MUSIC CUT AND PASTE: A PERSONALIZED MUSICAL MEDLEY GENERATING SYSTEM

I-Ting Liu **Yin-Tzu Lin** **Ja-Ling Wu**
Graduate Institute of Networking and Multimedia, National Taiwan University
{tinaliu,known,wjl}@cmlab.csie.ntu.edu.tw

ABSTRACT

A musical medley is a piece of music that is composed of parts of existing pieces. Manually creating medley is time consuming because it is not easy to find out proper clips to put in succession and seamlessly connect them. In this work, we propose a framework for creating personalized music medleys from users' music collection. Unlike existing similar works in which only low-level features are used to select candidate clips and locate possible transition points among clips, we take song structures and song phrasing into account during medley creation. Inspired by the musical dice game, we treat the medley generation process as an audio version of musical dice game. That is, once the analysis on the songs of user collection has been done, the system is able to generate various medleys with different probabilities. This flexibility brings us the ability to create medleys according to the user-specified conditions, such as the medley structure or some must-use clips. The preliminary subjective evaluations showed that the proposed system is effective in selecting connectable clips that preserved chord progression structure. Besides, connecting the clips at phrase boundaries acquired more user preference than previous works did.

1. INTRODUCTION

A musical medley is a music piece that is composed from parts of existing pieces [22]. It often composed from famous tracks of a specific artist, year or genre. The song excerpts can be played successively with or without cross-fading. In the past, medleys are usually made by professional audio engineers and published by music production companies. Nowadays, more and more music hobbyists create their own medleys from their favourite songs with the help of newly developed audio technologies and publish the results on websites like Youtube. The resulting medley can be used as the background music of personal films and slideshows or non-stopping dance suites. Since each song track just appeared as short clips (usually less than 30 seconds) in a medley, the users can avoid copyright infringement while keeping their personal video with background music of their tastes. Existing editing tools like Goldwave [1] and Audition [2] enable users to cut and connect audio clips at the position they specify directly. However, these tools are not good enough to achieve the pre-described goal. In other words, users still face the following challenging issues: (i) how to choose suitable song excerpts to put together and (ii) how to euphoniously connect them. With the vast amount of digital music, it is not easy for users to find song excerpts that are suitable for playing together. Besides, once the user decides which song excerpts to adjoin, they still need to listen to all of them to decide the position for cutting audio files into clips and connecting them. Furthermore, users may need to manually adjust the tempi and volume levels of the clips if needed to make them smoothly connected. Some previous works provided automatic schemes to create medleys or medley-like mixes. However, they either only focused on the rhythm and beat synchronization issues of adjacent clips or on directly connecting full songs [12, 15]. User preferences of the picked songs, the phrase boundary, and the structure of the medley were rarely considered [14]. We believe that whether music sounds pleasant highly depends on each person's taste. Thus, in this work, a framework for creating personalized musical medleys from users' music collection is proposed, in which the completeness of phrases and chord progression of a song can be preserved. As shown in Figure 1, users specify the structure of the target medley, and optionally select a few song excerpts that should appear at certain positions in the medley, i.e. the darker parts in the figure. Then the system will complete the medley with song excerpts automatically selected from the song collection user provided. The excerpts, automatically segmented by our system, are about four-bars long (∼10 seconds on average.) We treat the medley generation process as an audio version of musical dice game. Various

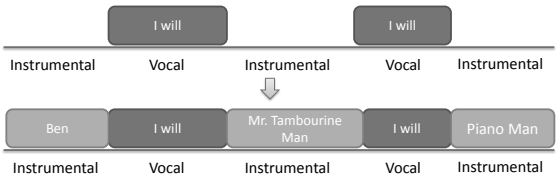

Figure 1. The scenario of the proposed system. Upper part: the user-specified structure and must-used clips. Bottom part: the generated medley by our system.

[1] http://www.goldwave.com/
[2] http://www.adobe.com/products/audition.html

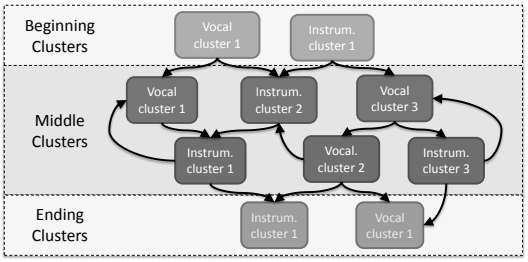

Figure 2. An example of a musical dice graph.

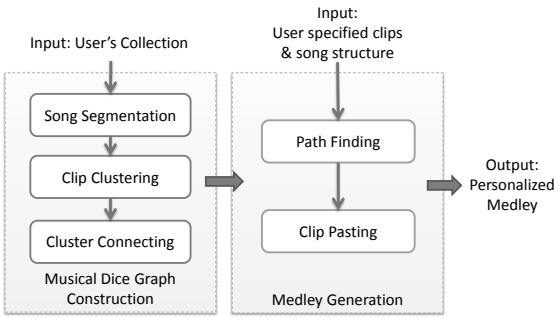

Figure 3. The proposed system framework.

medleys can be generated once we've finished analyzing user's song collection. The flexibility allows us to create medleys based on user needs. We also provide an interactive scheme for users to change selected clips, adjust connecting positions and the overlap ratios between clips if users are not satisfied with the result.

2. RELATED WORK

The problem of medley creation can be divided into two parts: (i) finding proper song excerpts and their orders (including the order of the clips), and (ii) concatenating the selected clips (including finding connecting positions). There are several previous works addressed these issues separately, but few considered them both.

The clip selection issue is similar to the song selection problem in the study of playlist generation. The general approach to these tasks is to select similar songs based on some specific criteria. Commonly used criteria usually fall into one of the following types: (i) meta-data based (e.g. same artist, album, or genre), (ii) content-based (e.g. audio feature similarity [7, 17], key, tempo [5]), and (iii) collaborative filtering based (e.g. occurrence of the songs the the user's friends' past playlists [1]). In our case, we dealt with song excerpts rather than the whole song, i.e, songs are usually interrupted instead of being played to their end. As a result, the required properties used to find adjacent clips are stricter than that of finding adjacent songs. In addition to the global similarities between songs, local audio similarities between song excerpts should also be considered to meet the listeners' expectations for the musical flow.

The works focused on clip concatenation are often found in the studies of DJ tools [3, 9]. In those works, the clips are designated by users, and the authors deal with the concatenation issue only. Some studies of DJ tools also considered both the issues of clip selection and clip concatenation [12, 15]. However, in the aforementioned works, "rhythm similarity" and "beat alignment" are emphasized most because the results are often used for dance. Since a medley is still a kind of music composition, we focus more on the chord euphoniousness of the adjacent clips in this work.

There are also some works similar to us except that they targeted at single song only [16, 25]. In these works, self-similarity matrix of audio feature sequences are used to segment songs [16] or to find cut points [25], and thus the methods cannot be directly applied to our case. The corresponding computational complexity grows dramatically as the number of songs increases. Besides, one can only find near-identical clips with self-similarity matrix. In our case, clips are selected from different songs. If we group the clips based on the same criteria as these works, the resulting medley would be composed only with excerpts taken from the same song, violating the definition of a medley. Therefore, a higher level feature – the chord sequence similarity is adopted in this work. Another similar work was Bernardes et al.'s EarGram [4], but the unit they used to combine music is much shorter than us, i.e. as short as a musical note (usually less than 1 second). Consequently, the music piece produced by EarGram will not keep the phrases of the original songs. It becomes a totally new song rather than a combination of existing songs, and the main interesting feature of medley songs – it is a combination of different songs – is lost.

The work most related to ours is Lin et al.'s "Music Paste" [14]. They dealt with both the clip selection and the clip concatenation issues in their work. However, they did not consider phrase structures while finding connecting positions among clips. Consequently, clips might be cut in the middle of a phrase, resulting in unsmooth transitions. In addition, in Lin et al.' s work, the order of the clips is determined only by matching similar short-time chroma features. It is, therefore, difficult to incorporate user's preferences into the process of clip ordering. In our work, it is easy to put users' preference into consideration, and the chord progression can be preserved better.

3. PROPOSED FRAMEWORK

The proposed framework is inspired by the musical dice game in the symbolic domain [18]. In a musical dice game, players throw dices to randomly "generate" new music from the pre-composed interchangeable musical figures [3] at each bar. Similarly, in our system, we would like to generate medleys by choosing the clip at a given position from a set of interchangeable clips. In order to achieve that, we first analyze the songs in the user-provided collection and cut the songs into self-coherent clips. Then, we group similar clips into clusters. The clips in the same cluster are musically similar to each other, and thus are assumed to be interchangeable. We then connect clusters according to the transition probability calculated from clips' connectivities in the songs they are extracted from. We term it

[3] A short musical phrase [22].

the "musical dice graph" in the rest of this paper. A path on the graph is a version of a medley. Figure 2 shows an example of a musical dice graph. With this graph, we can generate numerous medleys based on user's preferences and the transition probability. Then, the aforementioned two issues in medley creation, "clip selection" and "clip concatenation", can now be turned into "musical dice graph construction" and "medley generation from the walk on the graph", respectively. Figure 3 illustrates the proposed framework. We will discuss the details in the following sections.

4. MUSICAL DICE GRAPH CONSTRUCTION

We divide the construction of the musical dice graph into three steps: song segmentation, clip clustering, and transition probability calculation.

4.1 Song Segmentation

The goal of song segmentation is to divide songs into self-coherent segments. According to [24], phrasing is one of the most important factors to be considered when concatenating different tracks. Interruption happened in the middle of a musical phrase is just as unpleasant and unexpected as that of an oral sentence in a conversation. Therefore, the transition between clips should occur at the boundary of musical phrases. The length of a musical phrase is ambiguous. Here, we define a musical phrase as a self-coherent clip that sounds like ending, usually four or eight-bars long because pop songs usually take a thirty-two-bar form [4] . In order to identify the appropriate timing of transition, i.e., the boundary of musical phrases, we perform singing voice detection on each song track. The reason is that most medleys are composed of pop songs and that the boundary of each singing voice section often corresponds to that of a musical phrase. As a result, we cut the songs into clips based on the boundary of detected vocal segments, and use them as the basic unit for creating medleys.

4.1.1 Singing voice detection

The goal of singing voice detection is to recognize a given audio segment as vocal or instrumental. An instrumental segment is defined as a segment consisting of purely instrumental sounds, e.g. the bridges and the intro. A vocal segment, on the other hand, is defined as a mixture of singing voice with or without background music, as was defined in [21]. Many studies have been done in the past focusing on the task of singing voice detection. The typical procedure performed to solve this problem is to extract short-time audio features, and train a two-class classifier to classify each frame of the audio into instrumental or vocal class. Common classifiers used are Gaussian Mixture Models (GMM), Hidden Markov Models (HMM) and their variants, and Support Vector Machines (SVM). Frequently used features include Mel Frequency Cepstral Coefficients (MFCC), Linear Prediction Coefficients (LPC), and Zero-Crossing Rate (ZCR) [19]. Often, temporal smoothing techniques are also performed to constrain the length of each vocal and instrumental segment afterwards to prevent over-segmentation [23]. Here, we employ beat-synchronized MFCC as audio features and HMM as the classifier. The idea of using beat-synchronized features comes from the phenomenon that voices are more likely to join the accompaniment at beat times [13]. MFCCs are first extracted during the training phase. The MFCCs within a musical beat act as the observed sequence for an HMM classifier. The beats are detected by using BeatRoot [6], a state-of-the-art beat-detection tool. In the testing phase, each beat in a test song can then be classified as vocal or instrumental. Consecutive vocal/instrumental beats can then be group into vocal/instrument clips, i.e. the unit we used to create medleys. In addition, in order to avoid over-segmentation, we apply a median filter of 8 beats long (i.e. about 4 seconds) to the singing voice detection result.

[4] also known as AABA form [22]

4.2 Clip Clustering

After dividing songs into clips, we group these clips according to the similarities among them. There are numerous metrics for measuring similarity among clips, e.g. volume, rhythm, timbre, tonality, genre, etc.. Here we choose the chord sequence similarity because in the musical dice game, the candidates of musical figures are often chord-similar or dominant-pitch-similar. The clip clustering process can be divided into three steps: chord detection, chord sequence similarity computation and clustering.

4.2.1 Chord detection

First, we detect the chords of all songs with Harmony Progression Analyzer (HPA) [20], a state-of-the-art chord estimation system. The number of possible chords estimated is limited to 25, which are 12 major chords, 12 minor chords, and no-chord for the periods of silence. Then, beat-synchronized chord sequences are extracted for each clip by aligning the chord estimation results with the detected beats of the clips.

4.2.2 Chord Sequence Similarity Computation

We then measure the chord sequence similarity between each clip pairs. A dynamic-time-warping (DTW)-like approach proposed by Hanna et al. [11] is used in our system to measure the edit distance between two given chord sequences. In order to better capture the harmonic relationship between the sequences, the substitution score used to calculate the edit distance varies with the consonance of the interval between the two given chord roots, as Hanna et al. [10] proposed. Consonant intervals are the intervals that sound stable [22], and chords whose roots form a consonance are given higher substitution scores.

4.2.3 Clustering

Given the similarity score between each pair of the chord sequences, we could then cluster the clips by hierarchical clustering. The vocal clips and the instrumental clips are clustered separately. Also, the clusters are categorized into three types according to the positions of the clips in the

song. In other words, we would have six types of clusters in total: beginning, ending and middle clusters, each of which can be either vocal or instrumental. Note that each cluster type also contains several clusters.

4.3 Cluster Connecting

Then, we connect the clusters according to the transition probability defined as in Equation (1). For two arbitrary clusters A and B, the transition probability $P(B|A)$ is defined as the proportion of clips in cluster A that originally concatenate with clips in cluster B, that is:

$$P(B|A) = \frac{|S|}{|A|},\ S = \{(a,b)|a \in A, b \in [B \cap N(a)]\} \tag{1}$$

where a and b stands for an arbitrary clip, and $N(a)$ is the set of clips that appeared next to clip a in the original song.

5. MEDLEY GENERATION

After the musical dice graph is constructed, we can then compose medleys by finding a path on the graph and concatenating them according to the path.

5.1 Path Finding

Now we find a path on the musical dice graph with the maximum transition probability along the path. First, we picked candidate clusters according to user-specified medley structure. For example, the user may designate the structure as I→V→I→V→I, where "I" and "V" stand for instrumental and vocal clips, respectively. Then, we choose clusters conforming to the types the user specified in the structure as candidate clusters. That is, instrumental-beginning clusters for the first clip slot, and vocal-middle clusters for the second, and so forth, for the previous example. If the user has specified the third slot to be clip a, then the cluster that clip a belongs to is chosen. Then, we use Viterbi algorithm [8] to find a path of candidate clusters with maximal transition probability, where the candidate clusters at each slot are regarded as the states in the algorithm. After determining the clusters, we randomly select one clip per cluster since clips in the same cluster are interchangeable. The selected clips then compose the final medley.

5.2 Clip Pasting

Once the clips are selected along the chosen path on the graph, we then concatenate them smoothly. For each selected clip, we adjust the tempi with Lin et al.'s method [14]. The algorithm takes just noticeable difference (JND) between the tempi of two clips into account, adjusting the tempi gradually to lessen listeners' discomfort. After adjusting the tempi, the volumes of the clips are then normalized, and the clips are finally concatenated by using a short-length logarithmic cross-fade in between.

6. EXPERIMENT

In this section, we conduct three experiments, two subjective and one objective, to evaluate the three components of the system separately: the performance of the song segmentation method and the clip selection mechanism, and the quality of the transition. The experimental data set contains 100 best-selling English songs from 1950s to 1990s collected from Youtube [5] , and the genres include folk, pop, jazz, musical and movie soundtrack, the lengths of which range from one and a half minutes to five and a half minutes. Two sets of annotation are built manually for each track: the singing voice annotation and the musical phrase annotation. The former was used to perform singing voice detection, while the latter is used as the ground truth of boundary detection task. All songs in this dataset contain singing and instrumental parts, i.e. none of them is a pure instrumental nor A cappella music.

Method	G-to-T	T-to-G	Pre.	Rec.	F-meas.
GT	0.00	0.00	0.73	0.78	0.75
GMM	2.85	10.29	0.20	0.17	0.18
HMM	2.55	**2.66**	**0.25**	**0.30**	**0.27**
DTM	**2.38**	5.13	0.23	0.18	0.20

Table 1. The song segmentation result.

6.1 Effectiveness of Song Segmentation

This experiment measures the effectiveness of using singing voice detection techniques to detect phase boundaries. We compare three kinds of segmentation scheme: singing voice detection with GMM and HMM classifier (which will be referred to as GMM and HMM later in this paper), and Dynamic Texture Model (DTM), a state-of-the-art song segmentation method proposed by Barrington et al. [2]. We evaluate the boundary detection performance by precision, recall and F-measure, where a detected boundary is regarded as a "hit" if its time interval between the nearest true boundary lies within a certain threshold. We also calculate two median time values: guess-to-true (G-to-T) and true-to-guess (T-to-G), which are the median time interval between each detected boundary and the closest true boundary and vice versa, as defined in [2]. For singing voice detection, all audio files are 22050 Hz-sampled, and 26 MFCCs are extracted every 256 samples, overlapping by half. Singing voice detection by GMM and HMM were performed respectively with 5-fold cross validation, and the number of mixtures of GMM and the number of hidden states of HMM are both empirically set to 5. The hit time interval was set to 0.5 second as in [2].

The results are presented in Table 1. The song segmentation performances using ground-truth singing voice annotation are also listed as a reference, denoted as GT. From Table 1, HMM and DTM outperform GMM obviously, while HMM performs approximately as good as DTM. In this work, we chose HMM over DTM for its simplicity and speed during the singing voice detection testing phase. Different parameters, such as the length of the median filter used to prevent over-segmentation, may influence the performance, and therefore introduce a trade-off

[5] Song names and URLs are listed at: `http://goo.gl/khjtB`.

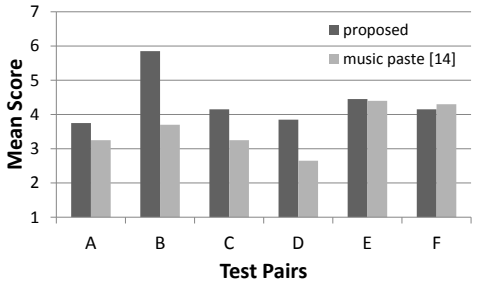

Figure 4. The user evaluation results of clip concatenation.

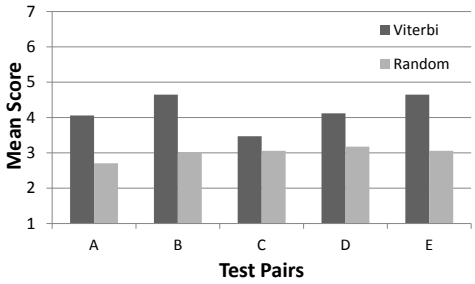

Figure 5. The results of user evaluation on clip selection.

between precision and recall. In this work, the goal of segmentation is to avoid interruption in the middle of a phrase when pasting songs. Thus, when choosing between different parameters, we would favour those results in higher precision.

6.2 Effectiveness of Clip Concatenation

To see whether segmentation really helped on song concatenation, we compare 6 pairs of medleys. Each pair contains two medleys, and each medley is composed of two song clips. The two song clips used in the medleys of one test pair are the same. For one medley in each pair, the transition of clips happens immediately and directly at the end of a phrase of the first clip (the proposed method). For the other medley, the transition point and cross-fade length are decided with Lin et al. [14]'s method, which is the location where the short-term chroma features of two clips matched the best and are in the middle of a phrase rather than the boundary. In this and the following experiments, we used human-labeled annotation for musical phrase information because we wish the results wouldn't be biased by inaccurate segmentations. The generated medleys used in the experiments can be found in `http://goo.gl/khjtB`.

We ask 20 users to listen to and compare these test pairs. Each pair of medleys are played twice. Users are asked to report how these two medleys sound according to transition smoothness between two adjacent clips. All questions are designed on a 7-point Likert scale. Figure 4 shows the score of each test pair. In 5 out of 6 pairs, the mean score of the medleys generated by the proposed method are higher than the one created by Lin et al.'s method.

6.3 Effectiveness of Clip Selection

This experiment is designed to measure the effectiveness of selecting clips based on their chord sequence transition probability. The test set for this experiment contains 5 pairs of medley. The number of transitions and the structure of the two medleys in a pair are the same, and the beginning and the ending clips of the medleys are specified to be the same. For one medley in each pair, the chosen clips and their order are decided by the Viterbi algorithm along with the calculated transition probability. For the other medley, clips are selected randomly.

We asked 17 participants to listen to and evaluate generated medleys. Each pair of medleys are played twice. Figure 5 illustrates the score of each test pair. We performed

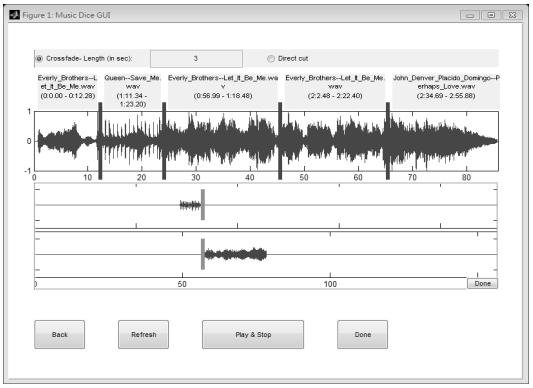

Figure 6. A screenshot of the proposed system.

pairwise t-test to analyze the results. Overall speaking, the mean score of medleys composed with Viterbi algorithm is significantly higher than the one composed with random clip selection ($p < 0.05$). If the 5 test sets are evaluated separately, the mean score of medleys composed with Viterbi is significantly higher than randomly generated one in 4 out of 5 test pairs (Pair A, B, D, and E, $p < 0.05$). The result shows that the proposed clip selecting scheme is effective in selecting clips that sound pleasant when they are concatenated and played in sequence.

7. USER INTERFACE

The quality of the generated medley is highly subjective and depends on users' tastes. For example, the actual boundary of a vocal phrase that fades out gradually is hard to determine and there is no correct answer. The cross-fade length between two clips is also subjective given that some may prefer longer overlaps because of its smoothness, while others may prefer shorter or no cross-fade to avoid blur of sounds. In order to better satisfy different user needs, we developed a simple GUI, as shown in Figure 6. With the interface, users could specify parameters, modify the segmentation result and change the selected clips.

8. CONCLUSION AND FUTURE WORK

In this work, we propose a framework for creating personalized music medleys from users' music collection. Borrowing the idea of musical dice, we construct a music dice graph with self-coherent clips that are segmented by singing voice detection. After graph construction, the system is able to generate numerous medleys, automatically. The completeness of phrases in a song is retained, and harmonic contents

are considered to improve the quality of the generated medley. Since the quality of a medley is highly dependent on listeners' tastes, we also provide a simple feedback mechanism that enables users to specify song clips, modify the transition positions and the length of cross-fades to better fit their preference. The objective experimental results demonstrated that song segmentation performed with singing voice detection is comparable with state-of-the-art methods. In addition, the preliminary subjective evaluations show that connecting clips at phrase boundary acquired more user preference than previous works did. We also demonstrated the proposed Viterbi-based algorithm is effective in selecting harmonious clips.

Many aspects of our system can be improved. First, more clip similarity measures can be introduced during clip clustering, e.g. rhythm, volume, genre, mood, etc.. Second, the number of cluster types can be extend into , for example, "first half verse", "second half chorus", "bridge", etc.. Third, song segmentation with singing voice detection restricted this work to vocal songs. In the future, we will delve into other music segmentation methods that are able to recognize phrases in instrumental music. Finally, automatic separation of background music from foreground singing voice may enable us to create and add intermediate bridges, allowing more flexibility in medley generation.

9. ACKNOWLEDGMENTS

Special thank to Professor Jyh-Shing Roger Jang for constructive discussion with us, and the course members of MSAR for helping annotate the songs.

10. REFERENCES

[1] C. Baccigalupo and E. Plaza: "Case-based Sequential Ordering of Songs for Playlist Recommendation," *LNCS*, vol. 4106, pp. 286–300, 2006.

[2] L. Barrington, *et al.*: "Modeling Music as a Dynamic Texture," *IEEE Trans. ASLP*, vol. 18, no. 3, pp. 602–612, 2010.

[3] S. Basu: "Mixing with Mozart," *Proc. ICMC*, 2004.

[4] G. Bernardes, *et al.*: "EarGram : an Application for Interactive Exploration of Large Databases of Audio Snippets for Creative Purposes," *Proc. CMMR*, June, pp. 19–22, 2012.

[5] L. Chiarandini, *et al.*: "A System for Dynamic Playlist Generation Driven by Multimodal Control Signals and Descriptors," *Proc. MMSP*, 2011.

[6] S. Dixon: "Evaluation of the Audio Beat Tracking System BeatRoot," *J. New Music Res.*, vol. 36, no. 1, pp. 39–50, 2007.

[7] A. Flexer, *et al.*: "Playlist Generation Using Start and End Songs," *Proc. ISMIR*, pp. 2–7, 2008.

[8] J. G. D. Forney: "The Viterbi Algorithm," *Proc. of the IEEE*, vol. 61, no. 3, pp. 302–309, 1973.

[9] G. Griffin, *et al.*: "Beat-Sync-Mash-Coder: a Web Application for Real-Time Creation of Beat-Synchronous Music Mashups," *Proc. ICASSP*, pp. 2–5, 2010.

[10] P. Hanna, *et al.*: "On Optimizing the Editing Algorithms for Evaluating Similarity Between Monophonic Musical Sequences," *J. New Music Res.*, vol. 36, no. 4, pp. 267–279, 2007.

[11] P. Hanna, *et al.*: "An Alignment Based System for Chord Sequence Retrieval," *Proc. JCDL*, p. 101, 2009.

[12] H. Ishizaki, *et al.*: "Full-Automatic DJ Mixing System with Optimal Tempo Adjustment based on Measurement Function of User Discomfort," *Proc. of ISMIR*, pp. 135–140, 2009.

[13] Y. Li and D. Wang: "Separation of Singing Voice From Music Accompaniment for Monaural Recordings," *IEEE Trans. ASLP*, vol. 15, no. 4, pp. 1475–1487, 2007.

[14] H.-Y. Lin, *et al.*: "Music Paste: Concatenating Music Clips Based on Chroma and Rhythm Features," *Proc. ISMIR*, Kobe, 2009.

[15] Q. Lin, *et al.*: "Music Rhythm Characterization with Application to Workout-Mix Generation," *Proc. ICASSP*, pp. 69–72, 2010.

[16] Z. Liu, *et al.*: "Adaptive Music Resizing with Stretching, Cropping and Insertion," *Multimedia Syst.*, 2012.

[17] B. Logan: "Content-based Playlist Generation: Exploratory Experiments," *Proc. ISMIR*, pp. 2–3, 2002.

[18] G. Loy: *Musimathics*, vol. 1, pp. 295–296,347–350, The MIT Press, USA, 2006.

[19] N. C. Maddage, *et al.*: "Content-based Music Structure Analysis with Applications to Music Semantics Understanding," *Proc. ACM MM*, p. 112, 2004.

[20] Y. Ni, *et al.*: "An End-to-End Machine Learning System for Harmonic Analysis of Music," *IEEE Trans. ASLP*, vol. 20, no. 6, pp. 1771–1783, 2012.

[21] T. L. Nwe, *et al.*: "Singing Voice Detection in Popular Music," *Proc. ACM MM*, p. 324, 2004.

[22] D. M. Randel: *The Harvard Dictionary of Music*, Belknap Press, 2003.

[23] L. Regnier and G. Peeters: "Singing Voice Detection in Music Tracks Using Direct Voice Vibrato Detection," *Proc. ICASSP*, pp. 1685–1688, 2009.

[24] S. Webber: *DJ Skills: The Essential Guide to Mixing and Scratching*, Focal Press, 2007.

[25] S. Wenger and M. Magnor: "Constrained Example-based Audio Synthesis," *Proc. ICME*, 2011.