Automatic Chorus Segmentation of Music Documents Using Self-Similarity

SIMON WIEST and ANDREAS ZELL
Wilhelm-Schickard-Institute for Computer Science
Universität Tübingen
Sand 1, D - 72076 Tübingen
GERMANY
http://www-ra.informatik.uni-tuebingen.de

Abstract: Content-based information retrieval is becoming increasingly important for audio document collections. We present a segmentation algorithm for discreetly sampled music documents that employs similarities in the frequency domain between small time slices to index chorus regions. Its performance is compared in two tasks with statistically derived estimators, trained on a test suite of music documents: In the audio thumbnailing task, a short time window has to be placed within a chorus segment; in the song structure task, a chorus detector has to segment a music document in chorus/non chorus regions.

Key-Words: Automatic audio segmentation, Self-similarity, Chorus detection, Audio thumbnail

1 Introduction

Three major current trends in the field of audio databases promote the need for new search and navigation techniques, especially content-based retrieval algorithms: First, the amount of digitally available documents is constantly increasing. Affordable digital storage capacity allows online access to vast audio collections. Second, helpful meta-data is often non-existent or flawed, because many of these databases are open collections (e.g. in peer-to-peer file sharing communities) — lacking centralized administration and indexing using established onthologies. Third, the nature of audio documents itself suggests novel query interfaces besides mere searching in textual meta-data, e.g. query-by-humming or query-by-example.

While text search engines have become common tools even for the casual Internet user, searching and retrieving audio files by content is still subject to re-search, from extracting meta-data by statistically analyzing the audio signal [11], segmenting at texture changes [9], generating ”audio fingerprints” of music pieces [1], developing interfaces like query-by-humming [4][7], and visualizing audio content [3].

Almost any audio application employs some form of temporal segmentation of the audio signal, i.e. seeks continuous regions in time where a specific feature is present (or absent). This paper focuses on segmentation by chorus in music documents.

Chorus detection has many obvious applications, like playing chorus thumbnails in sound file browsers or ”magnetic” region selectors in audio editors (see [2] for a more complete list). Another might exists in the field of open audio collections: In the absence of carefully crafted meta-data (e.g. peer-to-peer file shares) download transmission streams might be reordered to transfer a chorus segment first, then the complementing rest. This allows prehearing a characteristic portion of an audio document before downloading it completely. Chorus detection also might support quality assurance in open collections: Technically flawed versions of a music piece (e.g. consisting of a sound snippet repeated many times) can be spotted by the server offering the file.

The remaining paper is structured as following: Section 2 introduces a new segmentation algorithm used to find chorus sections in a music document. Section 3 explains the different estimators that are used for the experiments, followed by section 4 that describes the experiments and quality measures in detail. Finally, the results are discussed in section 5.

2 The Segmentation Algorithm

The chorus segmentation algorithm is based on self-similarity in the frequency domain. The basic idea is that a chorus is assumed to be a repeated region within a music document. The more often a region is repeated throughout the document, the more likely it belongs to a chorus segment. Our algorithm consists of five stages:
2.1 Preprocessing

All sound files are resampled to 22 kHz, 16 bit, mono channel in a preprocessing stage employing anti-aliasing filtering. This resampling reduces computing costs in the following frequency spectrum transformation step. If source documents already store information in the frequency domain (e.g. MPEG-1 Layer III), this data can be used directly.

2.2 Spectral Similarity

Then \( n \) log-power amplitude spectrum slices are calculated from hamming-scaled time windows of length \( \Delta t \). Each slice is compared with each other, resulting in a symmetrical self-similarity matrix \( S \) with \( n^2 \) elements. As similarity measure we use the dot product of the spectrum vectors, although other functions like the euclidean distance might be applied. As a small \( \Delta t \) leads to a large size of \( S \), we reduce both matrix dimensions after correlation for performance reasons by downsampling.

As Foote demonstrates in [3], the resulting matrix visualizes the temporal structure of a music document: Regions with the same music textures (e.g. containing a human solo singer as opposed to a fully arranged instrumental chorus) appear as visually different, rectangular blocks in the matrix (see Fig. 1a top left): Matching regions with high resp. low similarity result in bright resp. dark areas. Repeated sequences within the document are visible as white lines from top left to bottom right. The offset of these lines to the main diagonal in vertical and horizontal direction can be read as the lag time between an original region and its repetition.

2.3 Chorus Filtering

A critical step consists in extracting repeated sequences. We scan all lines parallel to the main diagonal at different offsets for continous windows of size \( w_{rep} \) that different in statistical terms from the window in the line above. The ratio of the means is employed as statistical feature in this paper, although others like the variance yield comparable results.

If the ratio is less than a given threshold \( r_{rep} \), the according window is marked as repeated region otherwise as non-repeated region. The result is the matrix \( R \), a filtered version of \( S \) (see Fig. 1a top right). Depending on \( w_{rep} \), the matrix \( R \) will show many short repetitions (beats and themes) or few long ones (chorus regions).

2.4 Merging

Inter-chorus repetitions are merged into one block by filtering the columns of \( R \) with a one-dimensional maximum filter kernel of height \( w_{rep} \). Additionally all repetitions with a time lag less than \( w_{rep} \) are removed.

2.5 Timeline projection

Finally, for each column in the matrix, the number of repeated blocks are summed up, resulting in a density function \( \delta(t) \) depicted in Fig. 1b. Regions with high density values (i.e. containing content that is repeated more often than other content in the audio document) are likely to be the chorus parts of a music piece due to its repeating nature; therefore regions above the average density \( \bar{\delta} \) are marked as chorus segments (see Fig. 1c).
3 Chorus Detectors

We define a chorus detector as a function that estimates for an input document for each moment in time \( t \), whether a specific signal – the chorus – is present or not:

\[
c(t) = \begin{cases} 
0, & t \text{ is outside chorus} \\
1, & t \text{ is inside chorus}
\end{cases}
\]

Additionally, we define an audio thumbnail time position \( t' \) in such way that the overlapping time of chorus regions in a document and a thumbnail window of size \( w_t \) centered around \( t' \) is maximal.

3.1 Statistically based detectors

To compare the performance of the new chorus detector based on self-similarity, we construct two other detectors that make use of statistical properties determined from a training set of segments done by human subjects. The set consists of music documents that were segmented independently by human subjects into chorus/no chorus sections resulting in a chorus feature function \( c_{s,d} \) for each document \( d \) and subject \( s \) (see subsection 4.1 for the material used in this research).

The global chorus probability for a given absolute playing time \( t \) is calculated using the normalized sum of all segmentations created by human subjects with

\[
p_{abs}(t) = \frac{1}{s \cdot d} \sum_{i=1}^{s} \sum_{j=1}^{d} c_{i,j}(t)
\]

This function can be used to construct a simple stochastic chorus detector

\[
c_{abs}(t) = \begin{cases} 
1, & p_{abs}(t) > 0.5 \\
0, & p_{abs}(t) \leq 0.5
\end{cases}
\]

that proposes chorus regions at absolute time positions derived from the training data (see Fig. 2). The audio thumbnail time is accordingly

\[
t'_{abs} = \arg \max s \mid s(t) = \int_{t-w_t/2}^{t+w_t/2} p_{abs}(x)dx
\]

Using relative playing time \( t_r = t/T \) (with \( T \) being the total length of a document) yields another chorus probability function (see Fig. 3)

\[
p_{rel}(t_r) = \frac{1}{s \cdot d} \sum_{i=1}^{s} \sum_{j=1}^{d} c_{i,j}(t_r \cdot T_d)
\]

Analog to the absolute case, a second, relative statistic detector \( c_{rel} \) and audio thumbnail time \( t'_{rel} \) can be deducted.

4 Experiments

4.1 Test suite material

To test the performance of the segmentation algorithm on real-world data, a benchmark set of 100 music files
was formed. Because commonly agreed test suites (e.g., like the TREC benchmarks in the field of text retrieval [6]) are still in the forming stage [10], the documents comprise the German Top 100 Single Charts of week 9 in 2002 as published by Media Control, a well-known German market research institute [5]. This approach excludes individual musical preferences of experimenters and subjects, while allowing statements about detector performance in a likely field of application.

The benchmark set was manually segmented by human subjects \(n = 3\), which took each subject between 12-15 h for the 100 files.

Figure 2 depicts \(p_{\text{abs}}\), suggesting a chorus segment in the time range between 01:55 and 03:50, with a thumbnail center position at \(t'_{\text{abs}} = 03:14\).

Figure 3 shows \(p_{\text{rel}}\) featuring a characteristic distribution with three peaks, segmenting the probability function in seven regions that map closely with the common composition scheme of pop songs: introduction and verse (A), chorus (B), verse (C), chorus (D), verse/intermezzo (E), chorus (F), coda (G). The chorus region F is about twice as long as B and D which reflects the habit of repeating the chorus several times at the end of a song. \(c_{\text{rel}}\) suggests three chorus segments at the positions given in table 1 and a thumbnail center position at \(t'_{\text{rel}} = 0.822\).

<table>
<thead>
<tr>
<th>segment</th>
<th>start time</th>
<th>end time</th>
<th>duration</th>
</tr>
</thead>
<tbody>
<tr>
<td>B</td>
<td>0.243</td>
<td>0.308</td>
<td>0.065</td>
</tr>
<tr>
<td>D</td>
<td>0.500</td>
<td>0.563</td>
<td>0.063</td>
</tr>
<tr>
<td>F</td>
<td>0.765</td>
<td>0.920</td>
<td>0.155</td>
</tr>
</tbody>
</table>

Tab. 1. Segmentation created by statistical estimator \(c_{\text{rel}}\). Time is relative to the length \(T\) of the individual audio document.

4.2 Task 1: Audio thumbnail

The first task is targeted towards music retrieval scenarios: A database query might return several hits from a larger music database. In order to inspect the results quickly, a result browser could play a few seconds of each audio document that ideally would be most characteristic for the piece. In the tradition of presenting query results from image databases as small "thumbnail" icons, the use of small, characteristic segments of an audio document is called audio thumbnailing [8].

In our task, thumbnails of window size \(w_t = 10\) s have to be chosen with maximum overlapping to human created chorus segments. A perfect thumbnail would overlap completely within a chorus region of a human segmentation, therefore the ratio \(o\) between overlapped chorus time and window size would be 1, while 0 in worst case. The mean overlap ratio \(\bar{o}_d\) averages \(o\) over all human segmentations for document \(d\).

Because human segmentations may differ and the ratio between chorus time and the total length varies from document to document, attaining a certain \(\bar{o}\) can be more or less likely. To make the ratios comparable between different documents, a cumulative distribution function assigns each ratio a probability \(P(\bar{o}_d)\) that indicates how likely a randomly chosen thumbnail overlaps equal or less time. A value of 1 therefore does not indicate that the thumbnail overlaps completely with a chorus region, but rather that it is equal or outperforming any other randomly chosen thumbnail.

The results are depicted in the histogram in Fig. 4. Absolute positioning leads to widely scattered scores, while relative positioning performs quite well. The reason is the common, characteristic structure of pop music pieces as revealed by \(p_{\text{rel}}\). Thumbnails chosen by self-similarity score best, with some few exceptions performing very weakly. Manual investigation turned
out that these songs lack clearly distinctable chorus regions; the segmentations of the different human subjects also disagreed more in these cases than in others.

Visual inspection also showed that the thumbnails placed by self-similarity tend to start at the beginning of chorus regions (in opposite to the statistical methods that place the thumbnails just somewhere within). This effect is strongly desirable for a number of applications and will be subject to further quantitative analysis.

4.3 Task 2: Song structure

In the second task we examine how the chorus detectors compare with humans in segmenting a complete song in chorus/non-chorus regions. It is important to bear in mind, that segmentations may vary among human subjects: there is nothing like a single, correct human segmentation.

To get a similarity quality measure, we first define the overlapping agreement between an estimated segmentation $c'$ and a human segmentation $c$ by integrating over the time of the document using a metric that rewards overlapping chorus respectively non-chorus time in both segmentations:

$$f(t) = \begin{cases} 1, & c'(t) = c(t) \\ 0, & c'(t) \neq c(t) \end{cases}$$

$$o(c', c) = \frac{1}{T} \int_0^T f(t) dt$$

For a document $d$, the average agreement $\bar{o}_d$ with the human segmentations is calculated. Again, as human segmentations may differ, $\bar{o}_d$ is linearly rescaled to $\bar{o}'_d \in [0; 1]$ using $\bar{o}_{d,max}$ as upper and $\bar{o}_{d,min}$ as lower boundaries with

$$\bar{\tau}_d(t) = \frac{1}{s} \sum_{i=1}^s c_{s,d}(t)$$

$$g(t) = \begin{cases} \bar{\tau}_d(t), & \bar{\tau}_d(t) > 0.5 \\ 1 - \bar{\tau}_d(t), & \bar{\tau}_d(t) \leq 0.5 \end{cases}$$

$$\bar{\tau}_{d,max} = \frac{1}{T} \int_0^T g(t) dt$$

$$\bar{\tau}_{d,min} = 1 - \bar{\tau}_{d,max}$$

A value $\bar{o}'_d$ of 1 indicates a perfect agreement at any given $t$ with the majority of human segmentations (“ideal segmentation”), while 0 results from complete disagreement. An unbiased random detector would score 0.5 for all documents in this metric.

The results of the structure task are shown in the histogram in Fig. 5. While $c_{abs}$ performs worst with only 1% of the songs over 0.75 and $c_{rel}$ scoring better ($c_{abs}$: 79.3%, $c_{rel}$: 87.1%).

In the structure task, segmentations by $c_{sel}$ agree to 75.9% with the ”ideal” human segmentations. For the same reason as stated above, statistic detectors perform well again, but worse than $c_{sel}$ ($c_{abs}$: 79.3%, $c_{rel}$: 87.1%).

Clearly, more research has to be done, like the applicability of segmentation by self-similarity to classes of different music styles (classic, country, folk, ...). Additionally, the number of human segmentation data sets

5 Conclusions

We presented a chorus detector for music documents based on similarities in the frequency domain. In comparison with randomly chosen thumbnails, the ones suggested by the detector $c_{sel}$ score better or equal in 94.4%. Due to the common structure of pop music pieces, statistic detectors perform also well, but worse than $c_{sel}$ ($c_{abs}$: 79.3%, $c_{rel}$: 87.1%).

In the structure task, segmentations by $c_{sel}$ agree to 75.9% with the ”ideal” human segmentations. For the same reason as stated above, statistic detectors perform well again, but worse than $c_{sel}$ ($c_{abs}$: 58.3%, $c_{rel}$: 66.7%).

Clearly, more research has to be done, like the applicability of segmentation by self-similarity to classes of different music styles (classic, country, folk, ...). Additionally, the number of human segmentation data sets
must be increased to reduce the influence of individual preferences.

In contrast to the statistical detectors, finding thumbnails by self-similarity has the ability to place audio thumbnails exactly at the beginning of a chorus segment, not just somewhere within. Because this behaviour is strongly desired in sound browsing scenarios, it will be a worthwhile subject for further research.

References:


