TIME-SERIES ACTIVE SEARCH FOR QUICK RETRIEVAL OF AUDIO AND VIDEO

Kunio Kashino, Gavin Smith and Hiroshi Murase*

NTT Basic Research Laboratories
3-1 Morinosato-Wakamiya, Atsugi-shi, Kanagawa, 243-0198 Japan.
kunio@ca-sun1.brl.ntt.co.jp, murase@eye.brl.ntt.co.jp
* Currently with Cambridge University, gas1003@eng.cam.ac.uk

ABSTRACT

This paper proposes a search method that can quickly detect and locate known sound (video) in a long audio (video) stream. The method is based on active search [1]. Active search reduces the number of candidate matches between reference and input signals by approximately 10 to 100 times compared to exhaustive search, while guaranteeing the same retrieval accuracy. We proposed a quick search method in [2], and here we focus on improvement of the accuracy. Thus the feature used has been extended to the audio power spectrum and temporal division of the histogram windows has been introduced to incorporate time information. Tests carried out under practical circumstances clearly show the accuracy improvement. The proposed method is still so fast that it can correctly retrieve a 15-s commercial in a 6-h recording of TV broadcasting within 2 s, once the features are calculated.

1. INTRODUCTION

This paper discusses a method to search quickly through a long audio or video stream (termed an *input signal*) to detect and locate a known reference audio or video signal (termed a *reference signal*). One application in mind is searching and retrieval of music from unlabeled audio archives, videos or the Internet. Another is monitoring occurrences of a TV commercial or the theme music of a TV program.

Even if a reference signal is known, a huge amount of computation is required for the feature matching when a long input signal is assumed. Adopting heuristic time-skipping in the matching process may partially reduce the computational load, but may also result in deterioration of the recall rate [1] (increase of misses).

We proposed a quick audio retrieval method using the active search algorithm [2]. However, the method was not necessarily accurate enough under practical circumstances (e.g. search for real TV recordings). Therefore we focus on improving retrieval accuracy maintaining the quickness that characterizes active search. To this end, the feature is extended to the audio power spectrum. In addition, the temporally divided histogram windows are introduced to incorporate time information. The framework also integrates video retrieval using color features.

Section 2 overviews the search algorithm. Section 3 evaluates the speed and accuracy of the algorithm using recordings of real TV broadcasting. Concluding remarks are given in Section 4.

[1] The recall rate is defined as the number of correctly retrieved objects divided by the number of objects that should be retrieved. The precision rate (appearing in Section 3) is defined as the number of correctly retrieved objects divided by the number of all retrieved objects.

2. SEARCH ALGORITHM

2.1. Overview

Figure 1 outlines the proposed algorithm. Firstly, the feature vectors are calculated from both the reference signal and input signal. The windows are then applied to both the reference and input feature vectors. The feature vectors over the windows create the histogram. The window length may be the same as the reference signal duration. For the incorporation of time-sequence information, however, the windows can be temporally divided into N_{div} subwindows as shown in the figure. Thirdly, similarity between the reference histogram and input histogram is calculated. When the similarity exceeds a threshold value chosen in advance, the reference signal is detected and located. In the last step, the window on the input signal is shifted forward in time and the search proceeds.

2.2. Feature Extraction

The features are the audio power spectrum and colors.

Audio feature vector $f(k)$ is written as

$$f(k) = (\ f_1(k),\ f_2(k),\ \cdots,\ f_N(k)\ )\ , \tag{1}$$

where k is the sampled time. An element of $f(k)$ is the normalized short-time power spectrum, which is given as

$$f_j(k) = \alpha(k)\ Y_j(k), \tag{2}$$

$$Y_j(k) = \sum_{t=k-M+1}^{k} y_j^2(t), \tag{3}$$

$$k = lM \quad (l = 1, 2, \cdots), \tag{4}$$

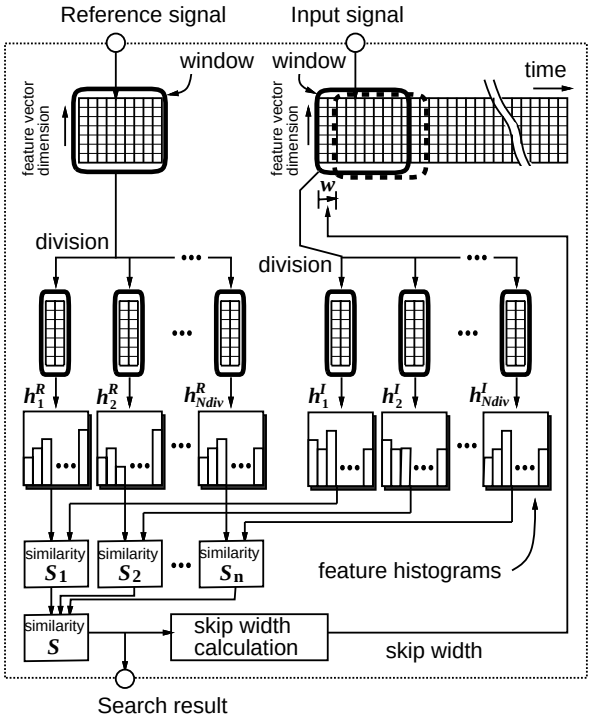

Figure 1: Block diagram of the proposed search algorithm.

where $y_j(t)$ is the output waveform of bandpass filter j at time t, M is the time support of the feature vector, N is the number of frequency channels, and $\alpha(k)$ is a normalization constant defined as

$$\alpha(k) = \frac{1}{\max_j(Y_j(k))}. \tag{5}$$

Bandpass filter $y_j(t)$ can be implemented as a 2nd order IIR filter that is computationally inexpensive. Other audio features, such as cepstral coefficients, may be used; however, the cepstrum is computationally expensive and not as suitable for quick searching as the feature discussed here.

To calculate the video feature vector, the image in each video frame is divided into D pieces of subimages. Letting p designate the video frame number and d the subimage ($d = 1, \cdots, D$), the video feature vector $u(p,d)$ is given as

$$u(p,d) = (\ u_r(p,d),\ u_g(p,d),\ u_b(p,d)\ ), \tag{6}$$

where u_r, u_g, and u_b respectively denote the red, green, and blue values averaged over the pixels in each subimage. For example,

$$u_r(p,d) = \frac{1}{|I(p,d)|} \sum_{q \in I(p,d)} r(q), \tag{7}$$

where $r(q)$ is the red value of pixel q , $I(p,d)$ is the subimage, and $|\cdot|$ stands for the number of pixels. The color information is employed because it has been successfully applied in visual object recognition [3, 1].

2.3. Histogram Modeling

A histogram is used as a non-parametric signal model for both the reference and input signals over the window considered. Swain *et al.* have shown that the histogram space provides sufficient inter-object discrimination in vision [3].

The similarity between the reference and input feature vector histograms over the windows can be determined in several ways; for example, histogram intersection. The histogram intersection for the i-th subwindow is defined as

$$S_i(h_i^R, h_i^I) = \frac{1}{L} \sum_{l=1}^{L} \min(\ h_{il}^R,\ h_{il}^I\ ), \tag{8}$$

where h_i^R and h_i^I are the histograms for the reference and the input signal respectively, and L is the number of histogram bins. The similarity over the whole windows, S, is defined using S_i as

$$S(h^R, h^I) = \min_i(\ S_i(h_i^R, h_i^I)\ ). \tag{9}$$

The histogram intersection measure is used because it is computationally simple, lends itself to an analytically-simple upper bound theorem [1, 2], and because it has been used successfully in visual object detection [3].

However, it is necessary to choose a suitable number of bins and binwidths for each dimension. As the number of bins increases, the computation increases. Moreover, the resolution of the histogram model may become so fine that noise-corrupted feature vectors may significantly distort the histogram. However, if the number of bins is too low, the histogram can not sufficiently discriminate between different audio or visual objects. In our experimentation, the number of bins are chosen empirically as described in Section 3. When the number of bins for each element is b and the feature dimension is N, the total number of bins L is given by

$$L = b^N. \tag{10}$$

The bin boundaries are selected so that the same number of feature vectors fall in the bins for each dimension. This is done by sampling feature vectors before the search procedure starts.

2.4. Similarity Upper Bound and Skip Width

As the window for the input signal shifts forward in time, similarity based on the reference and input feature vector histograms shows considerable correlation from one time step to the next. The time-series active search algorithm takes advantage of this by computing an upper bound of the similarity measure as a function of the time step and skipping all intermediate time-step similarity evaluations until this upper bound exceeds the detection threshold [2].

The upper bound on $S(h_i^R, h_i^I)$ is

$$S_{ub}(h_i^R, h_i^I(n_2)) = S(h_i^R, h_i^I(n_1)) + \frac{n_2 - n_1}{P_i}, \tag{11}$$

where $h_i^I(n_1)$ and $h_i^I(n_2)$ are the histograms created by the input window for frame numbers n_1 and n_2, and P_i is the number of frames (= the number of feature vectors) in each histogram [2]. Using Eq.(11), the derivation of the skip width for i-th subwindow is straightforward:

$$w_i = \begin{cases} \text{floor}\,(P_i(\theta - S_i)) + 1 & \text{if } S_i < \theta, \\ 1 & \text{otherwise,} \end{cases} \quad (12)$$

where w_i is the skip width, and floor(x) means the greatest integral value less than x. Thus, the skip width w for the whole window is given as

$$w = \max_i(\, w_i \,) \,. \quad (13)$$

2.5. Detection Criterion

It is necessary to decide the detection threshold above which a histogram intersection indicates a correct match. Preliminary investigations show that the histogram intersection plot for different reference templates has different descriptive statistics *e.g.* mean, standard deviation, and correct match values. Thus, a detection threshold that can adapt to different reference templates is required. We chose the search threshold, θ, such that

$$\theta = m + c\,\sigma, \quad (14)$$

where m and σ are the mean and standard deviation of the similarity values obtained from the feature vector sampling, and c is an empirically determined constant.

3. EXPERIMENTS

The proposed method was implemented on a workstation (SGI O_2) and tested with regard to search speed and accuracy using recordings of real TV broadcasts.

3.1. Experiment 1: Search Speed

The task was to search for and find a commercial (15 s) in a video recording of TV broadcasting (6 h).

In the audio feature extraction, the audio track (VHS Hi-Fi format) of the recording was first digitized at 11.0kHz sampling frequency and 8bit quantization accuracy, and then analyzed by a seven-channel (N=7) 2nd-order IIR bandpass filter bank (the filter Q=15). The filter center frequencies were equally spaced in a log frequency scale. The feature vectors [Eq.(2)] were calculated every 128 input samples (M=128).

In the video feature extraction, the video signal (NTSC) was captured at 30 frames/s. Each frame image was divided into six subimages (D=6) and feature vectors were calculated [Eq.(6)].

The time required for the search comprises (1) the feature extraction time and (2) the search time based on the extracted feature vectors.

(1) The feature extraction in this experiment was performed for 6 h and 15 s worth of signals in total. The CPU time needed was approximately 175 s for the audio feature and 50 s for the video feature.

(2) The search time depends on the reference signal, input signal, number of histogram bins, number of temporal divisions of windows, and detection threshold. Typical search times are shown in Table 1; here, the number of histogram bins for each feature dimension, b in Eq.(10), was 3 for the audio feature and 8 for the video feature.

The bottom two rows in the table are the results based on the ZCR (zero-crossing rate) feature employed in [2] (In [2], waveform sampling frequency and quantization accuracy were respectively 44.1 kHz and 16bits, but here they are 11.0 kHz and 8 bits). In all cases in Table 1, the search results were correct (*i.e.* no misses and no surplus detections).

Table 1: Search time

Feature	N_{div}	CPU time*	Speed-up	Result
Audio (spectrum)	1	0.68 (24.9) s	37 times	Fig.2
Audio (spectrum)	4	1.02 (24.7) s	24 times	Fig.3
Video	1	1.09 (22.1) s	20 times	Fig.4
Video	4	1.18 (21.4) s	18 times	Fig.5
Audio (ZCR) [2]	1	0.62 (11.1) s	18 times	Fig.6
Audio (ZCR)	4	0.73 (10.9) s	15 times	-

* In (), the CPU time in case of exhaustive search (=the case where w is fixed to 1) is given.

The CPU time required for search through the 6-h audio and video stream is significantly shorter than that for conventional spectral matching; in our test, it took approximately 20 min (CPU time) for conventional spectral matching using the inner products of the feature vectors, although the same feature vectors as in Experiment 1 were used.

Figs.2 to 6 show the corresponding similarity patterns; in these figures the horizontal axis is time and the vertical axis is the similarity. The circles indicate the detected places whereas the horizontal dotted lines stand for the detection threshold levels. It is clear that the introduction of time sequence information by dividing the windows enlarges the similarity margin for thresholding. It is also shown that the margin in Fig.2 is greater than that in Fig.6.

3.2. Experiment 2: Search Accuracy

The accuracy was evaluated using another TV recording. Firstly, a 20-min recording of TV broadcasting was captured twice; once as a source of reference signals and then as an input signal. The search was repeated 100 times; in each trial, a 15-s reference signal was randomly chosen from the first recording, and the latter signal was scanned. All parameter values were the same as in Experiment 1.

The results are shown in Table 2 and Fig.7. Here, the accuracy value was the average of the precision rate and the recall rate maximized by changing the c value in Eq.(14). However, the c value was fixed during the 100 repetitions. Table 2 shows that the spectrum features provide better discrimination property in comparison with the ZCR feature. This is also clearly shown in Fig.7.

4. CONCLUSIONS

This paper has proposed a search method that can quickly detect and locate a known reference audio (video) in a long audio (video) stream. The framework [2] has been extended to include the audio power spectrum and colors. In addition, the time information has been introduced by dividing the temporal windows. The experiments showed that these have improved the retrieval accuracy. The experiments also

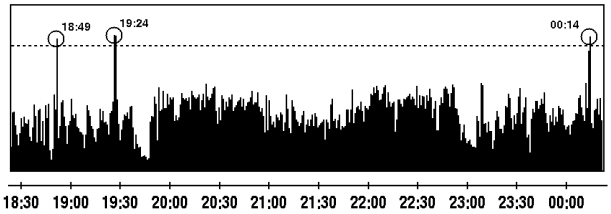

Figure 2: Search result (Audio, spectrum, N_{div}=1)

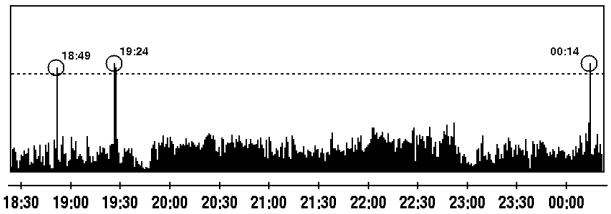

Figure 3: Search result (Audio, spectrum, N_{div}=4)

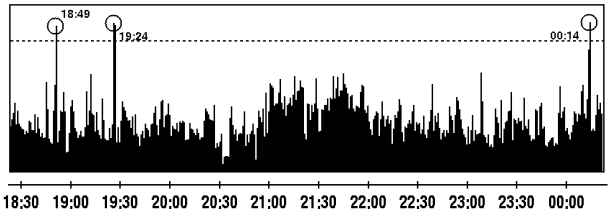

Figure 4: Search result (Video, N_{div}=1)

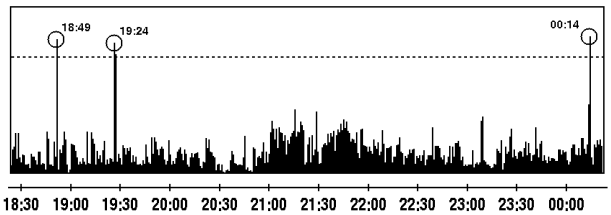

Figure 5: Search result (Video, N_{div}=4)

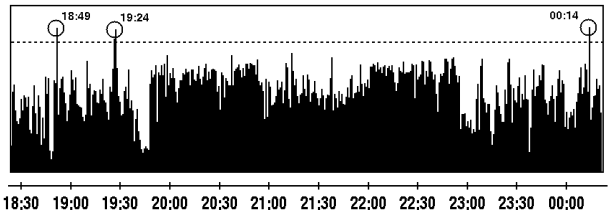

Figure 6: Search result (Audio, ZCR [2], N_{div}=1)

Table 2: Search accuracy

Feature	N_{div}	Accuracy*
Audio (spectrum)	1	99.0 [%]
Audio (spectrum)	4	100.0 [%]
Video	1	96.5 [%]
Video	4	96.5 [%]
Audio (ZCR) [2]	1	89.5 [%]
Audio (ZCR)	4	92.6 [%]

* Accuracy = (Precision+Recall)/2

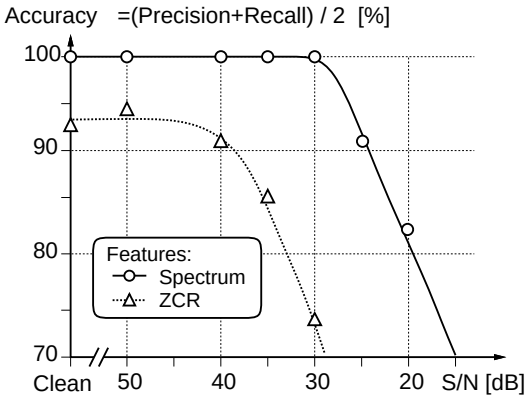

Figure 7: Search accuracy (Audio, N_{div}=4)

showed that the quickness that characterizes active search is still maintained; the proposed method can correctly detect and locate a 15-s commercial in a 6-h recording of TV broadcasting within 2 s, once feature vectors are calculated. The feature vector calculation is also fast; for a 6-h signal, it took approximately 175 s for the audio feature and 50 s for the video feature (CPU time; SGI O_2). Future work will include an application to the content-based retrieval [6, 7] and multimodal search using audio and visual features simultaneously.

5. ACKNOWLEDGMENTS

The authors wish to thank Dr. Yo'ichi Tohkura and Dr. Ken'ichiro Ishii for their help and encouragement.

6. REFERENCES

[1] Vinod V.V. and Murase H.: "Focused Color Intersection with Efficient Searching for Object Extraction", *Pattern Recognition*, Vol.30, No.10 (1997).

[2] Smith G., Murase H. and Kashino K.: "Quick Audio Retrieval Using Active Search", *Proc. of ICASSP-98*, Vol.6 (1998).

[3] Swain M. J. and Ballard D. H.: "Color indexing", *Int. J. Computer Vision*, Vol.7, No.1 (1991).

[4] Finlayson G. D. and Funt B. V.: "Color Constant Color Indexing", *IEEE Trans. PAMI*, Vol.17, No.5 (1995).

[5] Wu J. K., Narasimhalu A. D., Mehtre B. M., Lam C. P., and Gao Y. J.: "CORE: A Content-based Retrieval Engine for Multimedia Information Systems", *ACM Multimedia Systems*, Vol.3, No.1 (1995).

[6] Wold E., Blum T., Keislar D. and Wheaton J.: "Content-Based Classification, Search, and Retrieval of Audio", *IEEE Multimedia*, Vol.3, No.3 (1996).

[7] Young S.J., Brown M.G., Foote J.T., Jones G.J.F and Sparck Jones K.: "Acoustic Indexing for Multimedia Retrieval and Browsing", *Proc. of ICASSP-97*, Vol.1 (1997).