Journal of Zhejiang University-SCIENCE C (Computers & Electronics)
ISSN 1869-1951 (Print); ISSN 1869-196X (Online)
www.zju.edu.cn/jzus; www.springerlink.com
E-mail: jzus@zju.edu.cn

An algorithm that minimizes audio fingerprints using the difference of Gaussians

MyoungBeom CHUNG, IlJu KO
(*Department of Media, Soongsil University, Seoul 156-743, Korea*)
E-mail: {nzin, andy}@ssu.ac.kr

Received Nov. 16, 2010; Revision accepted Mar. 18, 2011; Crosschecked Sept. 1, 2011

Abstract: Recently, many audio search sites headed by Google have used audio fingerprinting technology to search for the same audio and protect the music copyright using one part of the audio data. However, if there are fingerprints per audio file, then the amount of query data for the audio search increases. In this paper, we propose a novel method that can reduce the number of fingerprints while providing a level of performance similar to that of existing methods. The proposed method uses the difference of Gaussians which is often used in feature extraction during image signal processing. In the experiment, we use the proposed method and dynamic time warping and undertake an experimental search for the same audio with a success rate of 90%. The proposed method, therefore, can be used for an effective audio search.

Key words: Audio retrieval, Audio fingerprint, Audio signal processing, Difference of Gaussians (DoG)
doi:10.1631/jzus.C1000396 **Document code:** A **CLC number:** TN912.34

1 Introduction

According to continuing development in computer hardware and software technology, sharing of digital information data over the Internet is increasing exponentially. Similarly, as the devices for listening to music have changed from cassette tapes or CD players to MP3 players, portable multimedia players (PMPs), or mobile phones, sharing of audio data has also increased. According to the International Federation of the Phonographic Industry (IFPI), for example, digital music profit is 3700 million dollars in all the world, and a single track is downloaded up to 1400 million times (Kennedy, 2009).

Thus, the companies that supply and share music need methods for efficiently storing and searching huge amounts of music data. Several years ago, in audio searches, many researchers used text-based search methods (Ponte and Croft, 1998; Stephen, 1999; Pickens, 2002). However, if we did not query the correct keyword with which to search, we could not find the audio file required, because these searches found audio files that needed specific keywords for results. To solve this problem, recently many researchers have proposed content-based methods that extract fingerprints from audio files and searches (Kimura *et al.*, 2001; Mihak and Venkatesan, 2001; Rein and Reisslein, 2006; Kim *et al.*, 2008). As fingerprinting is a technology that can be used to recognize audio or video (Chung and Ko, 2010), we can search for the same audio using only a small part of the audio data. For example, Google uses a fingerprinting technology called Waveprint, and Enswer.me uses the technology that combines mel-frequency cepstral coefficients (MFCCs) and linear prediction coding (LPC).

When a search is conducted using fingerprints, the number of fingerprints is related to the amount of audio data that can be handled simultaneously. As Google's Waveprint uses 64 KB of memory per song, this search software can process about 30 700 songs in a 2 GB RAM system simultaneously (Baluja and Covell, 2007; 2008). Enswer.me's fingerprint technology uses only 6 KB of memory per song, and thus

can process about 300 000 songs in the same environment (Logan, 2000; Enswers Co., 2008). In other words, if we reduce the number of fingerprints of the audio data, the search system can process more songs at the same time than the existing systems.

In this paper, we propose a novel method that can reduce the number of fingerprints while maintaining a level of performance similar to that of the existing methods. The proposed method uses the difference of Gaussians (DoG), which is often used in feature extraction during image signal processing (Lowe, 2004). The DoG, which is an important technology of scale invariant feature transform (SIFT), can perform image matching and image fingerprinting. We apply the DoG to audio data. We can extract a region whose strength and weakness of the waveform audio change significantly. Then, we can create fingerprints using fast Fourier transform (FFT) from the extracted region. The proposed method creates 1–3 fingerprints per second and obtains about 300 fingerprints from 4-min audio data. The required memory of the fingerprints is about 3 KB per song and is able to process about 600 000 songs in the 2 GB RAM system simultaneously. The required memory in the proposed method is reduced by 95% compared with Google's fingerprint method and by 50% compared with Enswer.me's fingerprint method.

To evaluate whether the proposed method is as accurate as the existing methods, we selected 200 songs and conducted an audio search experiment using the proposed method and dynamic time warping (DTW). As a result, we undertook an experimental search for the same audio with a success rate of 90%. This success rate is similar to those of Google and Enswer.me's fingerprint technology.

2 Previous work

An audio fingerprint can be defined as the metadata that corresponds to the audio regardless of the audio format, frequency, changes in the audio environment, and quality of audio recording. In other words, an audio fingerprint is a compact, content-based signature that summarizes a group of audio data. The audio fingerprint is often used for unlabeled audio in content-based retrieval of the same and similar audio files. Cano *et al*. (2005) defined an audio identification system as having the properties of an audio fingerprint, such as accuracy, reliability, robustness, granularity, security, versatility, scalability, complexity, and fragility. They then provided a description of an audio fingerprint as an entity that should be a perceptual digest of the recording and does not change with distortion, is compact, and is easily computable.

In audio retrieval, in the early days of fingerprints, Wold *et al*. (1996) used a fingerprint for their system called Muscle Fish. They made the fingerprint using audio features such as sound loudness, brightness, pitch, and timbre. Then, they used Mahalanobis distance and nearest neighbor interpolation to search for audio using fingerprints. However, because the applicable range of this system was only for sound effects such as musical instrument sounds, animal voices, and environmental sounds, the system could not be applied to audio file searches. Wang and Smith (2002) proposed an audio search method that uses a set of landmark time-points and associated fingerprints. They expressed an audio file as the spectrum energy of a 32-frequency band and extracted a set of landmark time-points, and then computed the associated fingerprints from the landmark time-points and used the fingerprints in the audio search. However, because the number of fingerprints used in this method varied depending on the audio genre, the method could not guarantee the speed of the search. Moreover, the method is weak in codec or the bit-rate change of audio.

Rein and Reisslein (2006) proposed a method for using audio fingerprints that enables the identification of an unidentified classical music composition through the use of a wavelet dispersion vector and of neural nets. Their methodology obtains the wavelet dispersion vector through a combination of 39 different wavelets. The method is then able to identify the unknown composition via neural-net assessment of the similarity between unknown query vectors and known vectors. The proposed methodology is good regardless of audio frequency, environment, or recording quality.

Recently, Google's audio search system has used a fingerprint method called Waveprint (Baluja and Covell, 2007; 2008). Waveprint's key algorithm is based on wavelets, and this method is robust to codec or a bit-rate change of audio. Because

Waveprint's wave pattern is not changed by expansion or reduction, this method can extract an equal pattern even if the audio becomes slow or fast. Waveprint extracts about 4000 fingerprints from one 4-min audio file and uses an average of 64 KB of memory to store the fingerprints. Therefore, Google's fingerprint system can process about 30 700 songs in a 2 GB RAM system simultaneously.

Enswer.me, a movie and audio search speciality company in South Korea, uses a fingerprint method that is mixed MFCCs and LPC (Logan, 2000; Enswers Co., 2008). This system extracts MFCCs and LPC after dividing the audio data into 100 ms. Then the method uses frequency analysis to analyze the distribution pattern of the audio and extracts the fingerprint. The key feature of this method is that it uses a small number of fingerprints while maintaining search accuracy. As Enswer.me's fingerprinting needs 40 bytes of memory per second, the method uses about 6 KB of memory to store fingerprints from one 4-min audio file. As the number of fingerprints in this method is reduced by 90% compared with Google's Waveprint, this method can process about 300 000 songs in a 2 GB RAM system simultaneously. This method has a weakness, however. It is not robust to codec or a bit rate change because of the use of frequency analysis.

3 Audio fingerprint using the difference of Gaussians

The existing fingerprint methods extract features from the frequency transformation of audio. The frequency transformation data can be used to analyze the frequency pattern of the audio file. However, because the transformation loses information about audio time, it is necessary to transform all of the audio data into the frequency domain to search for the same audio. If all of the audio data is transformed into frequency data, then noninfluential parts such as silence, humming, and small audio sounds are also transformed, and this increases the number of fingerprints.

In this work, we propose a novel fingerprint method that uses the difference of Gaussians (DoG) to make up for this weakness. The DoG is an algorithm that is used mainly in the SIFT algorithm in the image processing field. The DoG of the SIFT algorithm extracts the maximum point and the minimum point from the scale space of the image and finds the key points of the image. The other methods that find the key points of an image are Gradient, Hessain, and the Harris corner function (Mikolajczyk and Schmid, 2004). The DoG algorithm is more stable than these methods. Therefore, we extract the features using the DoG from the audio waveform instead of the frequency transformation of the audio. The extraction process of the proposed fingerprint is shown in Fig. 1.

At the preprocessing step in Fig. 1, the proposed method converts stereo to mono data, and the converted mono data that is waveform is passed through the Gaussian filter to remove noise. The DoG data calculation step applies the Gaussian function to the waveform that was passed through the preprocessing step and obtains the DoG data. The feature point extraction step traces the maximum point from each frame of the DoG data and extracts the feature points. The region value extraction step applies FFT to the extracted feature point and extracts the region value of the feature point. Finally, the method defines the fingerprint that consists of six dimensions using the feature point and the five FFT dimensions of the region value.

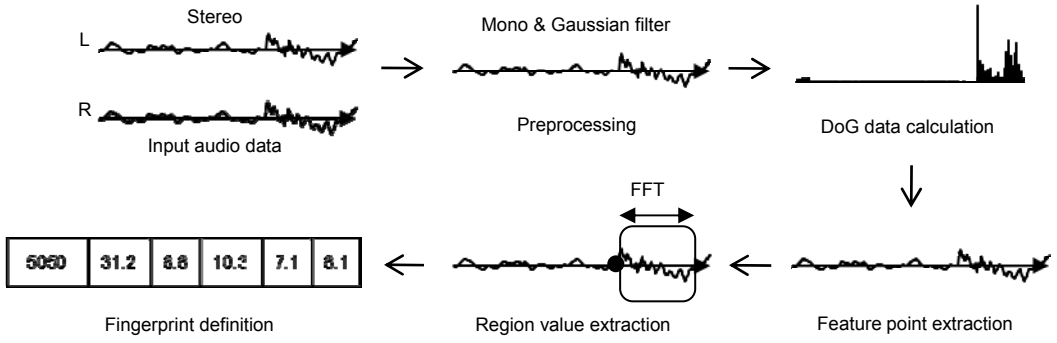

Fig. 1 The fingerprint extraction process using the difference of Gaussians (DoG) algorithm

3.1 DoG data calculation

We can obtain the DoG data using the DoG from the waveform of the audio data and estimate the change in the waveform from the DoG data. Fig. 2 shows the calculation process of the DoG data.

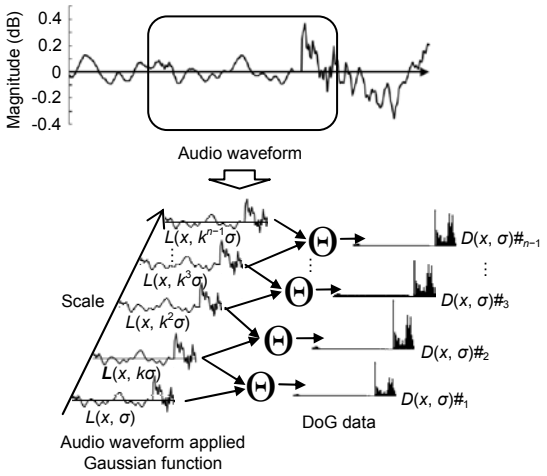

Fig. 2 The difference of Gaussians (DoG) data calculation process using the DoG algorithm

In Fig. 2, we apply the Gaussian function from 1 to n to the audio waveform and calculate the DoG data from the difference of each Gaussian waveform to which the Gaussian function is applied. Generally speaking, in the image processing field, because the DoG is an algorithm that extracts the features of an image, the DoG uses the two-dimensional Gaussian kernel function. However, audio data is only one-dimensional. Therefore, we use the modified one-dimensional Gaussian kernel function for calculation:

$$G(x,\sigma)=\frac{1}{\sqrt{2\pi\sigma^2}}\exp\left(-\frac{x^2}{2\sigma^2}\right), \tag{1}$$

where σ is the scale factor, and $G(x, \sigma)$ is the Gaussian filter. We obtain the Gaussian data from the applied Gaussian function using the Gaussian filter and audio waveform $A(x)$:

$$L(x,\sigma)=G(x,\sigma)\times A(x). \tag{2}$$

Then, we calculate the DoG data using the DoG data $L(x, \sigma)$ and $L(x, k\sigma)$:

$$\begin{aligned} D(x,\sigma)&=L(x,k\sigma)-L(x,\sigma)\\ &=(G(x,k\sigma)-G(x,\sigma))\times A(x). \end{aligned} \tag{3}$$

$D(x, \sigma)$ is the DoG data and has a high value at the part where the waveform changes rapidly. In contrast, the data has a low value at the part where the waveform changes slowly. The result is similar to DoG data that expresses the edge information of an image in the image processing field. Indeed, the DoG data of an audio file builds similar values at the same part although the quality of the audio data changes. Therefore, we calculate the DoG data from the audio for feature extraction. $D(x, \sigma)\#_1$ is calculated from $L(x, \sigma)$ and $L(x, k\sigma)$, $D(x, \sigma)\#_2$ is calculated from $L(x, k\sigma)$ and $L(x, k^2\sigma)$, and $D(x, \sigma)\#_{n-1}$ is calculated from $L(x, k^{n-2}\sigma)$ and $L(x, k^{n-1}\sigma)$.

3.2 Feature point extraction

We can calculate the DoG data, but one point of DoG data is the value of 0.023 ms of audio data that is sampled at 44.1 kHz. While it is necessary to have moderate amounts of audio data in order to recognize audio, one point of the DoG data is a very small value that cannot be recognized. Therefore, we must define the feature using an amount of DoG data that can be recognized. Thus, we compose DoG frames (F_1, F_2, …, F_j) by moving the search range by i ms. The DoG frames consist of the current search range, the prior search range, and the subsequent search range of DoG data ($D(x, \sigma)\#_1$, $D(x, \sigma)\#_2$, …, $D(x, \sigma)\#_{n-1}$) by i ms and are expressed in $3\times(n-1)$ form (Fig. 3).

In Fig. 3, ($L_1, L_2, \ldots, L_{n-1}$), ($C_1, C_2, \ldots, C_{n-1}$), and ($R_1, R_2, \ldots, R_{n-1}$) are the sums of each DoG dataset at i ms. Then, we compare the center value of the current search range with the surrounding values of the current range to judge whether the current range is the feature point. If the center value is larger than the surrounding values, then it is called the maximum value, and the current range is designated the feature point. According to the number of DoG datasets, the method for selecting the center value and the surrounding values is as shown in Fig. 4.

Fig. 4a shows two DoG datasets. In this case, the center value is selected from the larger value of C_1 and C_2, and the others except for the selected center are the surrounding values. Fig. 4b shows three DoG datasets, and we select C_2 as the center value. Fig. 4c shows $n-1$ DoG datasets. In this case, we select the largest value among C_2, C_3, …, C_{n-2} as the center value, and the nearest surrounding values from the selected center are the surrounding values.

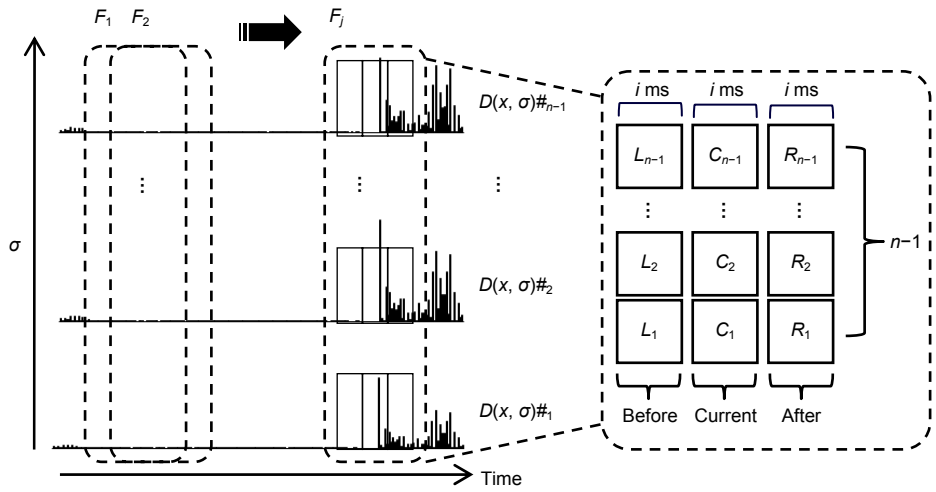

Fig. 3 The composition process of difference of Gaussians (DoG) frames using the DoG data

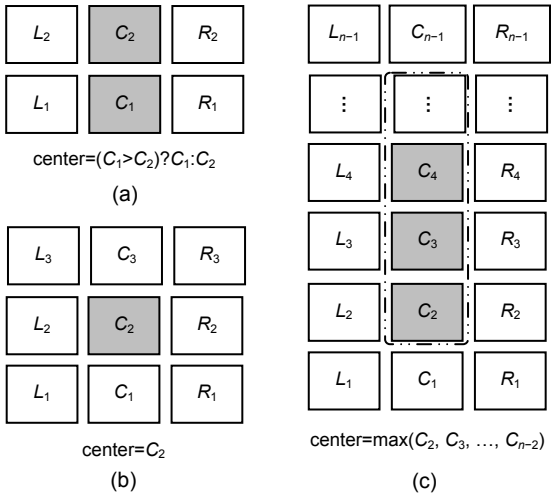

Fig. 4 The center selection method by the difference of Gaussians (DoG) dataset number

(a) Two DoG datasets; (b) Three DoG datasets; (c) $n-1$ DoG datasets

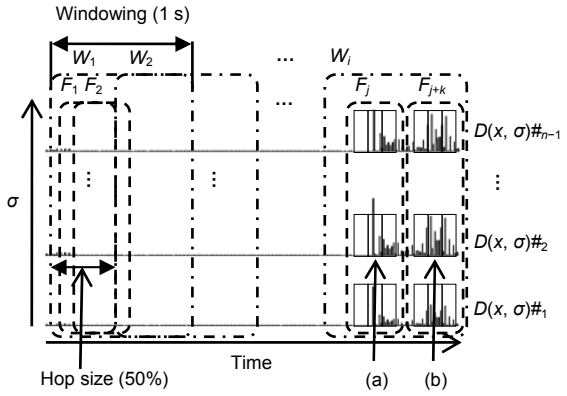

Fig. 5 The selection of the largest maximum value using a window

(a) and (b) are the maximum values of DoG frames F_j and F_{j+k}, respectively

When the feature point is defined by the maximum value, the maximum values of music whose audio waveform often changes rapidly, such as dance and rock, appear frequently. If we do not change or use those, then the number of fingerprints may become large according to the type of audio. Thus, we use a window and select the largest maximum value of the maximum values as the feature point in the window (Fig. 5).

In Fig. 5, the time length of the window is 1 s, and it moves by a 50% overlap. For example, in Fig. 5, (a) is the maximum value of DoG frame F_j, and (b) is the maximum value of DoG frame F_{j+k}. If we do not use a window, then both (a) and (b) are the selected feature points. If we use a window, because (a) and (b) belong to window W_i, then we compare (a) and (b). The largest maximum value between (a) and (b) is the selected feature point in window W_i. With this window processing, we can reduce the number of feature points in audio data such as dance and rock. As a result, the largest maximum value through window processing is the final feature point and is one of the fingerprint values that we propose.

3.3 Region value extraction of the feature point

We extract the region values of the feature point in the final step of fingerprint extraction. If we use only the feature point that is referred to in Section 3.2 for an audio search, then various audio files that have

similar waveforms can cause repetition errors. Thus, we transform the audio waveform into the frequency at the feature point and extract the region value around the feature point. Using the region value with the feature point for the audio search will reduce the number of repetition errors.

Region value extraction must be selected that expresses the same value even if the audio quality changes at the feature point. Therefore, we draw the frequency ingredients by applying FFT to the audio waveforms that belong to the same piece of music, but whose audio quality is different (Fig. 6).

In Fig. 6, the bit rates of the audio waveform used are 128, 192, 256, and 320 kb/s. Although the low and middle frequencies show similar distribution, the high frequency is slightly different. Indeed, the important parts of the audio data are the low and middle frequencies (1–201: 0–2100 Hz), as shown in Fig. 6 (Independent Recording Network, 2006). Therefore, we use low and middle frequencies that do not influence the quality change and extract the region value of the feature point (Fig. 7).

We calculate the frequency ingredient of FFT using the audio data of a 2048-point range from the feature point. Then, we divide 200 index bins belonging to low and middle frequencies among the total frequency into five parts and calculate the magnitude average of each part. The magnitude average of each part is the region value of the feature point.

4 Audio search using the proposed fingerprint method

In this work, we use the dynamic time warping (DTW) algorithm with the search method with the proposed fingerprint and calculate the similarity degree of the query data and the reference data (Youssef *et al.*, 2004; Brown *et al.*, 2006). Then, we decide the reference data whose similarity degree has the highest value as the result of an identical audio search. The DTW algorithm is a pattern matching algorithm that is often used in time-series patterns such as speech processing. This algorithm can measure the similarity degree of the query and reference data and search using one part of the query data without the whole. Fig. 8 shows the matching process of the proposed fingerprint using the DTW algorithm.

In Fig. 8, W_1, W_2, …, W_n are the windows that are equal to the numbers of fingerprints of the query data. For searching with reference data, we make $n(W)$ windows from each reference data as in Eq. (4) and compare the window fingerprint and the query fingerprint.

$$n(W) = n(D_i) - n(Q) + 1, \tag{4}$$

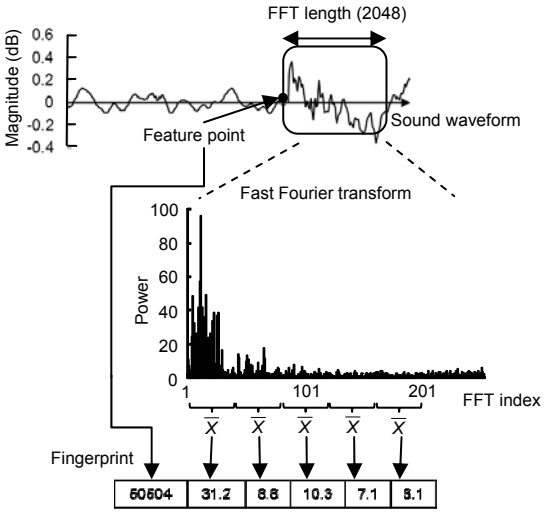

Fig. 7 Region value extraction using the low and middle frequencies of FFT

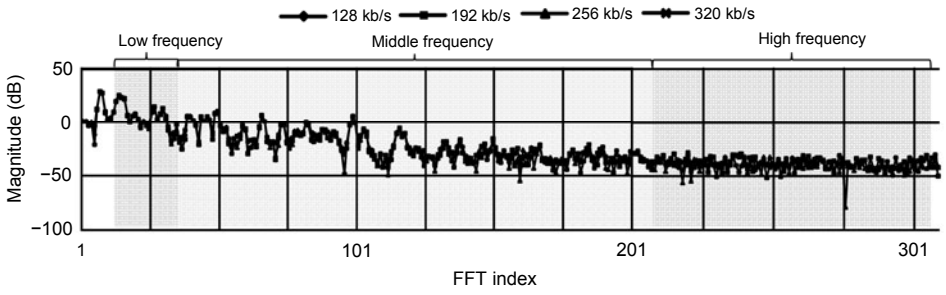

Fig. 6 Comparison of FFT frequency according to the change in bit rate

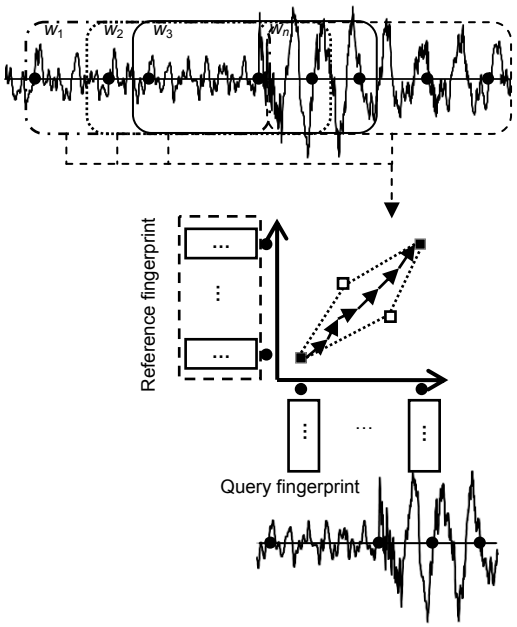

Fig. 8 The proposed fingerprint matching using the dynamic time warping (DTW) algorithm

where $n(D_i)$ is the number of fingerprints of reference data i, and $n(Q)$ is the number of fingerprints of the query data. The window for searching the reference data progresses sequentially, and the degree of similarity is measured using the DTW algorithm:

$$\mathrm{sim}(Q, D_i) = \frac{n(R_i)}{n(Q)} \times 100\%, \tag{5}$$

where $n(R_i)$ is the number of fingerprints of reference data that are identical to the fingerprints of the query data, and sim(Q, D_i) is the degree of similarity. Consequently, according to Eq. (5), the reference data that has the highest degree of similarity is the last search result.

5 Experiment and analysis

We conducted the same audio search experiment using the proposed fingerprint of the audio. An Intel 3.0 GHz Pentium IV computer with 2 GB of RAM was used. The C++ programming language was used to extract the fingerprints and MySQL 5.1.39 for the search database. We collected a set of 200 audio files which consisted of four types of music—ballads, dance, jazz, and rock. We divided the experiment data by genre to compare the number of fingerprints according to the audio genre and the audio features. All audio data files used in the experiment were MP3 files in stereo, 16 bits, and 44.1 kHz. When the fingerprint was extracted, $\sqrt{2}$ was used for the k value to calculate the DoG data.

To extract the proposed fingerprint, we needed a suitable number of DoG datasets and search range i ms. Therefore, we undertook a search experiment in which we changed the number of DoG datasets and fixed the search range to 10 ms to find a suitable number of DoG datasets. The query data was a set of 100 audio files whose bit rates were 128 kb/s and 192 kb/s, and each audio was 10 s long. The bit rate of the reference data was 128 kb/s, and all the fingerprints of each reference audio were stored in the search database. Fig. 9 shows the results graph of the search experiment and the relative computation time for changing the number of DoG datasets.

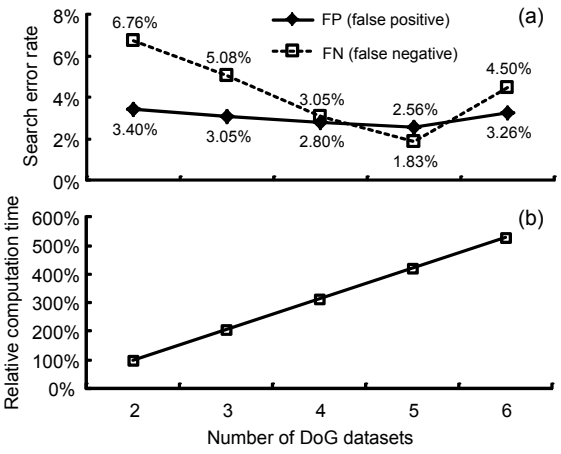

Fig. 9 Comparison of the search accuracy and computation time according to the number of DoG datasets
(a) Accuracy of the search experiment; (b) Search computation time (data length: 10 s)

Fig. 9a shows the accuracy of the audio search as the number of DoG datasets changed. False positive (FP) means that the query data and the result data are incorrect, and false negative (FN) means that a file is not found although the query audio is at the reference data. As the number of DoG datasets increased from two to five, the search error rate decreased. When the number of DoG datasets was six, however, the search error rate increased. Thus in the experiment, when the number of DoG datasets was five, the search accuracy

was the highest. Fig. 9b shows the relative computation time for the same audio search as the number of DoG datasets changed. As the number of DoG datasets increased, the computation time for search and fingerprint extraction increased. Indeed, if the length of the query data was prolonged, the computation time would increase rapidly. In this work, we used three DoG datasets to decrease the computation time.

Next, we undertook a search experiment, changing the search range and fixing the number of DoG datasets to three, to find a suitable size for *i*. The query data was a set of 100 audio files whose bit rates were 128 kb/s and 192 kb/s, and each audio was 10 s long. The bit rate of the reference data was 128 kb/s, and all fingerprints of each reference audio were stored in the search database. The search range was 5, 10, 12, 15, and 20 ms. Fig. 10 shows the results graph of the search experiment changing the search range.

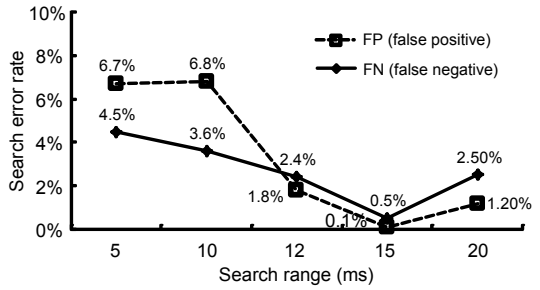

Fig. 10 Comparison of search accuracy according to search range

In Fig. 10, as the search range increased from 5 to 15 ms, the search error rate decreased. When *i*=20, however, the search error rate increased again. Thus in the experiment, when *i*=15, the search accuracy was the highest.

In these two experiments, we used three DoG datasets and *i*=15 for the same audio search using the proposed fingerprint. Table 2 shows the average number of fingerprints of the reference audio data by audio genre.

Table 2 The number of fingerprints of audio data by genre

Music type	Average number of fingerprints
Ballad	298
Dance	278
Jazz	303
Rock	307

The number of fingerprints using the proposed method was about 300 points per 4-min audio, and an average of only 3 KB of memory was needed to store these fingerprints. We experimented on the same audio search using the proposed fingerprint and the DTW algorithm. The bit rate of the reference data, a set of 200 songs, was 128 kb/s, and all fingerprints of each reference data were stored in the search database. Then, the query data was a set of 200 audio files whose bit rates were 128, 192, 256, and 320 kb/s. Each query data was 10, 30, and 60 s long at the 30, 60, and 120 s from the audio start. Table 3 shows the results of the search experiment.

Table 3 The results of same audio search using the proposed fingerprint method

Bit rate (kb/s)	Accuracy (%)			Average accuracy (%)
	10 s	30 s	60 s	
128	94	100	100	98.0
192	86	92	100	92.7
256	90	94	100	94.7
320	90	93	100	94.3
Average accuracy (%)	90	94.8	100	94.9

In Table 3, if the query data was short, the accuracy of the search was 90%. If the query data was prolonged, the accuracy of the search increased. The reason might be that the comparison targets of the fingerprint for the same audio search increased. When the audio quality of the query data and reference data was equal, the accuracy of the same audio search was 98%. When the audio quality of the query data and reference data was not equal, the accuracy of the search was 93.9%. In conclusion, the total accuracy of the same audio search was 94.9%. Google's Waveprint has demonstrated 97.5% as the best accuracy. However, in this case, the number of Google's Waveprint was increased, and the computation time also increased. The proposed method provided a performance similar to that of the existing method.

Table 4 shows a comparison of the average required search time of Waveprint of Google and the proposed method. The average required search time of the proposed method increased as the length of the query data increased, but the search time of the proposed method was similar to that of the existing method at the same query data length.

Table 4 Comparison of the average required search time

Data length (s)	Average search time (s)	
	Proposed method	Waveprint of Google
10	0.71	0.7
30	2.18	2.2
60	4.73	4.5

Table 5 shows a comparison of the number of fingerprints of the existing methods and the proposed method. The required memory of the proposed method was less than that of Google's Waveprint by 95% and less than that of Enswer.me by 50%. Therefore, we can use the same audio search that showed a similar performance to the existing methods and reduce the number of fingerprints using the proposed method.

Table 5 Comparison of the number of fingerprints of the existing methods and the proposed method

Method	Required memory per audio data (KB)	Processing quantity of 2 GB RAM
Waveprint of Google	64	About 31 000 songs
Enswer.me	6	About 300 000 songs
Proposed	3	About 600 000 songs

6 Conclusions

As the existing fingerprint methods transform total audio data into frequency data for feature extraction, the methods involve too many fingerprints. In this work, we propose a novel method that can reduce the number of fingerprints while maintaining a performance level similar to that of the existing methods. The proposed method can search for the same audio data using one part of the query data without the whole. This method can perform well even if the quality of the audio data changes.

Therefore, although the number of audio fingerprints used in the proposed method is smaller than the number for existing methods such as Google's Waveprint and Enswer.me, the proposed method still performs well in search-and-retrieval applications. The proposed method is immediately applicable as an add-on to existing audio search systems and can be used effectively in the same audio search and music copyright protection.

In future work, we will undertake a more detailed comparison of the many other existing methods with our method.

References

Baluja, S., Covell, M., 2007. Audio Fingerprinting: Combining Computer Vision & Data Stream Processing. Proc. Int. Conf. on Acoustics, Speech, and Signal Processing, p.213-216. [doi:10.1109/ICASSP.2007.366210]

Baluja, S., Covell, M., 2008. Waveprint: efficient wavelet-based audio fingerprinting. *Pattern Recogn.*, **41**(11): 3467-3480. [doi:10.1016/j.patcog.2008.05.006]

Brown, J.C., Hodgins-Davis, A., Miller, P.J.O., 2006. Classification of vocalizations of killer whales using dynamic time warping. *J. Acoust. Soc. Am.*, **119**(3):EL34-EL40. [doi:10.1121/1.2166949]

Cano, P., Batlle, E., Kalker, T., Haitsma, J., 2005. A review of audio fingerprinting. *J. VLSI Signal Process.*, **41**(3):271-284. [doi:10.1007/s11265-005-4151-3]

Chung, M.B., Ko, I.J., 2010. Identical-video retrieval using the low-peak feature of a video's audio information. *J. Zhejiang Univ.-Sci. C (Comput. & Electron.)*, **11**(3):151-159. [doi:10.1631/jzus.C0910472]

Enswers Co., 2008. Method and Apparatus for Generating Audio Fingerprint Data and Comparing Audio Data Using the Same (10-2008-0098878). Korean Intellectual Property.

Independent Recording Network, 2006. Interactive Frequency Charts. Available from http://www.independentrecording.net/irn/resources/freqchart/main_display.htm [Accessed on Sept. 2010].

Kennedy, J., 2009. Digital Music Report 2009. IFPI. Available from http://www.ifpi.org/content/library/dmr2009.pdf [Accessed on Sept. 2010].

Kim, S., Unal, E., Narayanan, S., 2008. Music Fingerprint Extraction for Classical Music Cover Song Identification. Proc. IEEE Int. Conf. on Multimedia and Expo, p.1261-1264. [doi:10.1109/ICME.2008.4607671]

Kimura, A., Kashino, K., Kurozumi, T., Murase, H., 2001. Very Quick Audio Searching: Introducing Global Pruning to the Time-Series Active Search. Proc. IEEE Int. Conf. on the Acoustics, Speech, and Signal Processing, p.1429-1432. [doi:10.1109/ICASSP.2001.941198]

Logan, B., 2000. Mel Frequency Cepstral Coefficients for Music Modeling. Proc. Int. Symp. on Music Information Retrieval.

Lowe, D.G., 2004. Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vis.*, **60**(2):91-110. [doi:10.1023/B:VISI.0000029664.99615.94]

Mihak, M., Venkatesan, R., 2001. A perceptual audio hashing algorithm: a tool for robust audio identification and information hiding. *LNCS*, **2137**:51-65. [doi:10.1007/3-540-45496-9]

Mikolajczyk, K., Schmid, C., 2004. Scale & affine invariant interest point detectors. *Int. J. Comput. Vis.*, **60**(1):63-86. [doi:10.1023/B:VISI.0000027790.02288.f2]

Pickens, J., 2002. A Comparison of Language Modeling and Probabilistic Text Information Retrieval Approaches to Monophonic Music Retrieval. Proc. Int. Symp. on Music Information Retrieval.

Ponte, J.M., Croft, W.B., 1998. A Language Modeling Approach to Information Retrieval. Proc. 21st Annual Int. ACM SIGIR Conf. on Research and Development in Information Retrieval, p.275-281. [doi:10.1145/290941.291008]

Rein, S., Reisslein, M., 2006. Identifying the classical music composition of an unknown performance with wavelet dispersion vector and neural nets. *Inf. Sci.*, **176**(12):1629-1655. [doi:10.1016/j.ins.2005.06.002]

Stephen, D.J., 1999. Music Retrieval as Text Retrieval: Simple yet Effective. Proc. 22nd Int. ACM SIGIR Conf. on Research and Development in Information Retrieval, p.297-298. [doi:10.1145/312624.312727]

Wang, A.L.C., Smith, J.O., 2002. Method for Search in an Audio Database. World Intellectual Property Organization Publication WO/2002/11123A2. Available from http://www.wipo.int/pctdb/en/wo.jspIA=WO2002011123 [Accessed on Sept. 2010].

Wold, E., Blum, T., Keislar, D., Wheaton, J., 1996. Content-based classification, search, and retrieval of audio. *IEEE Multimedia*, **3**(3):27-36. [doi:10.1109/93.556537]

Youssef, A.M., Abdel-Galil, T.K., El-Saadany, E.F., Salama, M.M.A., 2004. Disturbance classification utilizing dynamic time warping classifier. *IEEE Trans. Power Del.*, **19**(1):272-278. [doi:10.1109/TPWRD.2003.820178]