ROBUST SEARCH METHODS FOR MUSIC SIGNALS BASED ON SIMPLE REPRESENTATION

Kunio Kashino, Akisato Kimura, Hidehisa Nagano, Takayuki Kurozumi

NTT Communication Science Laboratories
Nippon Telegraph and Telephone Corporation
3-1, Morinosato-Wakamiya, Atsugi-shi, 243-0198, Japan

ABSTRACT

Signal similarity search is an important technique for music information retrieval. A basic task is finding identical signal segments on unlabeled music-signal archives, given a short music signal fragment as a query. In such a task, the search must be fast and sufficiently robust against possible signal fluctuations due to noise and distortions. In this special session paper, we describe a search method designed to cope with additive interferring sounds by spectral partitioning. Then, we introduce another method designed to be robust under multiplicative noise or distortion based on binary area representation.

Index Terms— search methods, information retrieval, music, multimedia databases

1. INTRODUCTION

The approaches to music information retrieval based on similarity search can be divided into two groups according to the objective. One group aims at retrieving music similar to a query in some musical way, such as the same melody or a rearrangement of the music. The other aims at retrieving segments almost the same as the query (i.e., reference signal) by signal-level comparison with the database (i.e., stored signal). The latter is often referred to as *signal* similarity search or audio fingerprint identification.

Until around the mid 1990s, signal similarity search had not received much attention from researchers in the audio signal processing field; most of work focused on automatic music transcription and music audio description. However, recent advances in audio coding technology and storage device technology, together with widespread broadband Internet accessibility, has highlighted the need for searching music via audio signal queries.

Signal similarity search can be simply accomplished by calculating the similarity between the reference signal and each section of the stored signal. The audio signal similarity can be calculated by extracting audio feature vectors from a signal and comparing them. For example, various features, such as zero-crossing rates, power spectra, and mel-frequency cepstrum coefficients, can be used.

However, the search requires many feature comparisons; therefore, an exhaustive (brute-force) algorithm is usually impractical. Thus, search acceleration or scalability extension has been a major research issue. In this regard, in the late 1990s, Kashino *et al.* proposed a method called time-series active search (TAS) [1, 2]. TAS accelerates the search, while guaranteeing exactly the same search results as brute-force search, by exploiting mathematical property of histograms of audio features. TAS has been extended to even faster algorithms by introducing piecewise dynamic segmentation (PDS) of the feature sequence [3]. While TAS reduces the number of matching calculations, PDS reduces the computational cost required for one matching calculation. This reduction is achieved by projecting a section of the feature sequence to a lower-dimensional space and performing matching in the projected space. Hash functions are also commonly used to accelerate the search.

Another important research issue is robustness against noise and distortion. In audio fingerprinting applications, the query and stored signals cannot be assumed to be exactly the same even in the corresponding sections of the same sound; for example, the query signal may be compressed, transmitted over mobile phones, or mixed with various irrelevant sounds. For robust search, Haitsma *et al.* proposed a system [4] that employs binary representation of the sound spectrum, and accelerates the search using the hash technique. Wang developed a feature-point-based approach to increase the robustness and speed [5].

The above-mentioned methods and other fingerprinting methods have been introduced in real applications. The applications include broadcast monitoring for commercial messages and music, music information retrieval by audio queries via mobile phones, and copyright checking for music and video. However, we consider that applications of signal similarity search will not be limited to existing ones. As many Internet search sites are primarily based on *text* similarity search, we expect that signal similarity search will also become important for music and multimedia.

With this in mind, this paper focuses on robust methods for signal similarity search. Section 2 overviews a search method designed to be robust against additive noise. Section 3 describes a method that can cope with severe multiplicative distortion. Finally, section 4 concludes the paper.

2. THE DIVIDE AND LOCATE METHOD

The existing work on audio fingerprinting often assumes that the target sound is loud enough in comparison with interfering sounds. However, this is not always the case. For example, in broadcasting programs, musical pieces may be used as the background for narrations and conversations. We refer to the task of detecting and identifying the background music as background music (BGM) identification [Fig. 1].

Our approach to BGM identification is to divide a spectrogram into a number of small regions and performing matching for each region to locate in the database, which we call the divide and locate method (DAL). The idea of spectral decomposition was presented in the self-optimized spectral correlation method (SSC) [6]. However, spectral decomposition usually results in an increase of computational cost. A computationally simpler method has been also re-

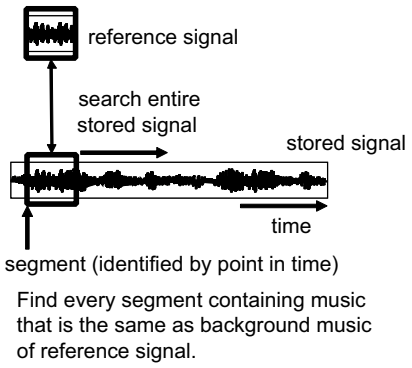


Fig. 1. Basic BGM identification task.

ported [7], but the search is still slower than that of the conventional TAS. The DAL method we discuss here further reduces computational cost in comparison with the above-mentioned methods. The basic idea is to apply vector quantization (VQ) to each decomposed component and then to search for similar components by looking up a similarity table among the VQ codes.

2.1. Method

First, time-frequency power spectra are extracted for the stored signal. Each spectrum is then decomposed into a number of small time-frequency components of a uniform size and normalized by its average power, as in Fig. 2(a), where G_{t,w_m} denotes the component at time t and the frequency band w_m of the stored signal. Next, the spectrum corresponding to each component is classified by VQ. A VQ codebook is prepared for each frequency band using the LBG algorithm. Then, an index is made for the VQ codes of the stored signal. The index is a list of positions where each VQ code appears. We perform these processes prior to the search stage.

The main part of the processing comprises the following four steps.

Step 1 The time-frequency spectra of a query are extracted and decomposed into components [Fig. 2(a)], as done for the stored signals in the preprocessing. Here, let F_{t_i,w_m} be the decomposed component at time t_i of the query, where w_m is the frequency band of F_{t_i,w_m}; $T_R = \{t_1, t_2, \ldots\}$ be the entire set of t_i of decomposed components of the query; and $W = \{w_1, w_2, \ldots\}$ be the entire set of frequency bands. These decomposed components are classified by VQ at each frequency band using the same VQ codebook as that used for the stored signals.

Step 2 As shown in Fig. 2(b), components similar to F_{t_i,w_m} are detected. This is easily done using a look-up table of the similarities between the VQ codes and the index built in the preprocessing.

Step 3 As in Fig. 2(c), the similarities with respect to each component detected in Step 2 are integrated, and the total similarity $S(t)$ for each segment on the stored signal is calculated as

$$S(t) = \frac{1}{|T_R||W|} \sum_{t_i \in T_R} \sum_{w_m \in W} s^p(F_{t_i,w_m}, G_{t+t_i,w_m}), \quad (1)$$

where $s^p(F_{t_i,w_m}, G_{t+t_i,w_m})$ is the similarity between F_{t_i,w_m} and G_{t+t_i,w_m}. In the calculation of $S(t)$, $s^p(F_{t_i,w_m}, G_{t+t_i,w_m})$ is evaluated only if G_{t+t_i,w_m} is detected as a component similar to F_{t_i,w_m} in Step 2.

Step 4 Segments whose total similarities exceed a search threshold are determined to contain the same music as the BGM of the query.

2.2. Experiments

Here, we show that the proposed algorithm achieves practical accuracy under a realistic condition, assuming the broadcasting playlist generation task.

The test signals were two 161-m audio signals: one was music and the other was speech. The music signal was prepared by connecting 161 different 50-s music signals with a 10-s gap between them. The music was chosen from the RWC Music Database (Classical Music Database and Popular Music Database) [8]. The speech comprises a narration, conversation, and artist interview, edited so that it does not include any blank segments longer than 1 s. Those signals were digitized at 11.025 kHz, 8-bit accuracy. We mixed these two signals with various SNRs, from $+3$ to -9 dB, on a computer. The SNR is defined as

$$10 \log_{10}\left(\frac{\text{average power of music signal}}{\text{average power of speech signal}}\right) \quad \text{(dB)}. \quad (2)$$

For spectrum extraction, we used FFT of 1,024 points. In this experiment, the number of components along the frequency axis was chosen to be one. The components were extracted from the signals every 50 ms. The VQ codebook size was 256.

The playlist was created as follows. First, a 5-s segment was chosen every one second of the input signal, and each segment was used as a query. This means that there were a total of 9,656 $(161 \times 60 - 4)$ queries for each SNR condition. For each query, the database, which contains 0.58 million music pieces, was searched to find a match. A piece is approximately 4.5 minutes in length on average. All of the above-mentioned 161 music pieces were included in the database. A series of continuous hits on the same music piece was concatenated to create a playlist. The accuracy was measured in terms of the recall rates R, when the similarity threshold for detection was set to a constant value that gives the 100 % precision rate (no redundant detections) throughout the tests. The P and R are defined as

$$P = \#(\text{correct retrieved titles})/\#(\text{retrieved titles}), \quad (3)$$

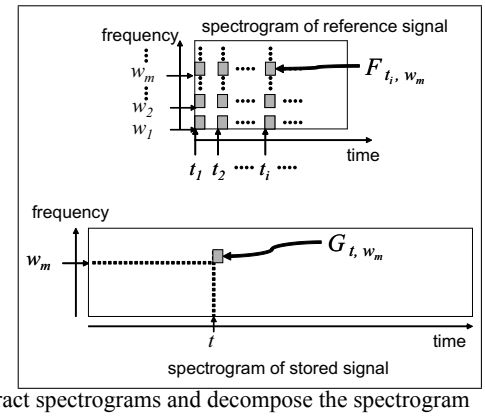
$$R = \#(\text{retrieved target titles})/\#(\text{target titles}). \quad (4)$$

The target title is the title that should appear in the playlist.

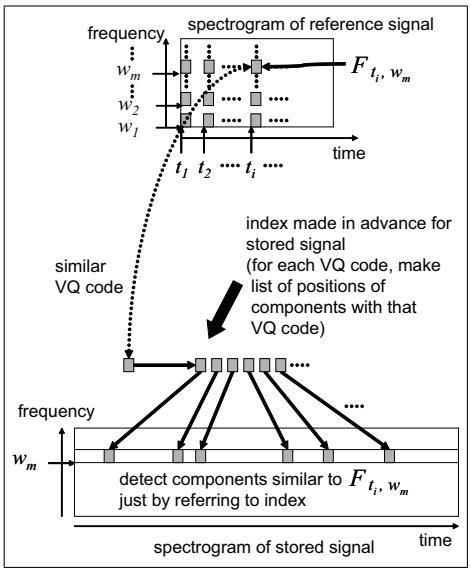
Figure 3 shows the results. The recall was better than 90 % when the SNR was above -6 dB, and was 100% when the SNR was 0 dB and 3 dB.

3. BINARY AREA MATCHING

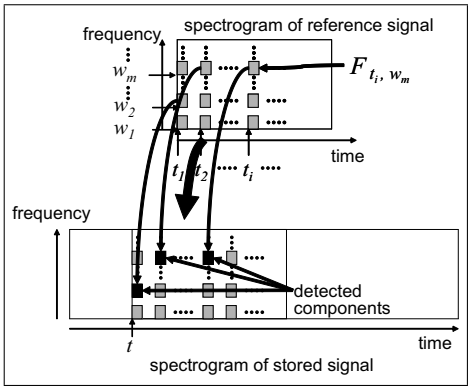
The DAL method assumes that the noise is additive; which is a valid assumption in the BGM identification task for TV and radio programs. However, generally, there are other typical noises in audio signals. For example, when we consider audio queries captured by a mobile phone, the signal may suffer severe multiplicative noise or distortion due to terminal characteristics, codec characteristics, and environment acoustics, in addition to additive noise. For such applications, the accuracy of signal matching methods may become insufficient because the feature vectors or VQ codes may be altered. A solution to this problem is to match multiple vectors or VQ codes in parallel, as in [4]. However, we here describe another approach.



(a) Extract spectrograms and decompose the spectrogram of the query into small components.



(b) Detect every similar component for each F_{t_i,w_m}.



(c) Calculate total similarity at t from local similarities of detected components.

Fig. 2. Overview of the proposed DAL search method.

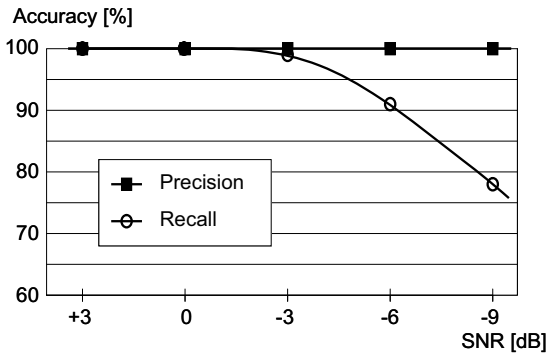


Fig. 3. Playlist accuracy with the DAL method

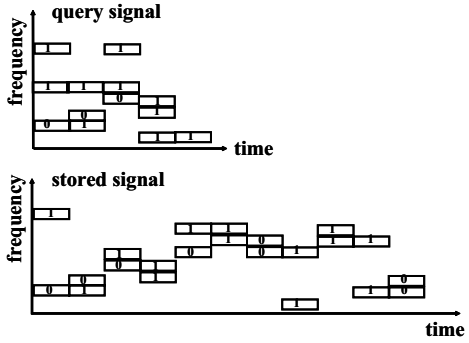


Fig. 4. Example binary area extraction

3.1. Method

Our solution is a method called binary area matching (BAM). In the BAM method, we first perform frequency analysis and then calculate the normalized power for each decomposed time-frequency region. The power normalization is done by the partial normalization method reported earlier [10]. Next, we introduce two threshold values, θ_0 and θ_1 ($\theta_0 \leq \theta_1$), and quantize the normalized power $p_n(t, f)$ for each region at time t and frequency f. That is, the power is quantized to 0 if $p_n(t, f) < \theta_0$, and 1 if $p_n(t, f) > \theta_1$. Note that the regions where $\theta_0 \leq p_n(f, t) \leq \theta_1$ are just discarded. An example of the quantized results are shown Fig. 4. We call them the binary area representation of signals. The matching calculation is simply done by counting the number of cooccurrences of the quantized codes at the corresponding positions in the maching window.

3.2. Experiments

We performed two experiments: one to compare the accuracy with a conventional method (TAS)[2], and the other to check the accuracy under a realistic setup for moblie music information retrieval.

In the first test, the stored signals were a 20-minute original music signal comprising 34 music pieces. For query signals, the same signal was played by a loudspeaker in a noisy coffee shop and captured by a mobile phone at a distance d of 0.75 meter from the loudspeaker, as shown in Figure 5. The sound was then transmitted through the public telephone network and recorded on a telephone. From the recorded signal, 400 fragments were randomly chosen as the queries. The duration of each query was 15 seconds. Both the query and stored signals were digitized at 5.6 kHz, 8-bit accuracy.

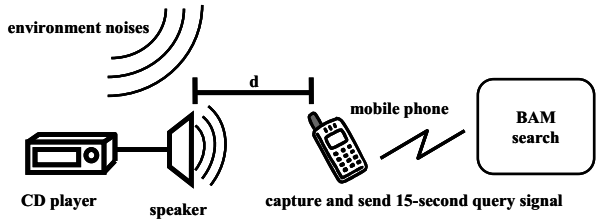


Fig. 5. Experimental setup for the BAM method.

Table 1. Experimental results with the BAM method (1).

method	BAM (proposed)	TAS (conventional)
break-even rate (P=R)	93.4%	15.9 %

Condition: a noisy coffee shop, 2G mobile phone (6.7 kbps)

The spectrogram was calculated by FFT with an analysis window of 1,024 points (samples), and the window was shifted every 100 samples. The bandwidth between 350 and 2,800 Hz was used for binary area extraction.

The output of the test system was a list of segment locations where the similarity exceeds a predefined threshold value. Each detected segment was determined correct if the time position error was less than 7.5 seconds. The accuracy was evaluated in terms of precision P and recall R rates, in terms of correct or incorrect detections and misses. As shown in Table 1, the break-even rate (P or R when the threshold was chosen so that $P = R$) is considerably improved in comparison with the TAS method.

In the second test, the stored signals were 0.23 million music pieces. A piece is approximately 4.5 minutes in length on average. For this experiment, a system was built to receive telephone calls from mobile phones and record queries. It then searched the database. If the system found a music piece that includes a segment whose similarity value exceeds a predefined threshold, then the piece was returned as a search result. Thus, the result may include multiple music pieces. If the search result included the correct title, then the trial was counted as correct. If the system did not find any segments above the similarity threshold, the trial was counted as incorrect. The accuracy was evaluated by the correct hit rate, R_h, defined as

$$R_h = \#(\text{correct hit trials})/\#(\text{trials}) \quad (5)$$

The query audio segments were all 15-s in length and chosen from music pieces stored in the database. Noise prerecorded at a busy street crossing was played back at the same time from a different speaker. The sound pressure levels of music and noise were 72 and 66 dB at 1 meter, respectively. We used 61 types of mobile phone terminals used in Japan, and the results were averaged over those terminals and the trials. The correct hit rate R_h is shown in Fig. 6. The rate was 96.7 % when d=1 [m]. The rate gradually degrades when d increases. This is principally because most of the mobile phones used in this experiment cancel distant sounds as a noise, and due to this function, the number of silent queries increased for greater d.

4. CONCLUSION

We have described signal similarity search methods called DAL (divide-and-locate) and BAM (binary area matching). DAL is based on VQ code matching corresponding to small time-frequency signal components, and BAM employs the binarized spectrogram with only statistically significant regions. These two methods have been developed for different purposes: DAL for severe additive noise and BAM for sever multiplicative noise. We plan to integrate these two methods to cope with signals including severe additive and multiplicative noise simultaneously.

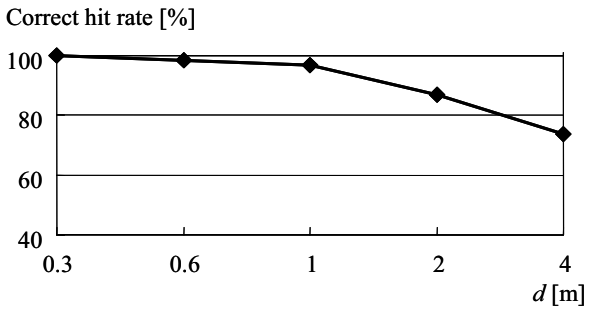


Fig. 6. Experimental results with the BAM method (2).

5. ACKNOWLEDGMENT

The authors would like to thank Dr. Junji Yamato, Dr. Shoji Makino, Dr. Hiromi Nakaiwa, and Dr. Yoshinobu Tonomura for their comments and support.

6. REFERENCES

[1] K. Kashino, G. Smith, and H. Murase: "Time-Series Active Search for Quick Retrieval of Audio and Video", *Proc. of ICASSP* (1999).

[2] K. Kashino, T. Kurozumi, and H. Murase: "A Quick Search Method for Audio and Video Signals Based on Histogram Pruning", *IEEE Trans. Multimedia*, vol.5, no.3 (2003).

[3] A. Kimura, K. Kashino, T. Kurozumi, and H. Murase: "Dynamic-segmentation-based feature dimension reduction for quick audio/video searching", *Proc. of ICASSP* (2003).

[4] J. Haitsma and T. Kalker: "A Highly Robust Audio Fingerprinting System," in *Proc. of ISMIR* (2002).

[5] A. Wang: "An Industrial Strength Audio Search Algorithm," *Proc. of ISMIR* (2003).

[6] M. Abe and M. Nishiguchi: "Self-optimized Spectral Correlation Method for Background Music Identification," *Proc. of ICME2002* (2002).

[7] H. Nagano, K. Kashino and H. Murase: "A Fast Search Algorithm for Background Music Signals Based on the Search for Numerous Small Signal Components," *Proc. of ICASSP* (2003).

[8] M. Goto, H. Hashiguchi, T. Nishimura, and R. Oka, "RWC music database: Popular, classical, and jazz music databases," in *Proc. of ISMIR* (2002).

[9] P. Cano, E. Batlle, T. Kalker, and J. Haitsma: "A Review of Algorithms for Audio Fingerprinting," *Proc. of MMSP* (2002).

[10] T. Kurozumi, K. Kashino, and H. Murase: "A Robust Audio Searching Method for Cellular-Phone-Based Music Information Retrieval", *Proc. of ICPR* (2002).