis was performed and then frames were added to the window. The signal was converted into the frequency domain through FFT, mel-frequency was calculated, and then the MFCCs were obtained. These MFCCs have the same value as the eigenvalues. We use IDEF0 (ICAM Definition Language)[13] to construct our system architecture, as depicted in Fig. 3.

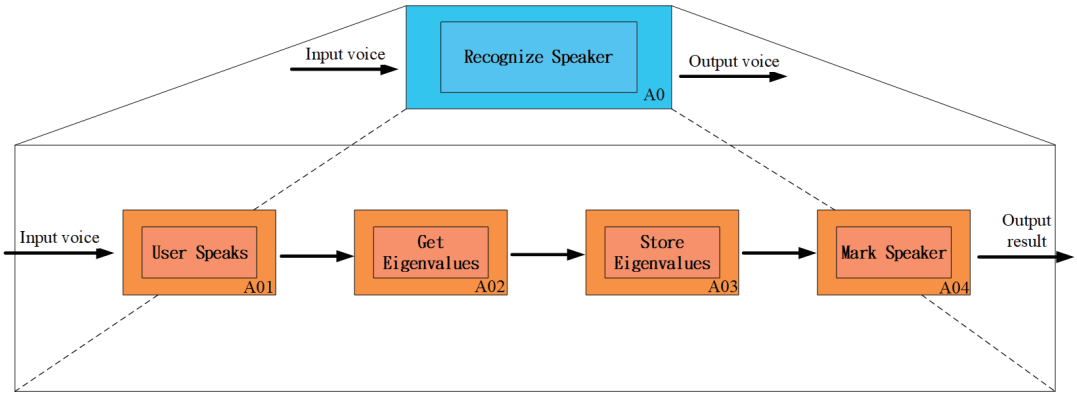

Fig. 2. (Color online) Architecture of the proposed system.

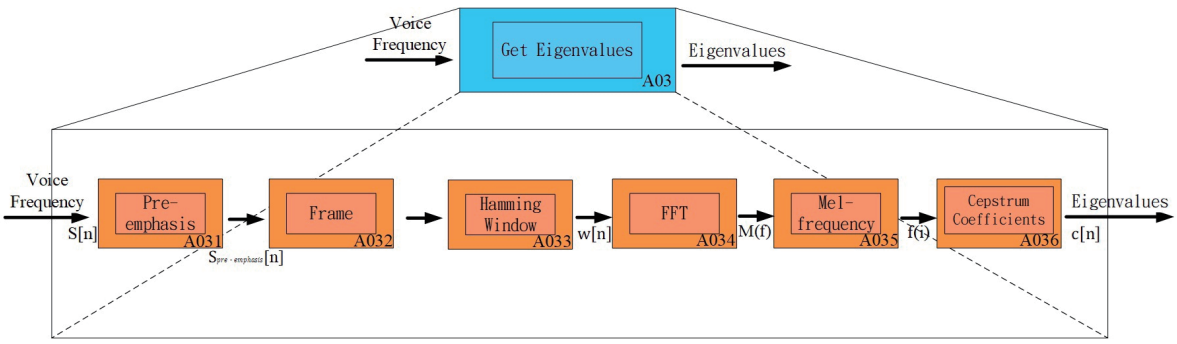

Fig. 3. (Color online) Architecture for obtaining eigenvalues.

The steps for obtaining eigenvalues were as follows:

(1) Pre-emphasis:

Pre-emphasis was performed during the inputting of 5 s of temporarily stored data first. The purpose of pre-emphasis was to strengthen the high-frequency part of the voice signal. In the voice signal, the energy at lower frequencies is higher than that at higher frequencies. Therefore, pre-emphasis was used to filter the low-frequency part of the data to make the high-frequency part more prominent. The pre-emphasis function is expressed as

$$S_{pre\text{-}emphasis}[n] = S[n] - \alpha \times S[n-1], \quad 0.9 \le \alpha \le 1. \tag{7}$$

(2) Frame processing:

After pre-emphasis, frame processing was performed to aggregate N sampling points into an observation unit. The coverage time set here was 0.025 s per frame and the sampling rate was 16000 Hz. Therefore, there were 400 sample points per frame.

(3) Hamming window:

An FFT was performed immediately after the frames were divided. However, the signals in each frame were processed periodically during the conversion. Therefore, the two ends of the frame were changed, and the converted signal spectrum was different from the original signal spectrum. Windows were added to the two ends to ensure that they do not change. The window-type function that we selected here is the Hamming window, which is expressed as

$$w[n] = 0.54 - 0.46\cos\left(\frac{2\pi n}{M}\right). \tag{8}$$

(4) FFT:

After windowing, we can safely perform FFT conversion. To perform FFT conversion on each frame, the length of the input data must be 2^k. In the above frame, we take 400 sample points per frame in our system, and we directly pad to zero to the nearest sample. Then, a 512-point conversion was performed on each frame after windowing to obtain the spectrum of each frame. The spectrum of the speech signal was squared. Then, we obtained the power spectrum of the speech signal to complete the FFT conversion.

(5) Mel-frequency:

First, the actual frequency was converted into the mel-frequency. From Eq. (9), the minimum actual frequency was 0 Hz and the maximum frequency was 8000 Hz. After mel-frequency conversion, 40 filters were used. The mel-frequency distribution was calculated using these 40 filters, and the mel-frequency was converted to the actual frequency using Eq. (10). After calculating the actual frequency, the f array was calculated from the array of actual frequencies using Eq. (11). Finally, the output of the filter was calculated using Eq. (12).

$$M(f) = 1125 \times \ln\left(1 + \frac{f}{700}\right) \tag{9}$$

$$M^{-1}(m) = 700\left(\exp\left(\frac{m}{1125}\right) - 1\right) \tag{10}$$

$$f(i) = floor\left((N+1) \times \frac{h(i)}{sample\,rate}\right) \tag{11}$$

$$H_m(k) = \begin{cases} 0, & k < f(m-1) \\ \dfrac{k - f(m-1)}{f(m) - f(m-1)}, & f(m-1) \le k \le f(m) \\ \dfrac{f(m+1) - k}{f(m+1) - f(m)}, & f(m) \le k \le f(m+1) \\ 0, & k > f(m+1) \end{cases} \tag{12}$$

(6) MFCC:

The logarithm of the mel scale was obtained using the output of the triangular filter. Thus, a result similar to the human ear of conversion was obtained, and DCT conversion was performed on the log energy mel spectrum. Here, we took the first 13-dimensional output as 13 feature parameters. After finding the first 13 dimensions of the output, we added the difference cepstrum parameter to obtain the change in the cepstrum parameter with respect to time, and then the MFCC was obtained as

$$c[n] = \sum_{i=0}^{L-1} \ln\left(S[i]\right) \cos\left(\frac{\pi n}{2L}\left(2i+1\right)\right). \tag{13}$$

4. Experimental Results

The experimental environment of the program comprised a Raspberry Pi board and a four-microphone (four-mic) array[14] for digital signal processors, as listed in Table 1.

Table 1
Hardware specifications.

Component	Specification
Operating system	Raspbian
Central processing unit	ARM Cortex-A72
Random access memory	4GB(LPDDR4)
Microphone array	Respeaker 4-Mic Array

The following is the algorithm used in this study:

Algorithm

```
public class MFCC{
    public MFCC () {
        declare a variable audio;
        declare a variable eigenvalue;
        declare marks of speakers;

        while audio input {

        do
                Pre-emphasis; // highlighting high frequencies;
                Frame;
                Hamming window;
                FFT; // transfer time domain to frequency domain
                    // Mel-frequency;
                Mel-frequency cepstral coefficients; // calculate the
                                                    // eigenvalues;
        then
                return eigenvalues;
    }

        for(::){
                identify the eigenvalues to match who or new one
        }

    }
}
```

The microphone array was combined with a Raspberry Pi module, as depicted in Fig. 4. The hardware block diagram is shown in Fig. 5. The microphone array recorded the sound in 8-bit mono channels. The sampling frequency was uniformly set at 8000 Hz. The length of the recordings was set to 5 s and basic data were unified subsequently. The basic data of each recording file were the same, but the value of conversion varied depending on the frequency of the sound in each recording file. These analyzed sound values were stored and subsequently

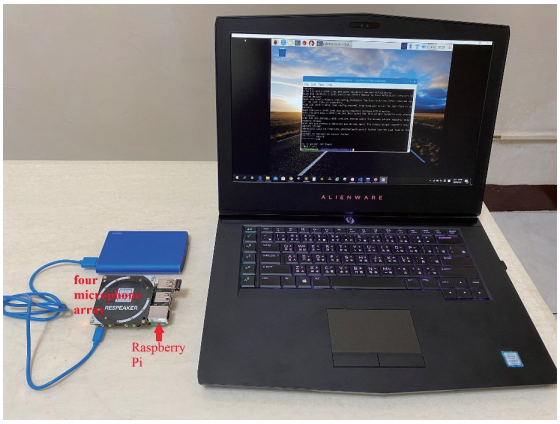

Fig. 4. (Color online) Experimental environment for Raspberry Pi and microphone array.

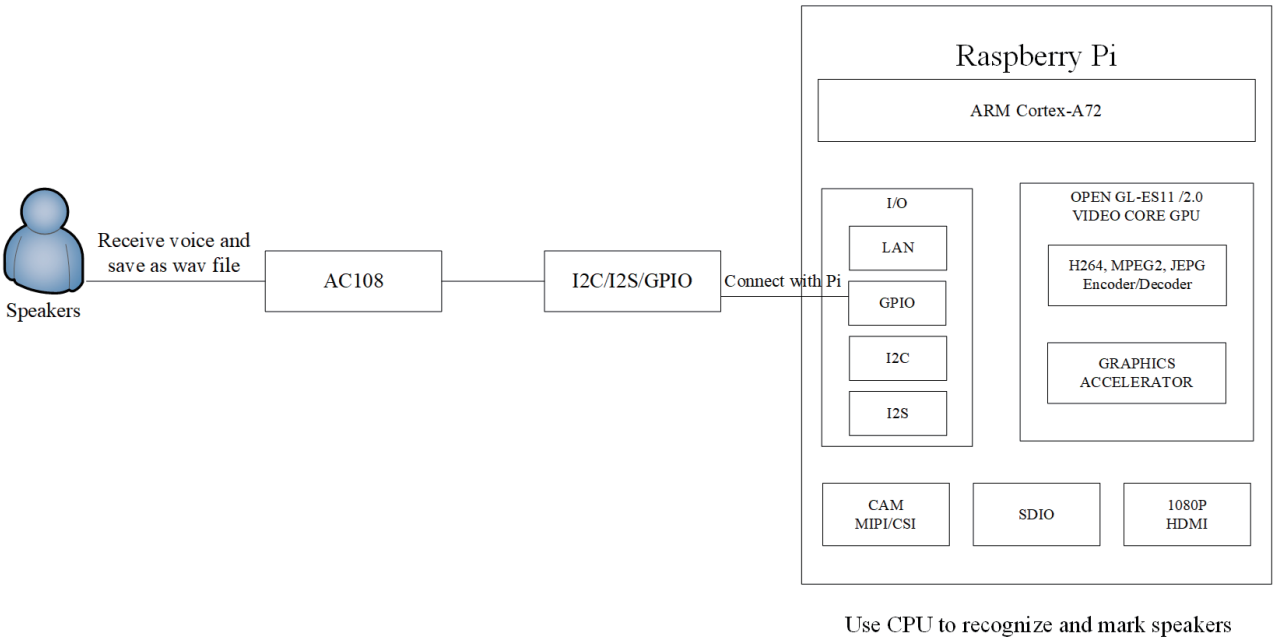

Fig. 5. (Color online) Hardware block diagram of experiment.

sent to the Raspberry Pi module to calculate the eigenvalues of each sound. These eigenvalues were compared and the speaker was marked.

C^{++} was used to write the program for obtaining eigenvalues. In this program, first, the Raspberry Pi is booted. The four-mic array receives and records the sound in a WAV audio file. These parsed data are processed with eigenvalues and then compared to mark the user. Eigenvalues were used to compare and mark the speaker because the characteristics of each person's voice are unique. The eigenvalues of each person's speech are calculated and stored, which are then used when the speaker speaks again. Thus, the speaker can be accurately marked.

In this experiment, the sound data of six people were stored. They were subsequently extracted and marked using eigenvalues. We used each loop to run 100 times to determine which speaker was speaking. In the results of this experiment, the speaker can be marked on the basis of the data in the recording file.

To evaluate the effectiveness of the system, tests were conducted at different times. In the second experiment, voice samples of the six people were recorded again. The eigenvalues obtained from the six new recordings were compared with those extracted from the previous six voice recordings. The voice sample was looped to run 100 times. The eigenvalues of the six new segments of the speech were compared with those of the first six segments of the speech. The eigenvalues extracted from the six new speech segments were compared with those extracted from the original six segments of the speech. The speaker was not recognized every time. The recognition accuracies for speakers 1–6 were 78, 76, 76, 75, 79, and 77%, respectively. The results are presented in Table 2.

Table 2
Second time experimental results.

Speakers	Marks	Successes	Success rate (%)
1	100	78	78
2	100	76	76
3	100	76	76
4	100	75	75
5	100	79	79
6	100	77	77

Table 3
Third time experimental results.

Speakers	Marks	Successes	Success rate (%)
1	100	76	76
2	100	77	77
3	100	68	68
4	100	77	77
5	100	75	75
6	100	78	78

The recognition accuracies of the first and second experiments were marginally different. Therefore, we recorded the latest six segments of the speech and compared the eigenvalues extracted for these speech segments with the original eigenvalues. We used the loop to run these new six segments of the speech 100 times and compared them with the original six segments of the speech. The recognition accuracies for speakers 1–6 were 76, 77, 68, 77, 75, and 78%, respectively, as shown in Table 3.

It was found that the eigenvalues of the second and third experiments were different from those of the first experiment. This difference could be attributed to the following reasons:

(1) Environmental factors:

In the recording, if noise from the environment is recorded by the microphones, then the calculated eigenvalues of this voice will be different from the original stored eigenvalues. However, noise may not always affect the eigenvalues. The probability that noise can cause problems in identification can be calculated.

(2) Current state of the speaker:

When recording, if the speaker is wearing a mask or is sick, then the voice of the speaker changes, which causes a difference in eigenvalue. In the third experiment, speaker 3 was the only person who had a cold. Because speaker 3 had a cold, the recognition accuracy for the third experiment was less than that for the second experiment. These two experiments were conducted on different days. The current state of the speaker considerably affects the accuracy (by approximately 5%).

In this experiment, we also investigated the time required to mark the speakers. The sound data of six people were run 100 times on a loop, and the time between each mark was calculated. After speaker 1 was marked, the time between marking speakers 1 and 2 was determined. This time is an indicator of the effectiveness of the proposed system. In the experimental results, the first mark was speaker 2, and then speakers were marked in the order of speakers 5, 3, 1, 4, and 6 to measure the response time between two speakers. The times from speakers 2 to 5, speakers 5 to 3, speakers 3 to 1, speakers 1 to 4, and speakers 4 to 6 are depicted in Figs. 6–10, respectively. The results showed that the response time of each speaker differed in each figure. Because the frequency of each person's voice is different, the sizes of the generated eigenvalues are also different. Thus, when we compare and mark speakers, such a time gap is expected.

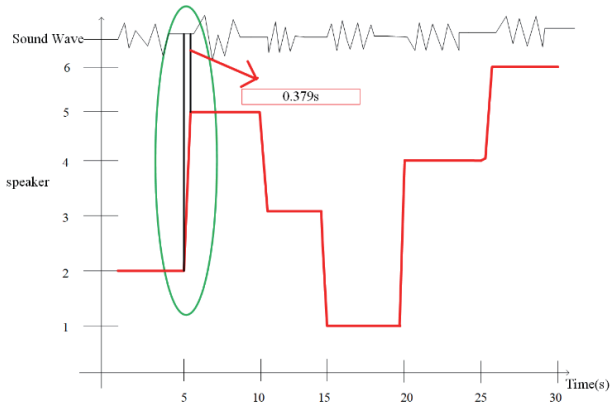

Fig. 6. (Color online) Time from speakers 2 to 5.

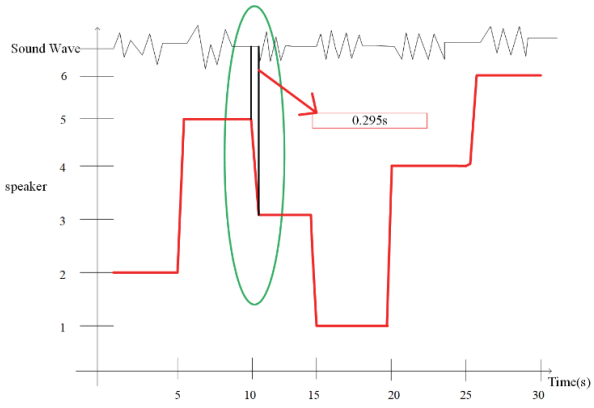

Fig. 7. (Color online) Time from speakers 5 to 3.

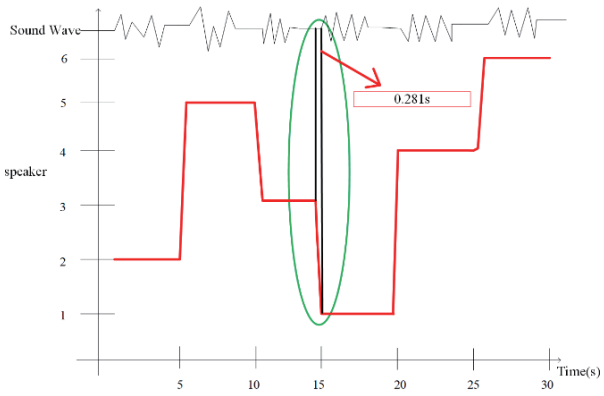

Fig. 8. (Color online) Time from speakers 3 to 1.

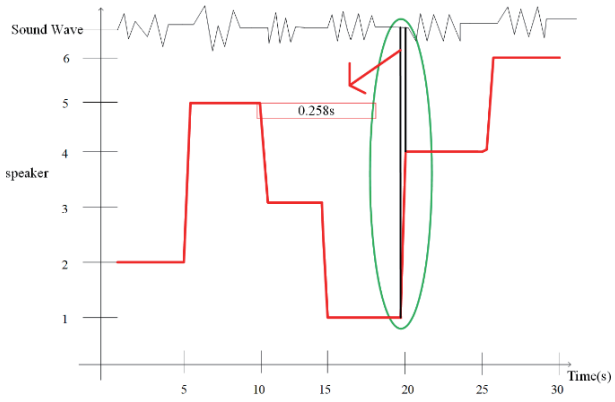

Fig. 9. (Color online) Time from speakers 1 to 4.

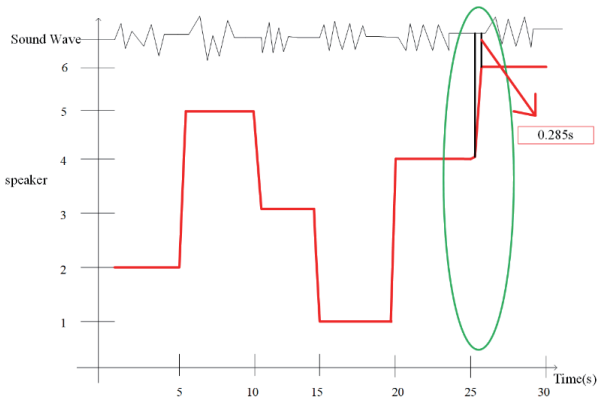

Fig. 10. (Color online) Time from speakers 4 to 6.

5. Conclusions

Eigenvalues were used to distinguish and mark speakers. The experimental results proved that the algorithm of voice recognition can be distinguished by different human voice characteristics and this algorithm of voice recognition is effective, but there may be noise received when speaking is detected. The experimental results shown in Figs. 6–10 demonstrated the calculated time required for this algorithm to mark a speaker. In the future, the results of this study can be used to develop an automatic meeting recorder or a class note recorder. The operation time of the system should be reduced and new functions should be incorporated to develop a fully automatic recorder.

References

1 C. H. Lee and S. M. Siniscalchi: Proc. IEEE. **101** (2013) 1089. https://doi.org/10.1109/JPROC.2013.2238591
2 Y. Takashima, R. Takashima, T. Takiguchi, and Y. Ariki: IEEE Access **7** (2019) 164320. https://doi.org/10.1109/ACCESS.2019.2951856
3 J. S. Sheu, K. W. Chiu, and T. L. Huang: Information **16** (2013) 7015.
4 C. Chi, Z. Li, Q. Li, and Y. Ariki: IEEE J. Oceanic Eng. **41** (2016) 249. https://doi.org/10.1109/JOE.2015.2429251
5 P. K. Meher, B. K. Mohanty, S. K. Patel, S. Ganguly, and T. Srikanthan: IEEE Trans. Circuits Syst. I Regul. Pap. **62** (2015) 2836. https://doi.org/10.1109/TCSI.2015.2495724
6 M. M. Jia, S. Sun, Y. Li, Z. G. Qian, and W. C. Chew: IEEE Trans. Antennas Propag. **64** (2016) 4559. https://doi.org/10.1109/TAP.2016.2593930
7 Z. Qian and M. Margala: IEEE Trans. Very Large Scale Integr. VLSI Syst. **24** (2016) 3008. https://doi.org/10.1109/TVLSI.2016.2544838
8 Y. J. Yang, H. Dang, L. Xin, X. Ke, and W. L. Yuan: Pro. 2013 IEEE 13th Int. Conf. Signal Processing. (IEEE, 2016) 496–501. https://doi.org/10.1109/ICSP.2016.7877884
9 R. Webster: IEEE Trans. Acoust. Speech Signal Process. **26** (1978) 269. https://doi.org/10.1109/TASSP.1978.1163093
10 Z. Wu and Z. Cao: Tsinghua Sci. Technol. **10** (2005) 158. https://doi.org/10.1016/S1007-0214(05)70048-1
11 S. Nakagawa, L. Wang, and S. Ohtsuka: IEEE Trans. Audio Speech Language Process. **20** (2012) 1085. https://doi.org/10.1109/TASL.2011.2172422
12 M. Sahidullah and G. Saha: IEEE Signal Process Lett. **20** (2013) 149. https://doi.org/10.1109/LSP.2012.2235067
13 J. S. Sheu and C. Y. Han: Adv. Technol. Innovation **5** (2020) 10. http://ojs.imeti.org/index.php/AITI/article/view/4284
14 ReSpeaker: https://wiki.seeedstudio.com/ReSpeaker_4_Mic_Array_for_Raspberry_Pi/ (accessed January 2020).

About the Authors

Jia-Shing Sheu received his MS and PhD degrees from the Department of Electrical Engineering at National Cheng Kung University, Tainan, Taiwan, in 1995 and 2002, respectively. He is currently in the Department of Computer Science, National Taipei University of Education, Taipei, Taiwan. His research interests include pattern recognition and image processing, especially focusing on embedded systems. (jiashing@tea.ntue.edu.tw)

Ching-Wen Chen received his BS degree from the Department of Computer Science and Information Engineering, Chinese Culture University, Taipei, Taiwan, in 2019. He is currently a graduate student at the Department of Computer Science, National Taipei University of Education, Taiwan. His main interests are in the field of voice and embedded systems.
(mikechen31@kimo.com)