A Dynamic Programming Approach to Segmenting DJ Mixed Music Shows

Tim Scarfe, Wouter M. Koolen and Yuri Kalnishkan

Computer Learning Research Centre and Department of Computer Science,
Royal Holloway, University of London, Egham, Surrey, TW20 0EX, United Kingdom
{tim,wouter,yura}@cs.rhul.ac.uk

Abstract. In this paper we describe a algorithm to segment DJ-mixed electronic dance music podcasts/radio shows into their respective tracks. In other words; we predict the song boundaries that a human domain expert would otherwise have to identify manually to perform this task.
Dance music podcasts comprise of tracks which are mixed together by the DJ. The purpose of this is to obfuscate the boundaries such that one can not tell when (two adjacent songs) song A ends and song B begins. Indeed; they may overlap by some significant margin.
Our technique involves transforming the podcast into a sequence of feature vectors each representing a small interval of time. The features are based on the fast fourier transform. Eventually we define a matrix showing the cost of a potential track starting at a given time with a given length.
Using a dynamic programming approach; we find the lowest total cost segmentation of this matrix.
We tested the algorithm on several episodes from three different (very popular) radio shows; A State of Trance with *Armin van Buuren*, Magic Island with *Roger Shah*, and Trance around the World with *Above and Beyond*.

1 Introduction

DJ mixes are an integral component of electronic dance music. Whether you listen to your favourite DJ on the radio, live or downloaded from the internet; the individual tracks will be mixed together. Indeed, mixing is the *modus operandi* in electronic music. Tracks are designed to be mixed together and even have parsimonious begin/endpoints to ameliorate the process.

More recently, podcasts (through Apple iTunes) have become the de facto standard empowering people to download radio shows for offline listening automatically. Users can play the mixes on their computers and/or mobile devices. Many shows are also broadcast on internet radio stations such as Digitally Imported and syndicated world wide on terrestrial radio.

One of the interesting features of audio is that you *can not scrub through it and get a complete picture at once in the same way you could with video.* Audio must be assimilated and experienced in real time to understand the context.

The technique we used in this paper is to segment the audio file into discrete time intervals (called *tiles*), and perform a transformation on those tiles into a domain where two tiles from the same track would be distinguishable by their cosine after normalisation. Our features are based on the Fourier transform. Intuitively speaking the Fourier features can be thought of as useful for distinguishing instruments (after filtering to remove noise and accentuate harmonics).

Because the features were noisy, we filtered them using a one dimensional convolution first-derivative Gaussian filter. In the following sections we will describe; the data set (Section 2), the evaluation criteria (Section 3), the test set (Section 4), how we pre-process the data (Section 5), the feature extraction (Section 6), the cost matrix we generate from the features (Section 7), computing the best cost segmentation from the cost matrix (Section 8) and finally our methodology and results (Section 9).

1.1 How do DJs mix the tracks together?

To mix tracks DJs always match the speed or *BPM* (beats per minute) of each adjacent track during a transition and align the major percussive elements in the time domain. This is the central component of removing any dissonance from overlapping tracks. If DJs want to bring in some faster tracks, they will slowly increase the speed during the course of one track or more tracks.

To make things sound literally more harmonious they often also apply spectral transformations (EQ) or mix adjacent tracks which are harmonically compatible to stop any dissonance between instruments.

DJs will know which "key" each track is in, and have access to a chart showing which other keys are harmonically compatible. This ensures that when both tracks are in, the tracks are harmonically congruent (each critical band of hearing in the human ear is only stimulated by one peak in the frequency domain avoiding a characteristic muddy sound being perceived). Many real instruments have a harmonic series of peaks in the frequency domain, usually with each harmonic at integer multiple of their fundamental frequency. Synthesized instruments tend to mimic this spectral pattern (which is called *timbre* in music).

1.2 Related Work

A similar problem is addressed using dynamic programming in [**?**] and [**?**]. The novelty of our approach is that it can be used in the unsupervised setting to find an entirely distinct sequence of clusters, it assumes one cluster will not appear again later on. All related work in this area is concerned with *summarisation* by which we mean inferring some kind of structure concerned with repeating elements from the audio. J. Foote et al ([**?**] [**?**]) has done a significant amount of work in this area. He has made use of similarity matrices for the purpose of structural analysis. His approach is based on statistical segment clustering.

A distinguishing feature of our approach is that we evaluate how well we are doing in respect of humans for the same task. We compare our error to the relative error of cue sheets created by human domain experts. Another important

feature of our work is that we focus directly on DJ mixed electronic dance music. We do not know of any previous work in this area.

2 Data Set

We have been supplied with several broadcasts from three popular radio shows. These are: *Magic Island, by Roger Shah* (208 shows); *A State of Trance with Armin Van Buuren (*110 *shows)*; and *Trance Around The World with Above and Beyond (*99 *shows).*

These shows are a mix of Progressive Trance, Uplifting Trance and Tech Trance in equal measures. Progressive trance is characterised as having a house riff (bass line) with melodic instrumental elements over the top. Tech trance has less or no riff and a more rhythmic focus achieved with percussion and instruments and less or no melody. Uplifting trance is faster and more melodic.

The music remains constant throughout after the introduction. All shows have an introduction track which is removed before processing (this would be a lot of speech). The shows come in 44100 samples per second, 16 bit stereo MP3 files sampled at 192Kbs or more recently 256Kbs. We will resample these to 4000Hz 16 bit mono (left+right channel) WAV files to allow us to process them faster. We have used the SoX [**?**] (Sound eXchange) program to do this.

These shows are all 2 hours long. See Figure 1 for a histogram showing the lengths of the tracks on the three shows.

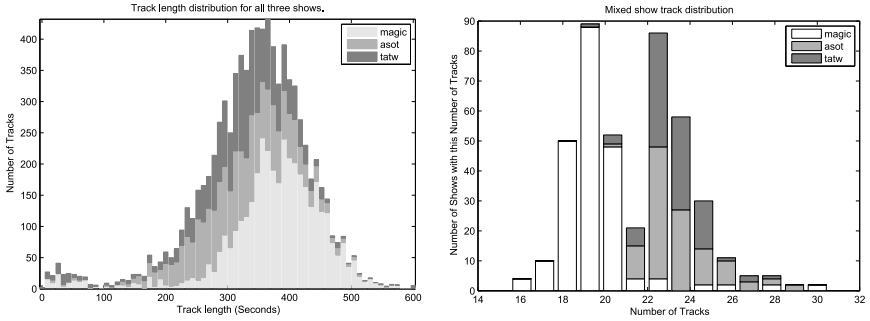


Fig. 1. The track length distribution is shown on the left. The *bump* of short tracks (less than 2 minutes) are assumed to be spoken introductions and were therefore removed. This removal threshold is a parameter in our algorithm (see Section 5). The average track length with these introduction tracks removed is roughly 5 and a half minutes. The right illustration shows how many tracks are in the shows. All the shows are 2 hours long, so the Magic Island show plays the tracks for longer before mixing them out (usually tracks do not get played in their entirety).

3 Evaluation Criteria

We perform two types of evaluation: average track accuracy (in seconds) and a measure of precision. The precision metric is the percentage of predicted indexes within monotonically increasing time thresholds from the actual index (see below). Note that the minimum track length parameters will stop any instances of multiple predicted indexes being placed within one accuracy threshold. The general use case of this algorithm is when someone has recorded a show, downloaded a track list and wanted to get the indexes. So for this reason the order of the indexes predicted is critical. The algorithm finds a fixed number of tracks, which is a parameter. The average track accuracy is calculated with the mean of the absolute differences between predicted tracks and actual tracks.

We measured the number of matched tracks within different intervals of time $\{60, 30, 20, 10, 5, 3, 1\}$ (in *seconds*) as a margin around any of the track indexes we have in the cue sheets.

Perhaps we should have some mathematical notation for the evaluation criteria but it's so simple do we even bother?

4 Test Set (Cue Sheets)

A cue sheet is a metadata file which describes track names and their respective indices for a given media file. Cue sheets are stored as plain text files and commonly have a ".cue" file name extension. Most popular media players support cue sheets, allowing the user to listen to the separate components of one large audio file.

There is already a large community of people very interested in getting track metadata for DJ sets. CueNation ([**?**]) is an example of this. CueNation is a website allowing people to submit cue sheets for popular DJ Mixes and radio shows. We had our cue sheets and radio shows provided to us by *Dennis Goncharov*; a domain expert and one of the principal contributors to CueNation. As a result of this configuration; we can assume the alignment between the cue sheet and the radio show recording is exact.

4.1 Human Capture Process

Dennis Goncharov provided us with a description of how he captured the indices in the cuesheets:

> The transition length is usually in factors of 8 bars (1 bar is 4 beats. At 135 beats per minute, 8 bars is 14.2 sec). It is a matter of personal preference which point of the transition to call the index. My preference is to consider the index to be the point at which the second track becomes the focus of attention and the first track is sent to the background. Most of the time the index is the point at which the bass line (400Hz and lower) of the previous track is cut and the bass line of the second track is introduced. If the DJ decides to exchange the adjacent tracks gradually over the time instead of mixing them abruptly then it is up

to the cuesheet maker to listen further into the second track noting the musical qualities of both tracks and then go back and choose at which point the second track actually becomes the focus of attention.

5 Data Preprocessing

We remove any tracks from the cuesheet that are shorter than duration C. This is a parameter of our algorithm. The reason for this is that these are usually not tracks at all but rather some kind of spoken introduction over the top of a track. Leaving these in has deleterious effects on our performance.

We always remove the first track from the cuesheet, but this time also the corresponding audio. These are show introductions and can be thought of as entirely independent of the tracks in the radio show. We are currently using the cuesheet to get the start time of the first track. We could estimate this reliably, but we will reserve this as an exercise for the future.

It is slightly cheating that we remove the first track, because we are looking at the cuesheet to see when the radio show kicks off proper. Leaving it in would cause unpredictable results, and it would be a reasonable parameter to ask a human using this algoritm to say when the first track is. In fact the first track can be estimated reasonably well depending on the show.

6 Feature Extraction

We used SoX [**?**] to downsample the shows to 4000Hz. We are not particularly interested in frequencies above around 2000Hz in the spectrum because there are not many instrument harmonics that go that high and they are not well defined even when they do. The Nyquist theorem ([**?**]) states that the highest representable frequency is half the sampling rate, so this explains our reason to use 4000Hz. We will refer to the sample rate as R. Let L be the length of the show in samples.

Next we binned/segmented the dataset again into *tiles* of equal size. Samples at the end of the show that do not fill a complete tile get discarded. We denote the tile width by T_{sec} (in seconds). This will become one of the key parameters later on. We stack the tiles on top of each other to create the matrix V of size $K \times \frac{N}{2}$, where $K = \lfloor \frac{(\frac{L}{R})}{T_{\text{sec}}} \rfloor$ is the number of tiles and N is the size of the Fourier vector (see next paragraph). Note that $T_{\text{sec}} = \frac{T_{\text{samp}}}{R}$ where T_{samp} is tile width in samples ($\lfloor \frac{L}{K} \rfloor$). As we are not always interested in the entire range of the spectrum, we use l to represent a low pass filter (in Hz) and h the high pass filter (in Hz).

Fourier analysis allows one to represent a time domain process as a set of integer oscillations of trigonometric functions. We used the discrete Fourier transform (DFT) to transform the tiles into the frequency domain. The DFT given as $F(x_k) = X_k = \sum_{n=0}^{N-1} x_n \cdot e^{-i2\pi \frac{k}{N} n}$ transforms a sequence of complex numbers $x_0, \ldots, x_N$ into another sequence of complex numbers $X_0, \ldots, X_N$ where $e^{-i2\pi \frac{k}{N} n}$ are points on the complex unit circle. Note that the fftw algorithm that we used to perform this computation (see [**?**] and [**?**]) operates significantly faster when N is a power of 2 so we zero pad the input to make that the case.

Let the matrix U of size $K \times (S_{\text{l}} - S_{\text{h}})$ represent the Fourier transforms of each tile where $S_{\text{h}} = \left(\lfloor h(\frac{N}{R}) \rfloor + 1\right)$ and $S_{\text{l}} = \left(\lfloor l(\frac{N}{R}) \rfloor + 1\right)$. We compute $D_i =$

$|F(V_{t_{\text{tile}},i})|, i \in \{1,\ldots,N\}$. We then insert $D_{[S_\text{h}:S_\text{l}]}$ into U for the corresponding tile.

When we visualised some tiles (t_{tile}) in U we noticed that it is rather noisy and the peaks in the frequency space were slightly out of alignment on adjacent tiles in the same song. To compensate for this we convolved the frequency space for each tile with a one dimensional Gaussian first derivative filter given as $-\frac{2G}{B^2}e^{-\frac{G^2}{B^2}}$ where $G = -\lfloor 2B \rfloor, \lfloor -2B+1 \rfloor, \ldots, \lfloor 2B \rfloor$ and $B = b\left(\frac{N}{R}\right)$. Note that b is the bandwidth of the filter in Hz and this will be used as one of our main parameters. The formula for one dimensional convolution is given as

$$(f * g)[n] \stackrel{\text{def}}{=} \sum_{m=-\infty}^{\infty} f[m]\, g[n-m].$$

This filtering technique is critical for us to reliably distinguish pairs of tiles from the same song from their cosine. An advantage of convolution filtering over binning is you do not lose any resolution on the features. This Gaussian removes noise very effectively; it accentuates and widens large gradients and flattens out smaller gradients. The concept is exactly the same as edge detection in computer vision. We noticed that the harmonic peaks for instruments were slightly out of alignment on adjacent tiles for the same song, and as a result their cosine did not indicate their similarity. This filter was the key step in resolving that issue. After this filter was applied, characteristic squares representing the tracks appeared on the diagonal of the dissimilarity matrix.

Some vision references – think sobel, prewit filters

We have K Fourier feature tiles each of length $S_\text{l} - S_\text{h}$ and normalize them all by $f_{\text{norm}} = \frac{f}{|f|}$ where f is one feature tile. Then we can compute an $K \times K$ dissimilarity matrix $S(i,j)$ by taking one minus the dot products of all pairs of normalised features (one minus because we will be finding the minimum cost assignment in Section 8). See Figure 2 for a partial annotated illustration of this matrix on one of the shows.

7 Cost Matrix

We now have a dissimilarity matrix $S(i,j)$ as described in Section 6.

Based on our experience, it is highly unlikely to have songs that are shorter than 3 minutes or longer than 10 minutes. For this reason we have bound the minimum and maximum song length with w and W equal to 3×60 seconds and 13×60 seconds respectively.

Intuitively, tiles within tracks are reasonably similar on the whole, while pairs of tiles that do not belong to the same track are significantly more dissimilar. We define $C(f,t)$, the cost of a candidate track from tile f through tile t, to be the sum of the dissimilarities between all pairs of tiles inside it:

$$C(f,t) \;=\; \sum_{i=f}^{t} \sum_{j=f}^{t} S(i,j)$$

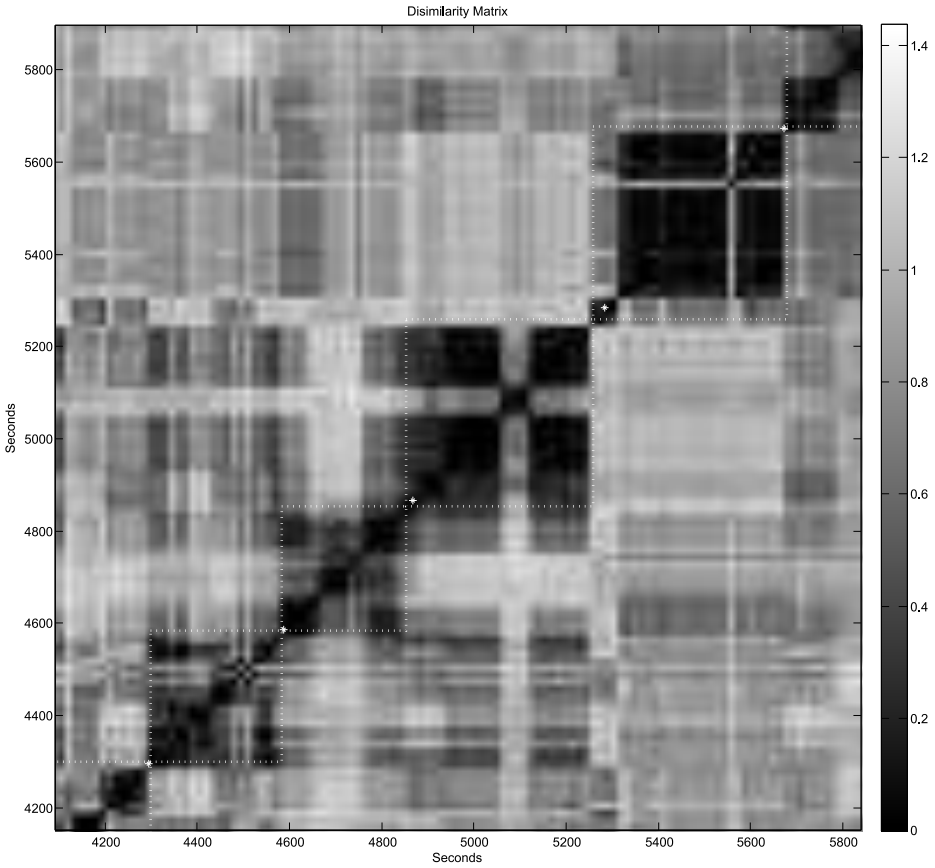


Fig. 2. An illustration of what the dissimilarity matrix $S(i, j)$ looks like.
white stars indicate the cue sheet indexes, and white dotted lines are the predicted indexes. This is from show *Trance Around The Workd 315 (Andy Duguid Guestmix) 9th April 2010*. Observe the transition overlapping two of the tracks.

We obtain the cost of a full segmentation by summing the costs of its tracks. The goal is now to efficiently compute the segmentation of least cost.

As a first step, we pre-compute C for each $1 \leq f \leq t \leq T$. Direct calculation using the definition takes $O(TW^3)$ time. However, we can compute the full cost matrix in $O(WT)$ time using the following recursion (for $f + 1 \leq t - 1$)

$$C(f,t) \;=\; C(f+1,t) + C(f,t-1) - C(f+1,t-1) + S(f,t) + S(t,f).$$

8 Computing Best Segmentation of Cost Matrix

A sequence $\boldsymbol{t} = (t_1, \ldots, t_{m+1})$ is called an m/T-segmentation if

$$1 = t_1 < \ldots < t_m < t_{m+1} = T + 1.$$

We use the interpretation that track $i \in \{1, \ldots, m\}$ comprises times $\{t_i, \ldots, t_{i+1} - 1\}$. Let $\mathbb{S}_m^T$ be the set of all m/T-segmentations. Note that there is a very large number of possible segmentations

$$|\mathbb{S}_m^T| \;=\; \binom{T-1}{m-1} = \frac{(T-1)!}{(m-1)!(T-m)!} = \frac{(T-1)(T-2)\cdots(T-m+1)}{(m-1)!} \geq \left(\frac{T}{m}\right)^{m-1}.$$

For large values of T, considering all possible segmentations using brute force is infeasible. For example, a two hour long show would have more than $\left(\frac{60^2 \times 2}{25}\right)^{24} \approx 1.06 \times 10^{59}$ possible segmentations!

We can reduce this number by imposing upper and lower bounds on the song length. Let $N(T, W, w, m)$ be the number of segmentations with time T (in seconds), W the upper bound (in seconds) of the song length, w the lower bound (in seconds) of the song length and m the number of tracks.

We can write the recursive relation $N(T, W, w, m) = \sum N(t_m - 1, W, w, m - 1)$, where the sum is taken over t_m such that

$$\begin{aligned} t_m &\leq T - w + 1 \\ t_m &\geq T - W + 1 \\ t_m &\geq (m-1)w + 1 \\ t_m &\leq (m-1)W + 1 \end{aligned}$$

The first two inequalities mean that the length of the last track is within an acceptable boundary between w and W. The last two inequalities mean that the lengths of the first $m - 1$ tracks are within the same boundaries.

We calculated the value of $N(7000, 60 \times 15, 190, 25)$ and got 5.20×10^{56} which is still infeasible to compute with brute force.

Our solution to this problem is to find a dynamic programming recursion.

The loss of an m/T-segmentation $\boldsymbol{t}$ is

$$\ell(\boldsymbol{t}) \;=\; \sum_{i=1}^{m} C(t_i, t_{i+1} - 1)$$

We want to compute

$$\mathcal{V}_m^T \;=\; \min_{\boldsymbol{t} \in \mathbb{S}_m^T} \ell(\boldsymbol{t})$$

To this end, we write the recurrence

$$\mathcal{V}_1^t \;=\; C(1, t)$$

and for $i \geq 2$

$$\begin{aligned}
\mathcal{V}_i^t &= \min_{\boldsymbol{t} \in \mathbb{S}_i^t} \ell(\boldsymbol{t}) \\
&= \min_{t_i} \min_{\boldsymbol{t} \in \mathbb{S}_{i-1}^{t_i - 1}} \ell(\boldsymbol{t}) + C(t_i, t) \\
&= \min_{t_i} C(t_i, t) + \min_{\boldsymbol{t} \in \mathbb{S}_{i-1}^{t_i - 1}} \ell(\boldsymbol{t}) \\
&= \min_{t_i} C(t_i, t) + \mathcal{V}_{i-1}^{t_i - 1}
\end{aligned}$$

In this formula t_i ranges from $(i-1)w + 1$ to $(i-1)W + 1$.

We have $T \times m$ values of $\mathcal{V}_m^T$ and calculating each takes at most $O(T)$ steps. The total time complexity is $O(T^2 m)$.

9 Methodology and Results

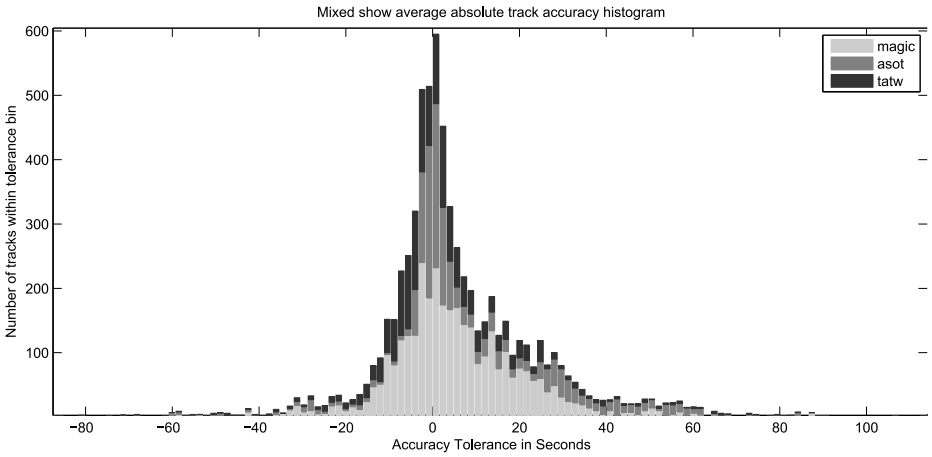
We took the first 10 episodes at from each radio show and found the best parameters with a naïve random search. The search was quite slow so we were only able to test a limited number of permutations (less than 1000) but believe that we have not suffered significantly on the performance because of this. We were optimising the parameter search for the overall average track accuracy, not the track precision metrics. For this reason the precisions are not as good as they could otherwise have been.

See Figures 3,4,5 for the results and Figure 9 for an overall absolute accuracy histogram.

10 Conclusion and Further Work

bla bla [1]

We feel that our results are reasonable for most practical applications. The average track length is 5 and a half minutes and the transitions alone can be longer in duration than 60 seconds which is the highest accuracy threshold used.



Parameters

C	Minimum Track Removal	120	Seconds
T_{sec}	Tile Size	5	Seconds
b	Bandwidth Filter	3	Hz
l	Low Pass Filter	1900	Hz
h	High Pass Filter	0	Hz

Index precision score

60 Seconds	93.4%
30 Seconds	74.6%
20 Seconds	63.6%
10 Seconds	54.5%
5 Seconds	49.3%
3 Seconds	40.2%
1 Seconds	18.7%

Fig. 3. A State of Trance results (ran on 100 shows, parameters tuned on first 10 shows). Overall average track accuracy (correct order) is 50.61 seconds.

Parameters

C	Minimum Track Removal	180	Seconds
T_{sec}	Tile Size	15	Seconds
b	Bandwidth Filter	5	Hz
l	Low Pass Filter	1900	Hz
h	High Pass Filter	0	Hz

Index precision score

60 Seconds	98.6%
30 Seconds	91.4%
20 Seconds	79.8%
10 Seconds	33.9%
5 Seconds	33.9%
3 Seconds	22.6%
1 Seconds	7.4%

Fig. 4. Magic Island results (ran on 198 shows, parameters tuned on first 10 shows). Overall average track accuracy (correct order) is 12.81 seconds.

Parameters

C	Minimum Track Removal	120	Seconds
T_{sec}	Tile Size	15	Seconds
b	Bandwidth Filter	5	Hz
l	Low Pass Filter	1900	Hz
h	High Pass Filter	0	Hz

Index precision score

60 Seconds	96.5%
30 Seconds	89.8%
20 Seconds	81.8%
10 Seconds	64.0%
5 Seconds	39.7%
3 Seconds	24.4%
1 Seconds	6.9%

Fig. 5. Trance Around The World results (ran on 89 shows, parameters tuned on first 10 shows). Overall average track accuracy (correct order) is 32.71 seconds.

Arguably; our results may be better than they appear because in many cases they are ostensibly more accurate than the human captured judging by the dissimilarity matrix visually.

The dissimilarity matrix we use is based mainly on instruments and we suspect humans also pay attention to the percussive/rhythmic features in the audio.

We have been trying to improve the results using a different type of feature extraction (autocorrelation based) which captures these percussive patterns, which is especially relevant for dance music (repetitive beats). At this stage we can't get a significant improvement in our results using this method so we have omitted it from the paper. Capturing rhythmic features is better suited for capturing the transitions themselves rather than the optimal change point during the transition.

References

1. A. Rauber, E. Pampalk, and D. Merkl, "Using psycho-acoustic models and self-organizing maps to create a hierarchical structuring of music by sound similarity," in *Proc. ISMIR*, pp. 71–80, 2002.
2. G. Tzanetakis and F. Cook, "A framework for audio analysis based on classification and temporal segmentation," in *EUROMICRO Conference, 1999. Proceedings. 25th*, vol. 2, pp. 61–67, IEEE, 1999.
3. G. Tzanetakis and P. Cook, "Multifeature audio segmentation for browsing and annotation," in *Applications of Signal Processing to Audio and Acoustics, 1999 IEEE Workshop on*, pp. 103–106, IEEE, 1999.
4. J. Foote and M. Cooper, "Visualizing musical structure and rhythm via self-similarity," in *Proceedings of the 2001 International Computer Music Conference*, pp. 419–422, 2001.
5. J. Foote, "Automatic audio segmentation using a measure of audio novelty," in *Multimedia and Expo, 2000. ICME 2000. 2000 IEEE International Conference on*, vol. 1, pp. 452–455, IEEE, 2000.
6. J. T. Foote and M. L. Cooper, "Media segmentation using self-similarity decomposition," in *Electronic Imaging 2003*, pp. 167–175, International Society for Optics and Photonics, 2003.
7. J. Foote, "A similarity measure for automatic audio classification," in *Proc. AAAI 1997 Spring Symposium on Intelligent Integration and Use of Text, Image, Video, and Audio Corpora*, 1997.
8. J. Foote, "Visualizing music and audio using self-similarity," in *Proceedings of the seventh ACM international conference on Multimedia (Part 1)*, pp. 77–80, ACM, 1999.
9. M. Cooper, J. Foote, E. Pampalk, and G. Tzanetakis, "Visualization in audio-based music information retrieval," *Computer Music Journal*, vol. 30, no. 2, pp. 42–62, 2006.
10. M. Cooper and J. Foote, "Scene boundary detection via video self-similarity analysis," in *Image Processing, 2001. Proceedings. 2001 International Conference on*, vol. 3, pp. 378–381, IEEE, 2001.
11. M. Cooper and J. Foote, "Summarizing popular music via structural similarity analysis," in *Applications of Signal Processing to Audio and Acoustics, 2003 IEEE Workshop on.*, pp. 127–130, IEEE, 2003.

12. T. Bolognesi, "Automatic composition: Experiments with self-similar music," *Computer Music Journal*, pp. 25–36, 1983.
13. M. M. Goodwin and J. Laroche, "A dynamic programming approach to audio segmentation and speech/music discrimination," in *Acoustics, Speech, and Signal Processing, 2004. Proceedings.(ICASSP'04). IEEE International Conference on*, vol. 4, pp. iv–309, IEEE, 2004.
14. A. Pikrakis, T. Giannakopoulos, and S. Theodoridis, "A speech/music discriminator of radio recordings based on dynamic programming and bayesian networks," *Multimedia, IEEE Transactions on*, vol. 10, no. 5, pp. 846–857, 2008.
15. M. M. Goodwin and J. Laroche, "Audio segmentation by feature-space clustering using linear discriminant analysis and dynamic programming," in *Applications of Signal Processing to Audio and Acoustics, 2003 IEEE Workshop on.*, pp. 131–134, IEEE, 2003.
16. J.-J. Aucouturier and M. Sandler, "Segmentation of musical signals using hidden markov models," *Preprints-Audio Engineering Society*, 2001.
17. J. S. Boreczky and L. D. Wilcox, "A hidden markov model framework for video segmentation using audio and image features," in *Acoustics, Speech and Signal Processing, 1998. Proceedings of the 1998 IEEE International Conference on*, vol. 6, pp. 3741–3744, IEEE, 1998.
18. G. Peeters, "Deriving musical structures from signal analysis for music audio summary generation:"sequence" and "state" approach," *Computer Music Modeling and Retrieval*, pp. 169–185, 2004.
19. O. Lartillot and P. Toiviainen, "A matlab toolbox for musical feature extraction from audio," in *Proceedings of the 10th International Conference on Digital Audio Effects, Bordeaux, France*, 2007.
20. J. Paulus, M. Müller, and A. Klapuri, "State of the art report: Audio-based music structure analysis," in *Proceedings of the 11th international society for music information retrieval conference*, pp. 625–36, 2010.
21. E. Peiszer, T. Lidy, and A. Rauber, "Automatic audio segmentation: Segment boundary and structure detection in popular music," *Proc. of LSAS*, 2008.
22. W. Chai and B. Vercoe, "Music thumbnailing via structural analysis," in *International Multimedia Conference: Proceedings of the eleventh ACM international conference on Multimedia*, vol. 2, pp. 223–226, 2003.
23. D. Kimber, L. Wilcox, *et al.*, "Acoustic segmentation for audio browsers," *Computing Science and Statistics*, pp. 295–304, 1997.
24. M.-H. Jian, C.-H. Lin, and A. L. Chen, "Perceptual analysis for music segmentation," in *Proceedings of SPIE*, vol. 5307, pp. 223–234, Citeseer, 2004.
25. H. Meinedo and J. Neto, "A stream-based audio segmentation, classification and clustering pre-processing system for broadcast news using ann models," in *Proceedings of Interspeech*, Citeseer, 2005.
26. B. Zhou and J. H. Hansen, "Efficient audio stream segmentation via the combined t¡ sup¿ 2¡/sup¿ statistic and bayesian information criterion," *Speech and Audio Processing, IEEE Transactions on*, vol. 13, no. 4, pp. 467–474, 2005.
27. D. Wang, R. Vogt, M. Mason, and S. Sridharan, "Automatic audio segmentation using the generalized likelihood ratio," in *Signal Processing and Communication Systems, 2008. ICSPCS 2008. 2nd International Conference on*, pp. 1–5, IEEE, 2008.
28. W. Chai, *Automated analysis of musical structure*. PhD thesis, Massachusetts Institute of Technology, 2005.

29. S. Chen and P. Gopalakrishnan, "Speaker, environment and channel change detection and clustering via the bayesian information criterion," in *Proc. DARPA Broadcast News Transcription and Understanding Workshop*, p. 8, Virginia, USA, 1998.
30. K. W. Church and J. I. Helfman, "Dotplot: A program for exploring self-similarity in millions of lines of text and code," *Journal of Computational and Graphical Statistics*, vol. 2, no. 2, pp. 153–174, 1993.
31. H. Kaprykowsky and X. Rodet, "Musical alignment using globally optimal short-time dynamic time warping," *FORTSCHRITTE DER AKUSTIK*, vol. 33, no. 2, p. 837, 2007.
32. M. Levy and M. Sandler, "Structural segmentation of musical audio by constrained clustering," *Audio, Speech, and Language Processing, IEEE Transactions on*, vol. 16, no. 2, pp. 318–326, 2008.
33. G. Peeters, A. La Burthe, and X. Rodet, "Toward automatic music audio summary generation from signal analysis," in *Proc. of ISMIR*, pp. 94–100, 2002.
34. M. Cettolo, M. Vescovi, and R. Rizzi, "Evaluation of bic-based algorithms for audio segmentation," *Computer Speech & Language*, vol. 19, no. 2, pp. 147–170, 2005.
35. S. Hargreaves, A. Klapuri, and M. Sandler, "Structural segmentation of multitrack audio,"
36. J.-J. Aucouturier and M. Sandler, "Finding repeating patterns in acoustic musical signals: Applications for audio thumbnailing.," *Watermark*, vol. 1, 2012.