

Automated Extraction of Music Snippets

Lie Lu, Hong-Jiang Zhang

Microsoft Research, Asia

Beijing, China, 100080

{llu, hjzhang}@microsoft.com

ABSTRACT

Similar to image and video thumbnail, music snippet is defined as the most representative or highlight excerpt of a music clip, and can be used efficiently for fast browsing large number of music files. Music snippet is usually a part of the repeated melody, main theme or chorus. In this paper, we present an approach to extracting music snippet automatically. In our approach, the most salient segment of the music is firstly detected based on its occurrence frequency and energy information. Meanwhile, the boundaries of musical phrases are also detected based on the estimated phrase length and phrase boundary confidence of each frame. These boundaries are used to ensure that an extracted snippet does not break musical phrases. Finally, the musical phrases including the most salient segment are extracted as music snippet. User study indicates that the proposed algorithm works very well on our music database.

Categories and Subject Descriptors

H.5.5 [Information Interfaces and Presentation]: Sound and Music Computing - *signal analysis, synthesis and processing; systems*; H.3.1 [Information Storage and Retrieval]: Content Analysis and Indexing - *abstracting methods*.

General Terms

Algorithms, Management, Design, Experimentation, Theory

Keywords

Music snippet, music thumbnail, music saliency, musical phrase, music structure, tempo estimation

1. INTRODUCTION

Browsing is one of the most useful ways to locate desired files or data from a large collection. Image and video thumbnails are commonly used as a visual representation to support efficient browsing of large collections of images and video clips. Image

thumbnail is a down-sampled version of the corresponding image and video thumbnail is usually a key frame of the video clip. From such visual thumbnail, one can get a rough idea about content of the corresponding file to decide if more detailed examining of a file is needed. There have been many works and applications which implemented the algorithms on automatic extraction of video thumbnails from a given video clip.

This paper addresses the issue of how to define and automatically extract a music “thumbnail” that can help to facilitate efficient browsing of a large collection of music clips or sounds. Suppose a user wants to find a specific music from a music album, but can not remember the title of the music; then, he/she will have to listen to the music clips one by one until the right piece is located. Obviously, this is a very time consuming process.

To provide an effective representation of music with similar abstraction power as a thumbnail to an image for music browsing, music snippet is proposed in this paper. Similar to image and video thumbnail, music snippet is defined as the most representative or highlight excerpt of a music clip. That is, a snippet is much shorter than the original music from which it is extracted; yet, it will capture the main theme or chorus of the original music. With music snippets, one can quickly “browse” the contents of a music album even without knowing the title of each clip.

However, there are little published works on music snippet extraction. The closest works to the music snippet is music summarizations [1][2][3]. Music summarization techniques typically use a segmentation phase followed by extraction of a representative excerpt from each segment. Then, a subset of these excerpts is combined as the summary of the music. Logan [1] proposed a method for music summary using key phrases, where clustering techniques and HMM models are used to discover the structure of the music. The key phrases are extracted based on the duration and frequency of occurrence. Cooper [2] has also proposed an automatic music summarization algorithm based on similarity analysis. The resulting summarization is selected to maximize quantitative measures of the similarity between candidate excerpts and the source audio as a whole, based on the 2-D similarity matrix. This is very time-consuming since it calculates the similarity between each pair of frames. In [3], Xu presented a music summation scheme, based on a clustering method which is applied to group segmented frames into different clusters to structure the music content. Finally, the music summary is generated based on the clustering results and domain-specific music knowledge. However, this work assumed that in

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

MM'03, November 2-8, 2003, Berkeley, California, USA.

Copyright 2003 ACM 1-58113-722-2/03/0011...\$5.00

summery generation each repetitive segment pair has the same frame number and the corresponding frame pairs belong to the same cluster. This assumption is too strong, since many noises exist in the segment pair because of manual performance and feature calculation. Bartsch [4] also presented an algorithm on music thumbnailing, which is very similar to [2] on thumbnail selection, but using a different chroma-based music feature. However, these summarization or thumbnail methods are mostly focused on looking for the most frequent segments, but fail to consider how to decide the boundary of the extracted segment. As a result, a summarized segment may begin from the middle of a musical phrase, which corresponding to linguistic sentence in speech; it is definitely not desirable for listeners.

Music snippet extraction relies on detection of repeated melody. However, current algorithm for repetitive melody pattern search are mostly for MIDI file [6][7], which are not practical in real acoustic music processing. For acoustic music data, it is very difficult if not impossible to extract melody curve and to discover its repeated pattern by matching the pairs of temporal sequence.

In this paper, we propose a new approach to automated music snippet extraction. Besides looking for the position of main theme by detecting the most salient segment, the proposed approach also detects musical phrase boundaries. The phrase that contains the most salient segment in a given music is extracted as a music snippet.

The proposed approach to music snippet extraction is illustrated in the Fig. 1, which consists of four main steps: basic feature extraction, salient segment detection, music structure analysis, and snippet formation. For a given music clip, firstly, basic features are extracted from each frame, which mainly include Mel-Frequency Cepstral Coefficient (MFCC) and octave-based spectral contrast. From these basic features, a set of higher-level features are also calculated. Then, in the saliency detection module, the most salient or most representative segment of the music clip is estimated based on its occurrence frequency, energy, and positional weighting. Meanwhile, in the music structure analysis module, the boundaries of musical phrase are detected, based on estimated tempo and the confidence of a frame being a phrase boundary. Finally, music snippet is selected as the musical phrases including the most salient segment. The length of music snippet can be adjusted by the users based on their preference. Each of these four processing and analysis steps will be presented in detail in this paper.

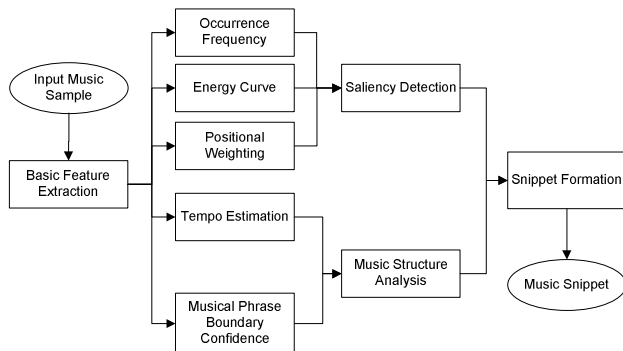


Fig. 1 A system framework of music snippet extraction

The rest of the paper is organized as follows. Section 2 discusses the basic features used in the algorithm. Section 3 presents the algorithm on estimating the most salient segment. Section 4 describes the approach to music snippet formation, following music structure analysis by detecting the boundaries of musical phrase. Experimental results and evaluations are presented in the Section 5. Conclusions are given in the Section 6.

2. BASIC FEATURE EXTRACTION

To extract basic features for snippet extraction, the input acoustic music data is firstly down-sampled into 16KHz with mono channel for uniform processing. Then, the acoustic data is pre-segmented by a sliding window of half second long with 80% overlapping. It means that the temporal resolution is 100ms. Each window is further divided into 25 ms frames for feature extraction. Silence frame is detected and removed based on its energy and zero-crossing rate.

In order to present the spectral characteristics of each segment, two main features, Mel-Frequency Cepstral Coefficient (MFCC) and octave-based spectral contrast [5], are extracted from each frame; and their statistics (mean and standard variation) are used to represent the window.

MFCC is computed from *FFT*. The log spectral coefficients are perceptually weighted by a non-linear map of the frequency scale, which is called Mel-scaling, using a triangular band-pass filter bank. Then, the Mel-weighted spectrum is transformed into *MFCC* with the COS transformation.

$$c_n = \sqrt{\frac{2}{K}} \sum_{k=1}^K (\log S_k) \cos[n(k-0.5)\pi/K] \quad n=1,2,\dots,L \quad (1)$$

where K is the number of band-pass filters, S_k is the Mel-weighted spectrum after passing k -th triangular band-pass filter, and L is the order of the cepstrum. In our method, 8-order MFCC are used.

MFCC is used with great success in speech and audio processing [2][3][8]. However, it averages the spectral distribution in each sub-band, thus loses the relative spectral information. To complement this feature, octave-based spectral contrast proposed in [5] is used. It considers the spectral peak, spectral valley and their difference in each sub-band. It can also roughly reflect the relative distribution of the harmonic and non-harmonic components in the spectrum.

Octave-based spectral contrast is also extracted from FFT. In order to ensure the steadiness of the feature, the strength of spectral peaks and spectral valleys are estimated by the average value in a small neighborhood around the maximum and minimum value respectively, instead of the exact maximum and minimum value themselves. A neighborhood factor α is used to describe the small neighborhood.

Suppose the FFT vector of k -th sub-band is $\{x_{k,1}, x_{k,2}, \dots, x_{k,N}\}$. After sorting it in a descending order, the new vector can be represented as $\{x'_{k,1}, x'_{k,2}, \dots, x'_{k,N}\}$, where $x'_{k,1} > x'_{k,2} > \dots > x'_{k,N}$.

Then the strength of spectral peaks and spectral valleys are estimated as:

$$Peak_k = \log\left\{\frac{1}{\alpha N} \sum_{i=1}^{\alpha N} x_{k,i}\right\} \quad (2)$$

$$Valley_k = \log\left\{\frac{1}{\alpha N} \sum_{i=1}^{\alpha N} x_{k,N-i+1}\right\} \quad (3)$$

And their difference is:

$$SC_k = Peak_k - Valley_k \quad (4)$$

Karhunen-Loeve transform is finally performed on these features to remove their relativity. In current implementation, 6 sub-bands are used, which include $[0, \frac{\omega_0}{2^6}), [\frac{\omega_0}{2^6}, \frac{\omega_0}{2^5}), \dots, [\frac{\omega_0}{2^2}, \frac{\omega_0}{2^1}]$, where ω_0 is the sampling rate.

These two feature sets can then be concatenated into a combined feature vector for each frame. But, the characteristics of the feature components are so different that it is not appropriate to simply concatenate these features into a feature vector. Each feature component should be normalized to make their scale similar. The normalization is processed as $x'_i = (x_i - \mu_i) / \sigma_i$, where x_i is the i -th feature component, the corresponding mean μ_i and standard derivation σ_i can be calculated from the ensemble of the training data. The normalized feature vector is considered as the final representation of a frame. Their statistics (mean and standard variation) are used to represent the characteristics of a segment of half-second window.

3. SALIENCY DETECTION

Ideally, music snippet should be an excerpt of the main melody or chorus. However, it is very difficult and time-consuming to find the exact repeated pattern by exploiting temporal sequences, since the sequence contains much noise from manual performance and feature extraction. In the proposed approach, an alternative method is employed. That is, we detect the most salient segment, instead of the repeated melody. The most salient segment is the most highlight or most representative excerpt of a music clip. Actually, the most salient segment usually locates at the main melody part; and therefore, it can be used as an indicator of repeated melody and as a suitable excerpt of music snippet in our approach. In this section, we exploit the saliency of each segment and give each segment a saliency value, from which the most salient segment can be detected.

3.1 Saliency Definition

As mentioned above, the most salient segment is usually a part of main melody, which will be repeated several times in a music or sound. Therefore, the occurrence frequency is used to evaluate the saliency, as did by clustering in [3]. Actually, besides occurrence frequency, amplitude information is also very helpful in saliency estimation. Obviously, at the highlight or chorus part in the music, the loudness, or the corresponding energy, will be much higher than those of other parts. It means that the higher the energy is, the more salient the segment is. Thus, the saliency of each segment can be measured based on the following two assumptions,

1) The saliency of a segment depends on its occurrence frequency. The most salient segment appears most frequently.

2) The saliency of a segment depends on its energy. The most salient segment has a high energy.

Meanwhile, since the main melody appears several times at different positions of the music, it is all acceptable to select the snippet from either of its first appearance, second appearance, or other appearances, if their energy is similar. Different users may have different preference on selection. To provide a more flexible approach, we also consider a positional weighting in the saliency calculation to accommodate the user's preference, although it is not a preemptory condition.

Based on the above assumptions, the saliency of each segment is defined as,

$$S_i = w_i \cdot F_i \cdot E_i \quad (5)$$

where S_i is the saliency value of i -th segment; F_i and E_i is the occurrence frequency and energy of the i -th segment; and w_i is the weighting which is set by the segment position. From the definition, the segment with the maximum saliency value can be selected as the most salient segment.

In this saliency value calculation, segment energy E_i can be calculated by many common methods. In our implementation, it is calculated as the square average of samples of the corresponding segment.

Positional weighting represents the user's preference on snippet selection. If a user prefers to select the snippet from the earlier part of a music clip than the later part, a higher weighting can be set for the segments in the earlier part, and lower weighting for those in the later part, as the following linear broken function shows.,

$$w_i = \begin{cases} 1 & i \leq N_0 \\ \frac{N-i}{N-N_0} & i > N_0 \end{cases} \quad (6)$$

where N_0 is corner point and N is the total segment number. In our implementation, such positional weighting function is used, given the earlier part of a music clip is a more preferred place for selecting the most salient segment. In general, the chorus of popular music always repeated for 2-3 times, so in our implementation, N_0 is set at the point of one third of N . Such weighting gets more acceptable result from our experiments. It should also be note this is only one kind of weighting scheme and other weighting scheme are possible as well.

In the following sub-section, we will discuss how to calculate occurrence frequency in detail.

3.2 Occurrence Frequency

Clustering is a good way to calculate the occurrence frequency. In our algorithm, a clustering method is utilized to group the segments with similar characteristics. Then the segment number of each cluster is used to measure the occurrence frequency of each segment in this cluster.

In the clustering, divergence shape [12] is used to measure the distance between two segments, as following.

$$D_{ij} = \text{tr}[(C_i - C_j)(C_j^{-1} - C_i^{-1})] \quad (7)$$

where C_i and C_j are the covariance of the feature vectors which are defined in Section 2, of segment i and j , respectively.

This measure considers the isolated two segments only. In order to give a more comprehensive representation of the distance, it is desirable that their neighboring temporal segments are taken into considerations. Suppose that the previous m and next m segments are considered with weights $[w_{-m}, \dots, w_m]$, a better distance is developed as follows.

$$D'_{ij} = \sum_{k=-m}^m w_k D_{i+k, j+k} \quad (8)$$

where a symmetric rectangle window is used as the weighting function.

Although many alternative clustering techniques can be used, LBG algorithm [9] is used in our approach because of its simplicity. Sixty-four clusters are used in our implementation. Actually, other clustering methods can be used to determine the cluster number adaptively. However, since the threshold is difficult to set, and the speed is a little slower, we did not employ them in our method.

Suppose each cluster is denoted as C_k and the number of segments in each cluster is N_k ($1 < k < 64$), where k is the index of clusters, then the occurrence frequency of i -th segment can be simply calculated as,

$$F_i = \frac{N_k}{\sum N_k} \quad (i \in C_k) \quad (9)$$

where the i -th segment belongs to cluster C_k

However, the music often shows a certain degree of local self-similarity, so that some contiguous segments are always classified into the same cluster. Thus, one vocal note may be repetitively counted in occurrence frequency calculation, which causes some noises. In our implementation, we set a threshold to avoid such a problem. If the segments are in a close range below the threshold, they will be taken as one segment; and they are counted only one time in the segment number counting. After such adjusting, the calculated value is more reasonable to represent the occurrence frequency.

4. MUSIC SNIPPET GENERATION

Once the most salient segment is detected for a given music, the next step is to extend the segment into music snippet. To this end, we need to determine the boundary of such an excerpt. It is not desirable to have a music snippet begin from the middle of a musical phrase. Thus, the boundary of the snippet should be aligned with the boundary of a musical phrase. In our approach, the musical phrases including the most salient segment are selected as music snippet.

In order to detect the boundaries of musical phrase, we should consider some characteristics of phrase, which include tempo and

vocal note duration. Based on empirical observations from popular music and some music theories, a musical phrase usually contains four bars. Thus, tempo can be used to estimate the length of a musical phrase. Meanwhile, at the end of the musical phrase, the vocal sound or music note will last relatively longer than those in the middle. It also gives a criterion for detecting musical phrase boundary. Phrase boundary confidence is thus obtained from it for each frame.

In this section, we will first present tempo estimation algorithm and the calculation of musical phrase boundary confidence, and then present our algorithm on music structure segmentation and music snippet formation. Since these algorithms should have more subtle temporal information, they are all performed on the frame-level rather than segment level.

4.1 Tempo Estimation

Tempo estimated here is used to measure the duration of two contiguous strong beats. The process of tempo estimation is illustrated in Fig. 2. After FFT is performed on each frame, an octave-scale filter-bank is used to divide the frequency domain into 6 sub-bands, as Section 2 does. The 6 sub-bands include $[0, \frac{\omega_0}{2^6}), [\frac{\omega_0}{2^6}, \frac{\omega_0}{2^5}), \dots, [\frac{\omega_0}{2^2}, \frac{\omega_0}{2^1}]$, where ω_0 refers to the sampling rate. All of the sub-bands are used to track the rhythm and tempo information. Some sub-band filtering methods are based on constant-Q band-pass filters [10]. In our approach, sub-band is segmented at FFT frequency domain directly for simple implementation without performance decrease in this application.

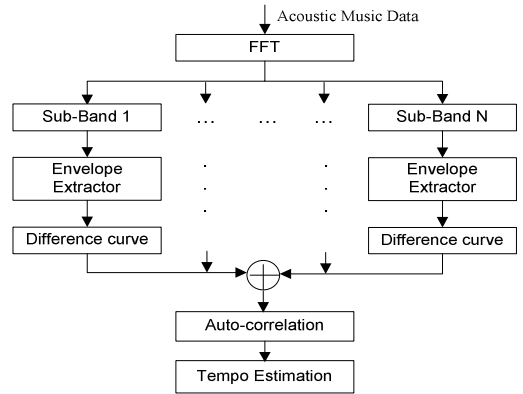


Fig. 2 The process of tempo estimation

The amplitude envelope of each sub-band is obtained by summing up the FFT coefficients of the sub-band:

$$A_i(n) = \sqrt{\sum_{k=L_i}^{H_i} A(n, k)} \quad (10)$$

where $A_i(n)$ is the amplitude of n -th frame in i -th sub-band, L_i and H_i are lower and upper bound of i -th sub-band, respectively; and $A(n, k)$ is the FFT coefficients of the n -th frame.

Then, amplitude envelope is extracted by using a half hamming (raise cosine) window.

$$A'_i(n) = A_i(n) \otimes h_w(n) \quad (11)$$

where $A'_i(n)$ is the amplitude envelope and $h_w(i)$ is the coefficients of half-hamming window,

$$h_w(n) = 0.5 - 0.5 \cos(2\pi \cdot n / 2L - 1) \quad (12)$$

where L is the length of half-hamming window and it is set as 12 in the implementation.

After getting the amplitude envelope, a Canny operator is used for onset sequence detection by estimating its difference function,

$$D_i(n) = A'_i(n) \otimes C(n) \quad (13)$$

where $D_i(n)$ is the difference function in the i -th band and $C(n)$ is the Canny operator with a Gaussian kernel,

$$C(n) = \frac{i}{\sigma^2} e^{-i^2/2\sigma^2} \quad n \in [-L_c, L_c] \quad (14)$$

where L_c is the length of Canny operator and the σ is used to control the operator's shape, which are set as 12 and 4 in our implementation, respectively.

The sum of difference curves of these six sub-bands is used to represent the rhythm information of the music.

$$D(n) = \sum D_i(n) \quad (15)$$

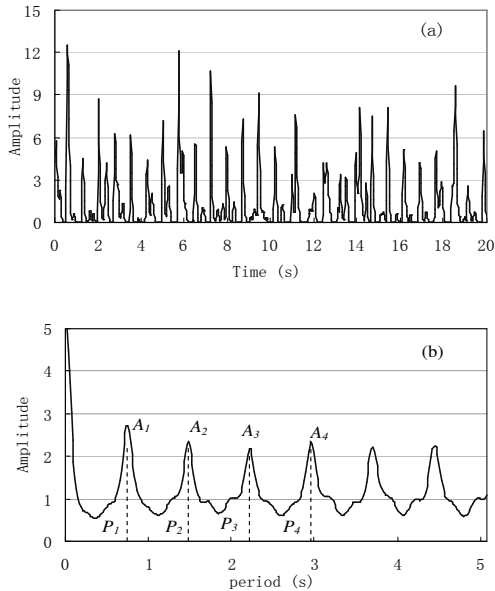


Fig. 3 Examples from a music clip of 20 seconds long (a) rhythm curve implying tempo information (b) corresponding auto-correlation curve with 5 seconds delay

Fig. 3 (a) illustrates such a rhythm curve of an example pop music clip of 20 seconds long. In order to estimate the tempo value,

auto-correlation is performed on the rhythm curve, like pitch detection in speech signal processing, and Fig. 3 (b) shows the corresponding autocorrelation curve, in which a regular tempo in the example music clip can be seen. Tempo value can be estimated as the common divisor of the peaks.

From the auto-correlation curve, the peaks are detected. Fig. 3 (b) illustrates the first four selected peaks. Their positions are denoted as P_1, P_2, P_3, P_4 , and their strengths are A_1, A_2, A_3, A_4 , from near to far from origin, respectively. Given the set of auto-correlation peaks, with magnitude and period values of each, there are many possible fundamental detection algorithms [11]. The detected fundamental is taken as the tempo value, assuming the average tempo does not vary in the music clip. Actually, most popular music has a regular and constant tempo.

The length of musical phrase can then be estimated. Under the assumption that the time signature is 4/4, which is the most common case in popular music, one bar contains two strong beats, thus, the length of a musical phrase is approximate to eight tempos.

4.2 Phrase Boundary Confidence

Based on music knowledge, at the end of a phrase, the last music note or vocal sound will be typically held longer than other notes in the middle of the musical phrase. This phenomenon can be observed from the most of popular songs. It provides us a good clue for musical phrase boundary detection. The longer the duration is, the more possible the note being the last note of a phrase. In this section, the duration of each vocal note is estimated, and then the possibility of being phrase boundary is calculated for each frame.

Vocal note duration is estimated from its energy curve. However, in songs, vocal is always mixed with instrumental sound. In order to get the duration of each vocal note, it will be ideal if we can separate the vocal from the instrumental sound. However, this task can not be implemented by current technology. Considering that vocal sounds focus on a narrower spectral band, while instrumental sounds occupy a much wider band, only the energy in such a narrow band is used for vocal energy estimation. In our real implementation, the band from 125Hz to 1000Hz is used, since most vocal sounds are in this band. The very low spectral band (less than 125Hz) is also removed to decrease the effect of bass drum, which always exists in the popular music.

Such band delimitation emphasizes more on vocal sound, although instrumental sound can not be removed clearly. It should also be noted that, for some segments without vocal sound, such delimitation also keeps the shape of amplitude curve of instrumental sound. So, it is also suitable to calculate the duration for instrumental note.

As mentioned above, in such a vocal energy curve, there still exist noises caused by other instrumental sounds and percussion sounds. In order to decrease these effects as much as possible, a smoothing process should be performed on the vocal energy curve. In the current implementation, a mid-value filtering is firstly performed on the energy curve. However, there may still exist some small fluctuations in the energy envelope of some notes.

They should be removed in order to prevent the effect of vocal shaking in singing and percussion sounds. The valleys and peaks of the curve are detected. If the difference between the contiguous valley and peak is very small, this small fluctuation is removed by simplifying it as a linear line.

After smoothing, each note is segmented by detecting the peaks and valleys. Since each note is bounded by its onset and offset, the range of each note can be considered the envelope in two contiguous valleys. As Fig. 4 shows, the duration denoted by D_1 and D_2 is a note, where D_1 can be considered as the onset duration and D_2 as offset duration. It is noted that such an envelope may also contains two notes or more. However, to achieve the objective of music snippet extraction, it is actually not necessary to extract all notes accurately.

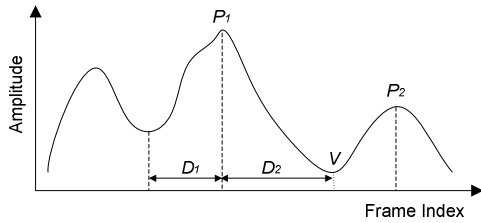


Fig. 4 A simple illustration of the note duration estimation and musical phrase boundary confidence calculation

Besides note duration, another criterion is also used to calculate the phrase boundary confidence for each frame, in our implementation. It is also observed from popular music, that if vocal energy decreases with greater magnitude, it is more possible to be near the end of a phrase. Thus, energy decrease and vocal note duration are both used to detect musical phrase boundary.

Meanwhile, since a music note is bounded by its onset and offset, each valley of the energy curve is considered as the boundary of a note. Consider the phrase boundary should be aligned with note boundary, the valleys of the curve are supposed as potential candidates of phrase boundary.

Based on these factors, we calculate the probability confidence of whether a frame represents the boundary of a musical phrase. That is, if the frame is not an energy valley, it is not possible to be a phrase boundary, and its boundary confidence is zero; otherwise, its confidence is calculated based on the energy decrease and duration of its note. A detailed formula is as,

$$PB_Conf_i \propto \begin{cases} 0 & i \notin ValleySet \\ \left(\frac{P_1 - V}{V}\right)^r (D_1 + D_2) & i \in ValleySet \end{cases} \quad (15)$$

where PB_Conf_i is the possibility of i -th frame being the boundary of a musical phrase; $ValleySet$ is the set of valleys detected in the energy curve; V is the current valley energy and P_1 is the peak energy of the corresponding note; r is the weighting for energy decrease. D_1 and D_2 are the duration of the note; as illustrated in the Fig. 4. In the real implementation, r is set as 0.5, since not every phrase boundary are with large energy decrease.

4.3 Musical Phrase Boundary Searching

In this step, the boundaries of musical phrases are determined based on the phrase boundary confidence and estimated musical phrase length, as described in the last sub-sections.

Although most musical phrases are four bars or eight tempos in general, it sometimes changes to, for example, six bars. So, in musical phrase boundary searching, we could not use the exact phrase length. Rather the musical phrase length should be a range, instead of an exact value. In our implementation, the range is set from three bars to six bars.

Thus, the musical phrase boundary is searched as the following: given the previous and the next musical phrase boundary, the new boundary is selected at the frame with largest confidence, in the range of the minimum phrase length away from the given boundaries. Then the above process is repeated until the length of all phrases is smaller than the maximum phrase length. Initially, the first and the last boundary are set at the first and the last non-silence frame, respectively.

It should be noted that this algorithm is not designed to segment each musical phrase. It is designed to ensure that the detected boundary is real musical phrase boundary. Accordingly, in the evaluations, we do not measure the performance of phrase segmentation. Instead, we only evaluate that how many extracted snippets are aligned with phrase boundaries.

4.4 Music Snippet Formation

Final music snippet formation is illustrated in the Fig. 5. The musical phrase which includes the most salient segment is used as music snippet. If the musical phrase is shorter than the required length of music snippet, the previous musical phrase or the next musical phrase will be integrated into the current snippet, based on the boundary confidence of these two phrases. The larger one is included in the snippet. Such selection ensures the highest possibility that the snippet is aligned with the phrase boundary.

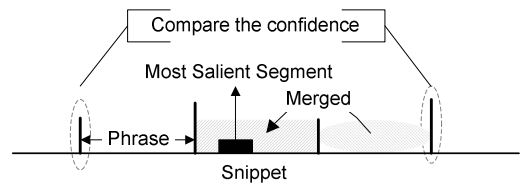


Fig. 5 An illustration on music snippet formation

The target snippet length can be set by users. However, the set length is only used as the minimum length of a snippet. The actual final length of a snippet will be determined by the length of the maximum number of musical phrases the extracted snippet covers.

5. EVALUATIONS

The evaluation of the proposed music snippet extraction algorithm has been performed on a test database of about 250 general popular songs, which include English pop music and Chinese pop

music. Most of the songs are with 44.1KHz or 48KHz, stereo and 16 bits per sample.

It is difficult to evaluate the proposed algorithm, since there is no absolute measure of the quality of extracted music snippets. In our experiments, we have evaluated the performance of our algorithm in objective and subjective manners.

In the objective evaluations, it is necessary to know which part of the music is a perfect music snippet. However, it is very difficult to get such a perfect excerpt, even by manual annotation. Different people will have different annotation. Fortunately, since music snippet is part of the repeated main melody or the chorus, if the extracted music snippet is a part of such main melody or chorus, we can say the results is reasonable. In light of this, we asked some subjects to manually select such melodies. In fact, the main melody will always appear several times. The earlier appearance with high energy is selected, which is accordant to our design of the algorithm. The annotated excerpt is taken as the ground truth. In the case of uncertainty, they are allowed to select a longer ground truth. Thus the evaluation is change to measure how much overlap between the extracted snippet and ground truth excerpt.

To measure the overlap between an extracted snippet and a ground truth excerpt, the following function, based on the similar technique measuring the precision of a retrieval system, is used

$$P = \frac{E \cap GT}{E} \quad (16)$$

where E is the extracted music snippet using our algorithm, GT is ground truth excerpt annotated manually; and P is the precision, or the overlap ratio between the estimated music snippet and truth excerpt. The larger the P is, the more reasonable the results are.

In evaluations, we compared our algorithm with other two thumbnail methods: one is similar to Logan’s summarization method [1], which selects the most frequent segment as music snippet. Another one is random selection, which selects a random segment of about N seconds in the first half of the music, which is accordant to our assumptions. The length of music snippet is set as 20 seconds in the evaluation. The average duration of annotated ground truth excerpt is about 34.9 seconds. Such evaluation can basically represent the performance of our algorithm, although the selection of ground truth excerpt is a little arbitrary.

Fig. 6 shows the comparison results in which both averaged precisions and standard derivations are both shown. From the performance comparison, it can be seen that our method has much higher precision than the other two. 77.6% of the extracted snippets are overlapped with the ground truths, In other words, our algorithm performs well on nearly 80% of music, while Logan’s selection is only 41.9%, only slightly better than the random method, which is about 32%. Sometimes, the snippets selected by our method catch another occurrence of the highlight chorus, instead of the ground truth, which contributes to the extraction errors. Logan’s method detects the most frequent segments and can catch a part of melody. However, it fails to consider the energy, thus less satisfying overlapping result.

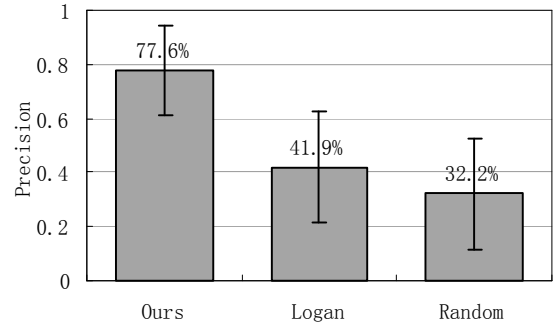


Fig. 6 Average precision comparison among our algorithm, Logan’s method and random method

In the subjective evaluations, ten subjects with music experiences were asked to score the snippet listened. The rating has three levels: 3, 2 and 1, representing good, acceptable and unacceptable, respectively. Subjects were not given any special instructions on how to measure the performance, and the evaluations were based on only their own perceptions, by comparing the extracted snippets and original music clips. In the evaluations, each subject was asked to evaluate twenty five songs, randomly selected from the database. All songs were evaluated after ten rounds.

Again, our algorithm is compared with Logan’s method and the random selection. The comparison results in terms of the average subjective score and corresponding standard derivation are shown in the Fig. 7. Our method was given a score of 2.62, about 27% higher than that of Logan’s selection and 50% better than that of random selection.

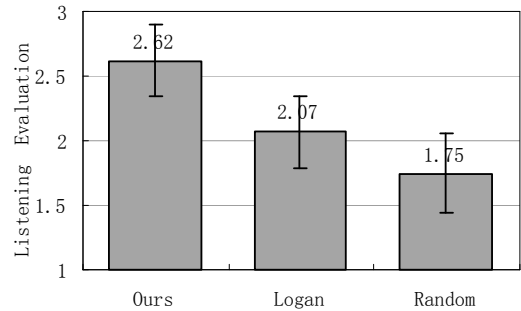


Fig. 7 Average subjective score comparison among our algorithm, Logan’s method and random method. (The subjective score level: 3–good; 2–acceptable; and 1–unacceptable.)

As shown in Fig. 7, the snippets obtained by Logan’s method are in average scored “acceptable”. However, since it only utilizes the occurrence frequency without the energy information, many of them are not good enough to be scored “good”. Logan’s method often catches the melody but misses the most highlight chorus. This fact also indicates that the energy information is very important in the snippet extraction.

It can be also seen from Fig. 7 that the random selection has a score of 1.75, which means that more than half of the random selections are also acceptable perceptually. This is because the pop music in our test database is quite repetitive so that sometimes random selection can also catch part of the melody.

Users usually give a ‘good’ score to the snippet which contains the chorus or the title of the songs, and they give an “acceptable” score if the snippet catches the other part of main melody though not the most highlight segment. Logan’s method often catches the melody but misses the most highlight chorus. That is why it has a low precision compared with the ground truth but still has an ‘acceptable’ score. Sometimes random selection can also catch the chorus part. However, because the boundaries of music snippet extracted this way often are located at the middle of a musical phrase, it causes some perceptually uneasy and thus a low score. This also shows that phrase boundary alignment has much effect on the human perception.

In the subjective evaluation, we also found some other interesting user behaviors. Subjects in general all prefer vocal portion. They give a higher score for the snippet with vocal sounds than that with instrumental sounds. The instrumental snippet always gets “unacceptable” scores, except when it also catches a highlight part of the melody.

We have also evaluated on the performance of musical phrase boundary detection. Phrase-aligned snippet improves the perceptual smoothness and always gets a high score, as observed from the subjective evaluations. In this evaluation, we do not measure all of the phrase boundaries of the music. Instead, we only measure if the extracted snippet is aligned with the phrase boundary. It can coarsely reflect the performance of musical phrase boundary detection. We compared our algorithm with random selection. In the experiments, we found that 75% of extracted snippets were aligned with the musical phrases, while only about 15% of random selections were phrase aligned, as shown in Table I.

Table I. The ratio of phrase-aligned snippet

	Ours	Random
Boundary Alignment	75%	15%

We also tested the computational complexity of our algorithm, with Pentium IV 1.8G running Windows XP, the snippet extraction process can be completed in less than 5% of the time-length of the music. That is, for a music clip of 4-minute long, the processing time is about 10 seconds.

6. CONCLUSIONS

Similar to image and video thumbnails, music snippets provide an efficient way for music browsing. In this paper, an effective approach to music snippet generation is proposed. In this approach, the most salient segment is detected firstly based on the occurrence frequency, energy information and position weighting. Then, the boundaries between musical phrases are detected, based on the estimated phrase length and boundary confidence of each frame. Finally, the phrases including the most salient segment are

selected as music snippet. The approach is able to extract the most highlight excerpt and avoid the perceptual unpleasantness caused by the broken phrases. User study has indicated that the proposed approach achieves satisfactory results.

There are still rooms to improve the proposed approach. For example, more effective features could be used. Vocal and instrumental discrimination can be performed to further avoid selecting the instrumental portion as music snippet, since users prefer vocal portion as indicated in our subjective evaluations. It can improve the music structure parsing as well, since at the end of musical phrase, there are always pure instrumental sounds.

7. REFERENCES

- [1] B. Logan and S. Chu. “Music Summarization Using Key Phrases” *Proc. of International Conference on Acoustics, Speech and Signal Processing*, Vol. II, pp 749-752, 2000
- [2] M. Cooper and J. Foote “Automatic Music Summarization via Similarity Analysis” *Proc. Intl. Symposium on Music Information Retrieval*, pp. 81-85, 2002
- [3] C. Xu, Y. Zhu and Q. Tian. “Automatic music summarization based on temporal, spectral and Cepstral features” *Proc. of IEEE International Conference on Multimedia and Expo*, 2002
- [4] M. A. Bartsch and G. H. Wakefield, “To Catch a Chorus: Using Chroma-Based Representation for Audio Thumbnailing”. *Proc. Int. Workshop on applications of Signal Processing to Audio and Acoustics*, pp 15-19, 2001
- [5] D. N. Jiang, L. Lu, H.-J. Zhang, J. H. Tao and L. H. Cai. “Music Type Classification by Spectral Contrast Features”, *Proc. of IEEE International Conference on Multimedia and Expo*, 2002.
- [6] E. Cambouropoulos, M. Crochemore, C.S. Iliopoulos, L. Mouchard and Y. J. Prinzon “Algorithms for Computing Approximate Repetitions in Musical Sequences”. *Proc. of the AWOCA'99 Workshop (Australasian Workshop on Combinatorial Algorithms)*. pp. 25-27, July, 1999.
- [7] H.-H. Shih, S. S. Narayanan, and C.-C. J. Kuo, "Automatic main melody extraction from MIDI files with a modified Lempel-Ziv algorithm", *International Symposium on Intelligent Multimedia, Video & Speech Processing*, 2001.
- [8] L. Rabiner, B.H. Juang. *Fundamentals of Speech Recognition*. Prentice-Hall, 1993.
- [9] Y. Linde, A. Buzo, and R.M. Gray. “An Algorithm for Vector Quantizer Design”, *IEEE Trans. on Comm.*, Com-28, No.1, pp. 84-95, 1980.
- [10] E. D. Scheirer. “Tempo and beat analysis of acoustic musical signals”, *J. Acoustic Society of America*, 103(1), pp. 588-601, 1998
- [11] J. Maher and J. Beauchamp, “Fundamental frequency estimation of musical signals using a two-way mismatch procedure”. *J. Acoustic Society of America*, 95(4), pp. 2254-2263, 1994.
- [12] L. Lu, H.-J. Zhang, "Speaker Change Detection and Tracking in Real-Time News Broadcasting Analysis", *ACM Multimedia*, 2002, pp. 602- 610.