SIP (2016), vol. 5, e4, page 1 of 9 © The Authors, 2016.

doi:10.1017/ATSIP.2016.4

ORIGINAL PAPER

Robust and efficient content-based music retrieval system

YUAN-SHAN LEE[1], YEN-LIN CHIANG[1], PEI-RUNG LIN[1], CHANG-HUNG LIN[1] AND TZU-CHIANG TAI[2]

This work proposes a query-by-singing (QBS) content-based music retrieval (CBMR) system that uses Approximate Karbunen–Loeve transform for noise reduction. The proposed QBS-CBMR system uses a music clip as a search key. First, a 51-dimensional matrix containing 39-Mel-frequency cepstral coefficients (MFCCs) features and 12-Chroma features are extracted from an input music clip. Next, adapted symbolic aggregate approximation (adapted SAX) is used to transform each dimension of features into a symbolic sequence. Each symbolic sequence corresponding to each dimension of MFCCs is then converted into a structure called advanced fast pattern index (AFPI) tree. The similarity between the query music clip and the songs in the database is evaluated by calculating a partial score for each AFPI tree. The final score is obtained by calculating the weighted sum of all partial scores, where the weighting of each partial score is determined by its entropy. Experimental results show that the proposed music retrieval system performs robustly and accurately with the entropy weighting mechanism.

Keywords: Query-by-singing, Music retrieval, Symbolic sequence, Pattern indexing, Information entropy

Received 31 May 2015; accepted 23 February 2016

I. INTRODUCTION

Digital music data on the Internet are explosively growing. Therefore, applications of content-based music retrieval (CBMR) system are more and more popular. Searching music by a particular melody of a song directly is more convenient than by a name of a song for people. Moreover, according to the survey from the United Nations [1], the 21st century will witness even more rapid population ageing than did the century just past; therefore, it is important to develop an efficient and accurate way to retrieve the music data.

A *CBMR* method is a more effective approach for a music retrieval system than the text-based method. A CBMR system aims to retrieve and query music by acoustic features of music, while a text-based music retrieval system only takes names, lyrics, and ID3 tags of songs into consideration.

Query-by-singing (*QBS*) is a popular method in *CBMR*. Many approaches based on *QBS* have been developed currently. Huang [2] proposed a *QBS* system by extracting the pitches and the volumes of the music. The data are used to build an index structure via *advanced fast pattern index* (*AFPI*) and *Alignment* [3] as its searching technique.

Lu *et al.* [4] proposed an extraction mechanism that regards audio data as a sequence of music notes, and then a hierarchical matching algorithm was performed. Finally, the similarity scores of each song to the query were combined with respect to the pitches and the rhythm by a linear ranking formula. This approach is accurate when an instrumental clip is given as the search key; however, the accuracy decreases when the input is the humming voice from a human. Cui *et al.* [5] introduced a music database that applied both text-based and content-based techniques simultaneously. Various acoustic features are regarded as indices and trained by a neural network mechanism. Such a design of the music database provides high efficiency and accuracy due to its good algorithms, but it lacks in portability since the implementation is too complicated. Inspired by Huang [2], a *QBS-CBMR* system was proposed in our previous work [6]. An entropy-weighting mechanism was developed to determine the final similarity.

Presently, state-of-the-art *QBS-CBMR* systems can achieve high accuracy under clean environments. However, under noisy environments, the performance might degrade due to the mismatch between the noisy feature and the clean-trained model. Motivated by this concern, this paper extends the previous work [6]. Noise effects are further reduced by applying *Approximate Karbunen–Loeve transform* (*AKLT*) [7] for preprocessing. The proposed robust *QBS-CBMR* system has five stages:

[1]Department of Computer Science and Information Engineering, National Central University, Jhongli, Taiwan
[2]Department of Computer Science and Information Engineering, Providence University, Taichung, Taiwan

Corresponding author:
Tzu-Chiang Tai
Email: tctai717@gmail.com

(1) Noise reduction: Considering the real case of music retrieval, the noise in the music clips may impact the results. Therefore, we use *AKLT* as preprocess to reduce the influence of the noise for all music clips.
(2) Feature extraction: The input music clip is first converted into a 39-dimensional *Mel-frequency cepstral coefficients* (*MFCCs*) [8, 9] and 12-dimensional *Chroma* [10]. For each music clip, there are totally 51-dimensional features. Second, each dimension of features is transformed into symbolic sequences using the adapted *symbolic aggregate approximation* (adapted *SAX*) method [11], which is proposed in this work. These symbolic sequences are also called the *SAX* representation.
(3) The *AFPI* tree structure: Following feature extraction stage, the input music is transformed into 51 symbolic sequences with respect to 51 features. In the proposed *QBS-CBMR* system, symbolic sequences are regarded as a search key. Finally, these symbolic sequences are stored by a tree structure called the *AFPI* tree [2] due to high efficiency for the retrieval task.
(4) Score calculation: The results of the music retrieval task are determined by the "scores". After the two stages mentioned above, music clips are transformed into 51 *AFPI* trees. A partial score is calculated for each *AFPI* tree first. The final score is then obtained by the weighted summation of all partial scores, where the weighting of each partial score is determined by its entropy [12]. The higher scores denote the higher similarity between the query music clip and the songs in the database.

The rest of this paper contains the following sections: Section II briefly reviews related works. Section III briefs the overview of the proposed music retrieval system. The details of the feature extraction stage are discussed in Section IV. Section V describes how the database works to search the music clip input in detail. In Section VI, we present the performance of the proposed system through some experiments. Finally, Section VII concludes the paper.

II. RELATED WORKS

A) Music content representation

The *MFCCs* were first proposed by Davis and Mermelstein in 1980 [13]. The *MFCCs* are non-parametric representations of the audio signals and are used to model the human auditory perception system [9]. Therefore, *MFCCs* are useful for audio recognition [14]. This method had made important contributions in music retrieval to date. Tao *et al.* [8] developed a *QBS* system by using the *MFCCs* matrix. For improved system efficiency, a two-stage clustering scheme was used to re-organize the database.

On the other hand, the *Chroma* feature proposed by Shepard [10] has been applied in studies of music retrieval with great effectiveness. Xiong *et al.* [15] proposed a music retrieval system that used *Chroma* feature and notes detection technology. The main concept of this system is to extract a music fingerprint from the *Chroma* feature. Sumi *et al.* [16] proposed a symbol-based retrieval system that uses *Chroma* feature and pitch features to build queries. Moreover, to make the system with high precision, *conditional random fields* has been used to enhance features.

Chroma features can work well when queries and reference data are played from different music scores. It has been found that *Chroma* features can identify songs in different versions. Hence, we can use *Chroma* features to identify all kinds of songs, even cover versions [17]. This research extends our previous work [6]. Compared with [6], a new feature vector containing 39-*MFCCs* features and 12-*Chroma* features are extracted.

B) Noise reduction

The actual application must eliminate environmental noise. Otherwise, the accuracy of the music retrieval results decreases. Shen *et al.* [18] proposed a two-layer structure *Hybrid Singer Identifier*, including a preprocessing module and a singer modeling module. In the preprocessing module, the given music clip is separated into vocal and non-vocal segments. After the audio features are extracted, vocal features are fed into *Vocal Timbre* and *Vocal Pitch* models, and non-vocal features are fed into *Instrument* and *Genre* models. It had been proven that the work of [18] is robust against different kinds of audio noises. However, the noise is not removed and so the performance will be still affected by noise.

Mittal and Phamdo [19] proposed a *Karhunen–Loeve transform* (*KLT*)-based approach for speech enhancement. The basic principle is to decompose the vector space of the noisy speech into two subspaces, one is speech-plus-noise subspace and the other is a noise subspace. The signal is enhanced by removing the noise subspace from the speech-plus-noise subspace [20]. The *KLT* can perform the decomposition of noisy speech. Since the computational complexity of *KLT* is very high, the proposed system uses *AKLT* with wavelet packet expansion [7] to process the noise reduction of input music clips.

III. THE SYSTEM ARCHITECTURE OVERVIEW

The structure of the proposed system is developed based on the work of Huang [2]. Figure 1 demonstrates the proposed system. The feature extraction stage converts music files into 51 symbolic sequences, which are stored using tree structures. The methods used in the feature extraction stage are discussed in detail in Section III.

The feature extraction stage mainly contains two steps: (1) transform music files into 39 of the *MFCCs* features [8, 9] and 12 of the *Chroma* features [10]; (2) convert each dimension of *MFCCs* into a symbolic sequence by the *piecewise aggregate approximation* (*PAA*) method [11] and the adapted *SAX* [11].

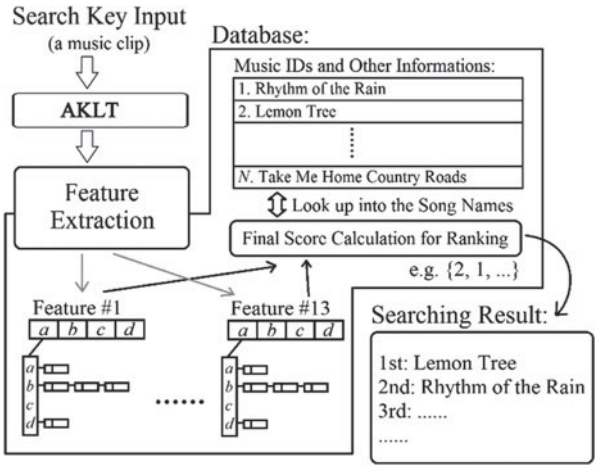

Fig. 1. The main structure of t he proposed music retrieval system.

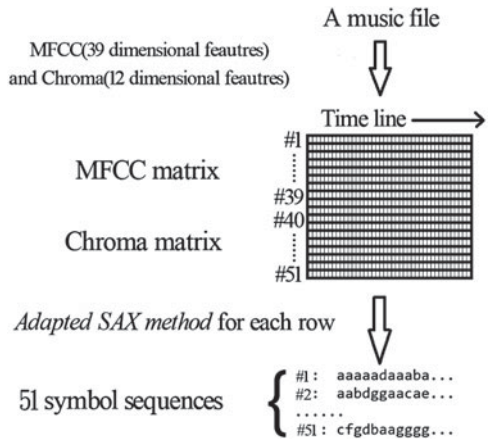

Fig. 2. A diagram for the *music retrieving process* in our system.

After feature extraction, each of the 51 symbolic sequences is then stored using a tree structure called the *AFPI* tree. Next, the 51 *AFPI* trees are used to generate a final score to evaluate the similarity between the query music clip and the songs in the database.

Two components stored in the database for each song are:

(1) 51 *AFPI* tree structures.
(2) Music IDs and other information.

The searching process only accesses these components instead of the original audio files, so that the proposed music retrieval system is portable.

In the proposed implementation, two music retrieval-related operations are performed: adding a complete music file into the database (the *ADD* operation), and searching from the database with a music clip file (the *SEARCH* operation). Both operations run the *music retrieving process* and access the tree structures in the database. However, this study is focused on the *SEARCH* operation for the following reasons:

- For a user, *searching* a database to find a song is more desirable than simply "donating" (*adding*) a song to the database.
- The only two differences between *ADD* and *SEARCH* are: (1) *ADD* builds the structure, while *SEARCH* searches the structure; (2) *SEARCH* analyzes the result from the database structures, while *ADD* does not.

IV. MUSIC RETRIEVAL PROCESS

Figure 2 shows the feature extraction stage, which is performed as follows:

A) MFCCs

*MFCC*s are non-parametric representations of audio signals, which model the human auditory perception system [9, 13]. Therefore, MFCCs are regarded as a useful feature for audio recognition.

The derivation of *MFCC*s is based on the powers of the Mel windows. Let X_ω denotes the ωth power spectral component of an audio signal, S^k be the power in kth Mel window, and M represents the number of the Mel windows usually ranging from 20 to 24. Then S^k can be calculated by:

$$S^k = \sum_{\omega=0}^{F/2-1} W_\omega^k \cdot X_\omega, \quad k = 1, 2, \ldots, M, \tag{1}$$

where W^k is the kth Mel window, and F is the number of samples in a frame, which must be a power of 2, and usually set to 256 or 512 that makes each frame ranging from 20 to 30 ms approximately.

Let L denote the desired order of the *MFCC*s. Then, we can calculate the *MFCC*s from logarithm and cosine transforms.

$$c_n = \sum_{k=1}^{M} \log\left(S^k\right) \cos\left[(k - 0.5)\frac{n\pi}{M}\right], \quad n = 1, 2, \ldots, L. \tag{2}$$

B) Chroma feature

Shepard proposed the use of tone height and *Chroma* to perform the perception of pitch [10]. The *Chroma* vector can be divided into 12 semi-tone families, between 0 and 1 into 12 equal parts. Additionally, 12 semi-tones constitute an octave. Shepard conceptualized pitch perceived by humans as a helix with a 1D line. Figure 3 illustrates this helix with its two dimensions. The vertical dimension is the continuous tone height, and the angular dimension is the *Chroma*. The Shepard decomposition of pitch can be expressed as

$$f_p = 2^{h+c}, \tag{3}$$

where p is pitch, f is frequency, h is tone height, $c \in [0, 1)$, and $h \in Zh \in Z$.

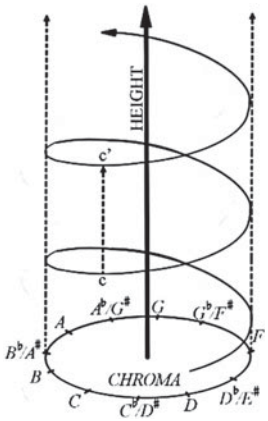

Fig. 3. Shepard helix of pitch perception. The vertical dimension is tone height, and the angular dimension is chroma.

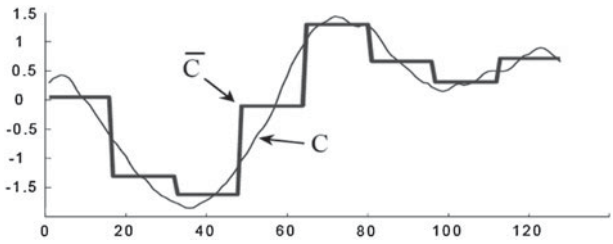

Fig. 4. A 128-dimensional time series vector is reduced to an 8-dimensional vector of *PAA* representation [11].

The *Chroma* for a given frequency can then be calculated as follows:

$$c = \log_2 f_p - \lfloor x \log_2 f_p \rfloor, \tag{4}$$

where $\lfloor \cdot \rfloor$ denotes the greatest integer function. *Chroma* is the fractional part of the 2-based logarithm of frequency. Like the ideas of pitch, some frequencies are mapped to the same class.

C) PAA and the adapted SAX

The *PAA* method [11] reduces an n-dimensional vector into a w-dimensional one. The vector generated by the *PAA* method is called the *PAA* representation. An example of the *PAA* representation is demonstrated in Fig. 4.

After converting each row of the feature matrix into the *PAA* representations, the adapted *SAX* method is applied to the *PAA* representations to construct symbolic sequences shown in Fig. 5. The adapted *SAX* method is developed based on the work of Lin *et al.* [11]. The difference between these two methods is that the original *SAX* uses the *Gaussian* distribution curves, while the adapted *SAX* uses *Cauchy* distribution, whose probability density function (*PDF*) and cumulative distribution function (*CDF*) curves both look similar to the *Gaussian* curves. Figure 6 illustrates the difference.

When implementing both of the original *SAX* or the adapted *SAX* method, the inverse function of the *CDF* of the specified distribution is required in order to determine, where the breakpoints are located (see Fig. 7).

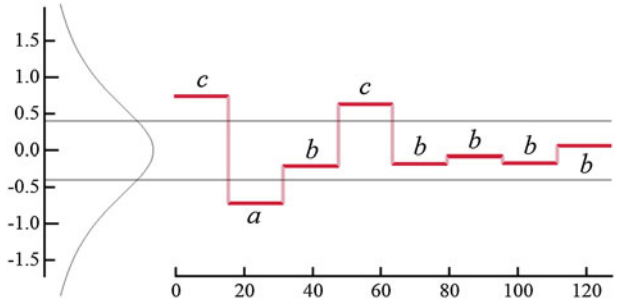

Fig. 5. An illustration of a symbolic sequence. The *PAA* representation shown in Fig. 4 is converted into a symbolic sequence of three distinct symbols: a, b, c, via the original *SAX* method.

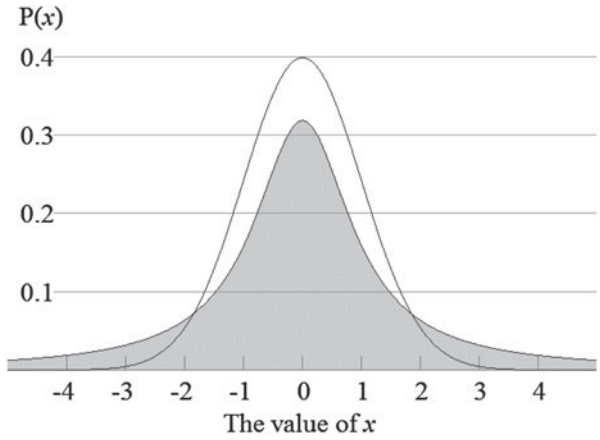

Fig. 6. The *pdf* curves for the standard *Cauchy* and the standard *Gaussian* distributions. The curve exactly on the solid-colored area is the *Cauchy* distribution curve.

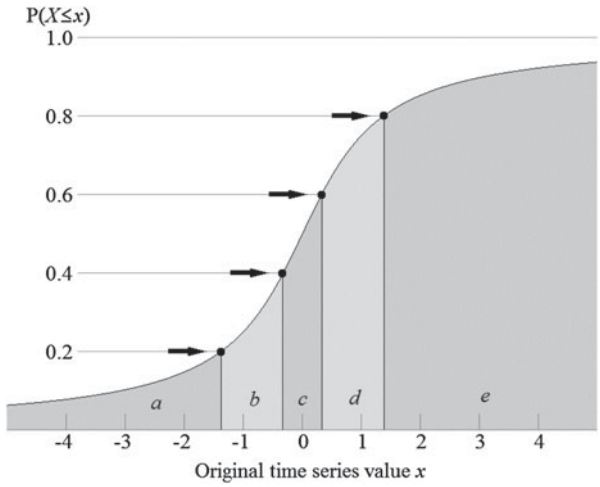

Fig. 7. An example of a cumulative distribution function (*CDF*) P of the time series variable x. When $n = 5$, the breakpoints are located in the positions of P(1/5), P(2/5), P(3/5), and P(4/5), respectively. To find x from P(x), the inverse function of a *CDF* is required.

The *CDF* of the *Gaussian* distribution is represented as the following equation:

$$\frac{1}{2}\left[1 + \operatorname{erf}\left(\frac{x-\mu}{\sqrt{2\sigma^2}}\right)\right], \tag{5}$$

where erf($\cdot$) denotes the complementary error function.

Based on the work of Huang [2], this paper proposes the adapted *SAX* by applying *Cauchy* distribution. The *CDF* of the *Cauchy* distribution is represented as:

$$\frac{1}{\pi}\left[1+\arctan\left(\frac{x-x_0}{\gamma}\right)\right]+\frac{1}{2}, \tag{6}$$

where $x_0 x_0$ is the location parameter, and $\gamma\gamma$ represents the half of the interquartile range.

The adapted *SAX* with (6) is a more feasible and convenient way than the original *SAX* with (5). Both (5) and (6) are *increasing* functions for all real numbers x, while (5) contains the *Gaussian error function* (erf), whose *inverse function* does not commonly exist in many standard libraries in popular programming languages such as $C++$ and *Java*. However, (6) has a "special function", the *arctan* function, whose inverse function is identical to the *tangent* function in almost all standard libraries of the major programming languages. Hence, implementation of the adapted *SAX* with (6) is more feasible and convenient compared with implementation of the original *SAX* with (5).

V. THE STORAGE STRUCTURES AND THE SCORING MECHANISM OF THE DATABASE

The storage of the database contains 51 *AFPI* tree structures [2]. Each tree is constructed by the *SAX* representation of the corresponding *MFCCs* feature. Firstly, the partial score for the similarity between the query music clip and the songs in the database is calculated from each *AFPI* tree. Secondly, the final score for the overall similarity is determined by the proposed weighted summation of the partial scores, with respect to the entropy [12] of the *SAX* representation in each feature of the query.

The proposed *QBS-CBMR* system only stores the *AFPI* trees and the information of the songs in the database. Therefore, the proposed system is portable. The two methods used to build the storage structure in the proposed system are described in detail as follows.

A) AFPI structure

Figure 8 shows an *AFPI* tree. The root of the tree consists of n pointers for each distinct symbol that can possibly be generated in the *SAX representation* [2, 3]. Each pointer in the root for the symbol S_i points to a child node, which contains n sets of *pattern relation* records, respectively, in which each set is for another symbol S_j that can be either the same as or different from S_i. Each record of *pattern relation* contains an integer for the *music ID* and a K-bit binary code for storing relation states.

The kth bit from the right in the K-bit binary code determines whether there exists a *subsequence* of the *SAX representation* from a particular *music ID*, in which the *subsequence* has a length of $k+1$ beginning with S_i and ending with S_j.

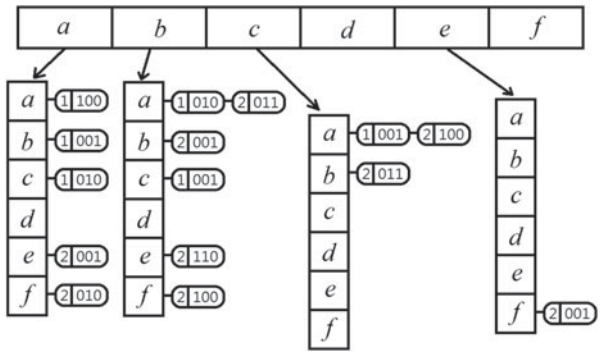

Fig. 8. Example of AFPI tree structure, where $n = 6$ and $K = 3$. This figure is redrafted from [2].

Table 1. Two examples of pattern relations.

		Pattern relation		
Song ID	*SAX* representation	S_i	S_j	Binary code
2	*cbbaef*	*c*	*b*	011
		c	*a*	100
		b	*b*	001
		b	*a*	011
		b	*e*	110
		b	*f*	100
		a	*e*	001
		a	*f*	010
		e	*f*	001
Music clip	*babb*	*b*	*a*	001
		b	*b*	111
		a	*b*	011

For example, in Fig. 8, the pointer for the symbol "*b*" at the root points to a child node, in which the *pattern relation* set for "*a*" contains a binary code of 010 for song 1 and a binary code of 011 for song 2. The binary code just mentioned for song 2 means that there are *subsequences* "*ba*" and "$b*a$", but not "$b**a$", of the *SAX representation* for the song. In this example, each asterisk can be replaced with any symbol.

When searching for a song with a music clip, the whole *SAX representation* of each *MFCCs* feature from the music clip is parsed into a set of *pattern relations*, which is then compared with the respective *AFPI* tree. Table 1 shows two examples of a *pattern relation* set for a *SAX representation*.

After a feature from the music clip is parsed into a *pattern relation* set, it is compared with all *pattern relation* sets stored in the *AFPI* tree. This step obtains a partial score for the *similarity* of the music clip to each song in the database. *Pattern relation* sets can be compared in the following two steps:

(1) Find the *intersection* of the two sets of (S_i, S_j) from each *pattern relation* set.
(2) For every (S_i, S_j) in the *intersection* set, find the number of common bits in the binary codes of the two *pattern relation* sets. This step is illustrated in Table 2 as an example.

Table 2 shows the common bits of the binary codes from song 2 and the music clip in the example in Table 1. Since the summation of the number of common bits is $1 + 2 = 3$, the partial score given by the *AFPI* tree is thus 3 points.

B) Entropy and the mechanism for merging the 51 partial scores given by each AFPI tree

The concept of information *entropy* was first introduced by Claude E. Shannon in 1948 [12]. Information Entropy Methods are still widely used in many computer science fields and an *entropy* method was applied in this study.

The information entropy [12] $H(X)$ of a discrete random variable X can be calculated by the following formula:

$$H(X) = \sum_i P(x_i) \log_b P(x_i), \tag{7}$$

where x_i denotes each possible event of X, and $P(x_i)$ is the probability that the event x_i occurs, while b is the logarithmic base used, usually set to 2 for binary logarithm.

In the proposed system, entropy in (7) is applied for the *SAX* representation sequence $\{a_1, a_2, \ldots, a_n\}$. We set x_i as each distinct symbol that can be found in the sequence, and $P(x_i)$ represents the probability that a_j equals x_i, where j is a uniformly distributed random variable of an integer between 1 and n (inclusive).

C) Mechanism for merging the 51 partial scores given by each AFPI tree

To search for a music clip, each of the 51 *AFPI* trees gives a particular partial score for the *similarity* to the music clip, and the 51 scores are combined. Next, the combined score is the "final score", which is then ranked as the final ranking result.

Table 2. The common bits.

		Binary codes			
S_i	S_j	Song 2	Music clip	Common bits*†	Number of common bits
b	*b*	001	111	No, No, Yes	1
b	*a*	011	001	Yes, No, Yes	2

*Common bits and the two binary codes must be in the same position.
†"Yes" and "No" are marked in the order of the content of the two binary codes, where "Yes" means that a pair of the common bits at the specified position exists.

The final score is obtained by calculating the weighted summation of all partial scores with the corresponding entropy, which can be evaluated as the following equation:

$$R_{final} = \sum_{k=1}^{13} R_k \cdot H(X_k), \tag{8}$$

where R_k is the partial score given with respect to the kth *MFCC*s feature, and $H(X_k)$ represents the entropy of the random variable X_k from the *SAX* representation sequence of the search key in the kth *MFCC*s feature.

D) Noise suppression based on AKLT

Walter and Zhang proposed [21] that, for N random vectors $\{\mathbf{x}_n \in R^d | n = 1, 2, \ldots, N\}$, the following steps can find the approximate *KL* basis. First, expand N vectors into complete wavelet packet coefficients. Then calculate the variance at each node and search this variance tree for the best basis. Sort the best basis vector in decreasing order, and select the top m best basis vectors to form a matrix $\mathbf{U}$. Finally, transform N random vectors using the matrix $\mathbf{U}$ and diagonalize the covariance matrix $\mathbf{R}_N$ of these vectors to obtain the eigenvectors. Since we expect $m \ll d$, the reduction in computational load must be considered. Figure 9 illustrates the flowchart.

A linear estimator is used for noise reduction [19]. Let $\mathbf{z}$, $\mathbf{y}$, and $\mathbf{w}$ be K-dimensional vectors denoting noisy speech, clean speech, and noise, respectively. Transform $\mathbf{z}$ and $\mathbf{w}$ into wavelet packet domain. Then, calculate $\mathbf{z}$ and $\mathbf{w}$ to build the variance trees T_z and T_w, respectively. Because clean speech and noises are independent, subtract T_w from T_z node-by-node, and calculate the eigen-decomposition of clean speech by transforming the subtraction of the variance tree with *AKLT* [7].

Let $\tilde{\mathbf{R}}_y = \mathbf{U}_y \mathbf{\Lambda}_y \mathbf{U}_y^H$ denote the eigen-decomposition. Finally, let M be the number of eigenvalue of $\tilde{\mathbf{R}}_y$ greater than zero. Let $\mathbf{U}_y = [\mathbf{U}_1, \mathbf{U}_2]$, where $\mathbf{U}_1$ donates the $K \times M$ matrix of eigenvectors with positive eigenvalues,

$$\mathbf{U}_1 = \left\{ u_{yk} \middle| \lambda_y(k) > 0 \right\}. \tag{9}$$

Let $\mathbf{z}^T = \mathbf{U}_y^H \mathbf{z} = \mathbf{U}_y^H \mathbf{y} + \mathbf{U}_y^H \mathbf{w} = \mathbf{y}^T + \mathbf{w}^T$. The covariance matrix $\mathbf{R}_{w^T}$ of $\mathbf{w}^T$ is $\mathbf{U}_y^H \mathbf{R}_w \mathbf{U}_y$. Let $\sigma_{wT}^2(k)$ be the kth

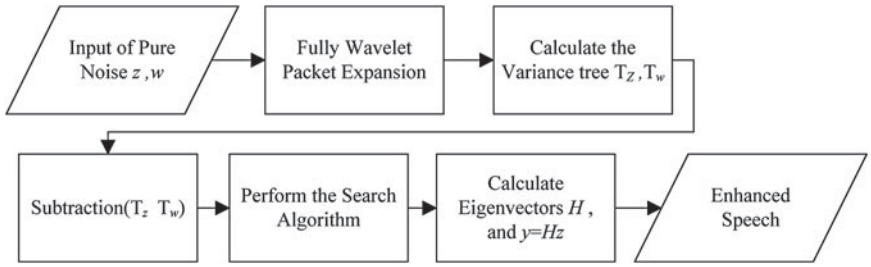

Fig. 9. The flowchart of *AKLT*.

diagonal element of $\mathbf{R}_{w_T}$. The obtained estimate of $\tilde{\mathbf{y}}$ is

$$\tilde{\mathbf{y}} = \mathbf{Hz}, \mathbf{H} = \mathbf{U}_y \mathbf{Q} \mathbf{U}_y^H, \tag{10}$$

where $\mathbf{Q}$ is a diagonal matrix.

$$\mathbf{Q} = diag\left(q_{kk}\right), q_{kk} = \begin{cases} a_k^{1/2}, & k = 1, 2, \ldots, M \\ 0, & \text{otherwise,} \end{cases} \tag{11}$$

and

$$a_k = \begin{cases} \exp\left(\dfrac{-v\sigma_{wT}^2(k)}{\lambda_y(k)}\right), & k = 1, 2, \ldots, M \\ 0, & \text{otherwise,} \end{cases} \tag{12}$$

where v (=0.5) is a predetermined constant.

The assumptions are that the noise is stationary and that the pure noise vector $\mathbf{w}$ is known. The pure noise vector $\mathbf{w}$ can be obtained from the noise only frame.

VI. EVALUATING THE PERFOMANCE BY EXPERIMENTS

In this section, we evaluate the performance in accuracy and efficiency of the proposed *QBS-CBMR* system.

A) Experimental data and measures

The database consists of 200 songs of various languages: 44 in Chinese, 51 in English, and 105 in Japanese. A total of 110 songs of the database are sung by male artists, 62 are by female artists, and the remaining 28 is a choral. In all, 67 different artists are involved and the songs are of various tempos and lengths so that the database is as diversified as possible. All songs in our database are sampled in 22 050 Hz and 8 bits-per-sample.

The proposed system is implemented with *Java SE7*, and the experiments are run on *Windows 8.1*, with a 3.4 GHz $i7-3770$ CPU, and a RAM of 4 GB.

The performance was measured in terms of *Accuracy*, which was calculated as follows:

$$Accuracy = \frac{\sum_{i=0}^{N} R(i)}{N}, \tag{13}$$

where N is the total number of tests. In each test, $R(i)$ returns 1 or 0 depending on whether a given music clip can or cannot be searched in the top n ranking, respectively. Therefore, different Accuracy rankings are calculated with different n.

Table 3 shows the results obtained when using this method to compute accuracy. The first row is the name of input music clips. When $n = 2$, there are two correct results of music retrieval that can be found in all four songs. Hence, the *Accuracy* is 50%.

Twenty songs were randomly chosen from the 100 songs in the database. In the prelude song, there are usually no obvious changes, until the chorus. Therefore, accuracy is higher when using music clips from the chorus than when using music clips from the prelude song. Music clips from the chorus are used in this experiment.

Table 3. The example of how to calculate accuracy.

Rank	Mirai-Fu	Oblivion...	SAKURA	October
1	Shining	**Oblivion...**	Future	Smile
2	**Mirai-Fu**	Gather	Go on	Honey !
3	HIKARI	Enemy	Prayer	**October**
4	AWAY	Wild	**SAKURA**	Rain
5	Hi-side	Future	BIBLE	HIKARI

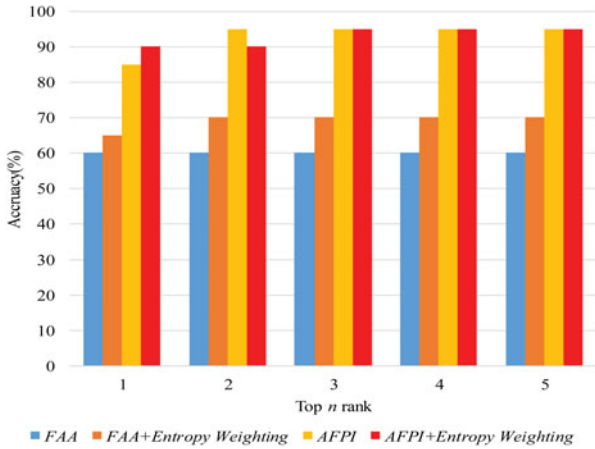

Fig. 10. Comparison of the proposed system and baseline system.

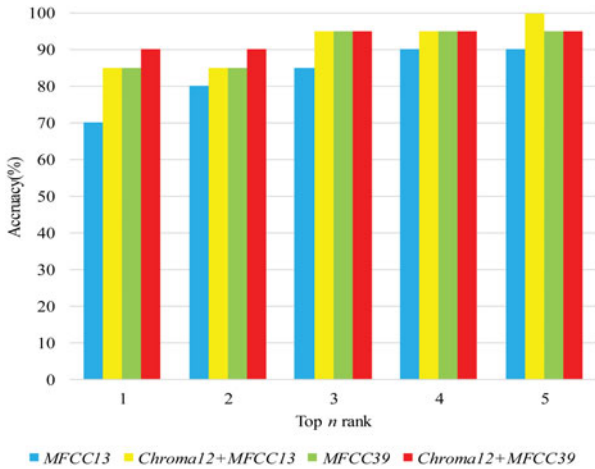

Fig. 11. Comparison between different dimensions of features.

The database contains N songs and M music clips. Each music clip is of t seconds. We test the total amount of the songs within the top n ranks among M search keys ($M \leq N$). In the following experiments, we set $N = 200$, $M = 20$, and $t = 20 \pm 1$ s.

B) Experimental results

The system, which uses the *AFPI* tree and the Fusion of *AFPI* and Alignment (FAA) tree without entropy-weighting mechanism is selected to be the baseline system [2]. Figure 10 shows the comparison of the proposed system and the baseline system. The experimental results show that the

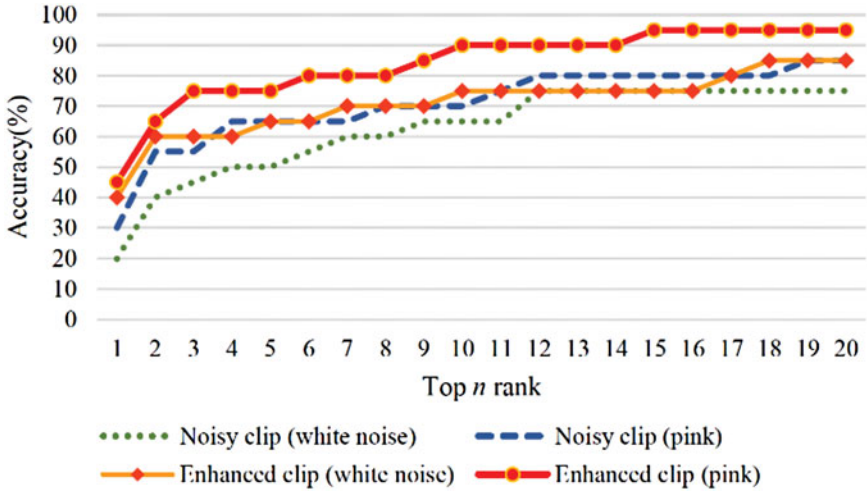

Fig. 12. Comparison between noised music clips and enhanced music clips.

proposed method with entropy-weighting outperforms the baseline system.

The first experiment shows that using *AFPI* with entropy weighting is better than using only *AFPI* or *FAA*. The proposed music retrieval system uses the *AFPI* method for the database and the entropy-weighting summation mechanism for determining the final ranking, while [2] claims that the *FAA* method performs better than using *AFPI* only, since the methods used in the music retrieval processes are different. The first experiment compared the accuracy of the methods with and without *AFPI* and *FAA* and determined whether or not entropy-weighting should be used in the summation mechanism for the final result.

The second experiment used *MFCC*s and *Chroma* as features for music retrieval. Every music clip is an interception of the chorus part in 20 s. Figure 11 shows the comparison between different dimensions of features. The result about the second experiment is that the accuracy of using *MFCC*s and *Chroma* features is obviously better than only use *MFCC*s feature.

In the third experiment, 5 db white noise and pink noise was added to each music clip, and *AKLT* was used to exclude the effect of noise. Figure 12 shows the result of noisy and enhanced music clips in music retrieval. The feature is as same as the previous experiments, *MFCC*s and *Chroma* features. It is clear to find that after using *AKLT* to enhance the music clips, the music retrieval result is better than without using *AKLT*.

VII. CONCLUSION

The robust *QBS* music retrieval system proposed in this study first converts the 51-dimensional features, which include *MFCC*s and *Chroma* features, into symbolic sequences by applying adapted *SAX* methods. The symbolic sequence is then used to construct the *AFPI* tree. Finally, the entropy-weighting mechanism is proposed to determine the final ranking. Noise effects are further reduced by applying *AKLT* preprocessing. The experimental results show that the proposed *QBS-CBMR* system outperforms the baseline system. Future studies will optimize the parameters of the proposed method such as the length of symbolic sequence and the dimension of the *PAA* representation. Moreover, a large database will be used to demonstrate the efficiency of the system.

ACKNOWLEDGEMENT

This research was supported in part by Ministry of Science and Technology, Taiwan, R.O.C., under Grant MOST 104-2218-E-126-004.

REFERENCES

[1] Chamie, J.: *World population ageing: 1950–2050*. Population Division, Department of Economic and Social Affairs, United Nations Secretariat, New York, 2005, 34–42.

[2] Huang, J.: Novel pattern indexing techniques for efficient content-based music retrieval. M.S. thesis, National Cheng Kung University, Institute of Computer Science and Information Engineering, Taiwan, R.O.C., 2008.

[3] Su, J.-H.; Huang, Y.-T.; Yeh, H.-H.; Tseng, V.-S.: Effective content-based video retrieval using pattern indexing and matching techniques. Expert Syst. Appl., 37 (7) (2010), 5068–5085.

[4] Lu, L.; You, H.; Zhang, H.-J.: A new approach to query by humming in music retrieval, in *IEEE Int. Conf. Multimedia and Expo (ICME)*, 2001, 595–598.

[5] Cui, B.; Liu, L.; Pu, C.; Shen, J.; Tan, K.-L.: QueST: querying music databases by acoustic and textual features, in *ACM Int. Conf. Multimedia*, 2007, 1055–1064.

[6] Chiang, Y.-L.; Lee, Y.-S.; Hsieh, W.-C.; Wang, J.-C.: Efficient and portable content-based music retrieval system, in *IEEE Int. Conf. Orange Technologies (ICOT)*, 2014, 153–156.

[7] Yang, C.-H.; Wang, J.-F.: Noise suppression based on approximate KLT with wavelet packet expansion, in *IEEE Int. Conf. Acoustics, Speech, and Signal Processing (ICASSP)*, 2002, 13–17.

[8] Tao, D.-C.; Liu, H.; Tang, X.-O.: K-box: a query-by-singing based music retrieval system, in *ACM Int. Conf. Multimedia*, 2004, 464–467.

[9] Tu, M.-C.; Liao, W.-K.; Chin, Y.-H.; Lin, C.-H.; Wang, J.-C.: Speech based boredom verification approach for modern education system,

in *Int. Symp. Information Technology in Medicine and Education*, 2012, 87–90.

[10] Shepard, R.N.: Circularity in judgments of relative pitch. J. Acoust. Soc. Am., 36 (12) (1964), 2346–2353.

[11] Lin, J.; Keogh, E.; Lonardi, S.; Chiu, B.: A symbolic representation of time series, with implications for streaming algorithm, in *Data Mining and Knowledge Discovery Workshop (DMKD)*, 2003, 2–11.

[12] Shannon, C.E.: A mathematical theory of communication. Bell Syst. Tech. J., 27 (3) (1948), 379–423.

[13] Davis, S.B.; Mermelstein, P.: Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences. IEEE Trans. Acoust. Speech Signal Process., 28 (4) (1980), 357–366.

[14] Tyagi, V.; Wellekens, C.: On desensitizing the Mel-cepstrum to spurious spectral components for robust speech recognition, in *IEEE Int. Conf. Acoustics, Speech, and Signal Processing (ICASSP)*, 2005, 529–532.

[15] Xiong, W.; Yu, X.-Q.; Shi, J.-H.: Music retrieval system using chroma feature and notes detection, in *IET Int. Conf. Smart and Sustainable City (ICSSC)*, 2013, 476–479.

[16] Sumi, K.; Arai, M.; Fujishima, T.; Hashimoto, S.: A music retrieval system using chroma and pitch features based on conditional random fields, in *IEEE Int. Conf. Acoustics, Speech and Signal Processing (ICASSP)*, 2012, 1997–2000.

[17] Peeters, G.: Chroma-based estimation of musical key from audio-signal analysis, in *Proc. 7th Int. Conf. Music Inf. Retrieval (ISMIR)*, 2006, 115–120.

[18] Shen, J.; Shepherd, J.; Cui, B.; Tan, K.-L.: A novel framework for efficient automated singer identification in large music databases. ACM Trans. Inf. Syst., 27 (3) (2009), 18:1–18:31.

[19] Mittal, U.; Phamdo, N.: Signal/noise KLT based approach for enhancing speech degraded by colored noise. IEEE Trans. Speech Audio Process., 8 (2) (2000), 159–167.

[20] Ephraim, Y.; Van Trees, H.L.: A signal subspace approach for speech enhancement. IEEE Trans. Speech Audio Process., 3 (4) (1995), 251–266.

[21] Walter, G.G.; Zhang, L.: Orthonormal wavelets with simple closed-form expressions. IEEE Trans. Signal Process., 46 (8) (1998), 2248–2251.

Yuan-Shan Lee received his M.S. degree in Applied Mathematics from National Dong Hwa University, Hualien, Taiwan, in 2013, and he is presently working toward the Ph.D. degree in Computer Science and Information Engineering in National Central University, Taoyuan, Taiwan. His current research interests include machine learning, blind source separation, neural network, and signal processing.

Yen-Lin Chiang received his B.S. degree in Computer Science from National Central University, Taiwan, in 2015. He is presently pursuing his M.S. degree at National Tsing Hua University, Taiwan. His general researches include machine learning and pattern recognition for multimedia processing.

Pei-Rung Lin is presently working toward the B.S. degree at National Central University, Taiwan. Her recent work has been in the areas of music retrieval and score following. Her general researches include machine learning and pattern recognition.

Chang-Hung Lin received his B.S. degree in Computer Science and Information Engineering from National Central University, Taoyuan, Taiwan. His general research interests include singing evaluation for karaoke applications, machine learning, and pattern recognition.

Tzu-Chiang Tai received his M.S. and Ph.D. degrees in Electrical Engineering from National Cheng Kung University, Tainan, Taiwan, in 1997 and 2010, respectively. He was with Philips Semiconductors and United Microelectronics Corporation (UMC) from 1999 to 2003. Presently, he is an Assistant Professor in the Department of Computer Science and Information Engineering, Providence University, Taichung, Taiwan. His research interests include signal processing, reconfigurable computing, VLSI design automation, and VLSI architecture design.