EUROGRAPHICS 2013 / I. Navazo, P. Poulin
(Guest Editors)

Volume 32 (2013), Number 2

Scalable Music: Automatic Music Retargeting and Synthesis

Simon Wenner[1] Jean-Charles Bazin[1] Alexander Sorkine-Hornung[2] Changil Kim[1,2] Markus Gross[1,2]

[1]ETH, Zurich, Switzerland [2]Disney Research Zurich, Switzerland

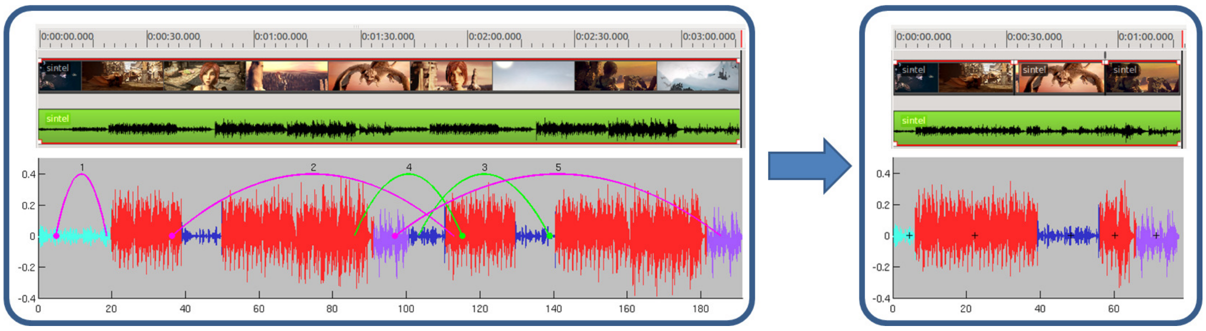


Figure 1: *Video editing as an application example of our structure-aware music retargeting. After several scenes have been cut from the video on the left by a video editor, our method automatically generates a correspondingly resized soundtrack by computing a series of inaudible, structure preserving jumps.*

Abstract

In this paper we propose a method for dynamic rescaling of music, inspired by recent works on image retargeting, video reshuffling and character animation in the computer graphics community. Given the desired target length of a piece of music and optional additional constraints such as position and importance of certain parts, we build on concepts from seam carving, video textures and motion graphs and extend them to allow for a global optimization of jumps in an audio signal. Based on an automatic feature extraction and spectral clustering for segmentation, we employ length-constrained least-costly path search via dynamic programming to synthesize a novel piece of music that best fulfills all desired constraints, with imperceptible transitions between reshuffled parts. We show various applications of music retargeting such as part removal, decreasing or increasing music duration, and in particular consistent joint video and audio editing.

Categories and Subject Descriptors (according to ACM CCS): I.3.8 [Computer Graphics]: Applications—H.5.5 [Information Interfaces and Presentation]: Sound and Music Computing—Signal analysis, synthesis, and processing

1. Introduction

In various applications related to computer graphics, a joint consideration of both the visual and auditory content is required. Prominent examples are the recent work of Berthouzoz et al. on placing cuts in interview videos [BLA12], or the growing interest in audio synthesis for physically based simulation [CJ11, ZJ10, CBBJR03]. A particular challenge arises when the duration of audio-visual data has to be adjusted. Consider the example in Figure 1. To shorten a video sequence it is often visually acceptable to simply cut out certain parts. However, straightforward removal of the corresponding parts from the accompanying audio track destroys the content continuity and leads to clearly noticeable, undesirable artifacts. Existing audio editing tools do not provide effective functionality for this type of rescaling, hence requiring cumbersome manual editing.

Various successful concepts have been proposed for *content-aware* rescaling of images [RGSS10], videos [SSSE00], or motion data [KGP02]. Applying similar ideas for content-aware retargeting of music seems evident. However, while for images and videos local structure analysis can produce convincing results [AS07,RSA08], musical structure, rhythm, and self-similarity are inherently more global, and violations of those relationships become immediately apparent to the listener.

In this paper we apply the concept of content-aware retargeting to audio, especially structured instrumental music, while respecting those global relationships. The core challenge addressed in our work lies in lengthening, shortening, or reshuffling a piece of music without introducing noticeable artifacts and without compromising its original (or any user-prescribed) high-level structure. Our solution to this problem is a combination of automatic music segmentation and analysis techniques, which are input to a global optimization framework. This optimization procedure synthesizes a novel piece of music by finding a set of least-noticeable jumps within the available audio signal that fulfill certain constraints such as overall duration, global structure, or audio-video synchronization. Our system works in real-time, and enables a variety of applications in graphics- and interaction-related areas such as audio editing, movie post-production, and video games.

2. Related Work

The most simple approach to retarget a musical piece would be to crop at the start and the end of the piece. But unlike in image processing where the borders of images mostly contain data of little importance, in music these parts often represent important structural or distinctive elements, e.g. the very recognizable sounds of clinking coins and ringing cash register at the beginning of the song "Money" by Pink Floyd. Simply removing these parts could alter the dramaturgy or story plot and also compromise the listening experience.

Professional music editing programs (such as Ableton Live, Logic and Cubase) are getting more and more popular. They offer functionalities to edit digital music including mixing, resampling, equalization and filtering. However they have very limited tools to retarget a musical piece. Therefore, in practice, an artist manually chooses segments to remove from the original music and the remaining segments are aligned to form the new musical piece. The boundaries between the segments are typically blended to restore continuity. High quality results can be obtained but it is a time consuming process and a highly skilled artist is required to find out appropriate cut positions. In contrast, our system can run in a fully automatic manner and offers an intuitive interface for non-expert users.

A more sophisticated type of edit supported by commercial editing tools is the audio time scale-pitch (TSP) modification. It changes the duration of an audio signal by adjusting the playback speed without affecting its pitch [GL08]. While it can produce reasonable results for small scaling factors (up to about 20%), it often leads to noticeable TSP artifacts (e.g. echo, drift and stretch distortion) for higher factors [LD99]. Recently methods have been proposed to allow for larger scaling factors but still suffer from significant limitations. For example the TSP implementation of *Paul's Extreme Sound Stretch* [Pau11] aims to minimize the amount of distortion. However the result has only little resemblance with the original music as it often gets transformed into a noisy sound texture. Based on a stretch-resistance measure, Liu et al. [LWB*11] elongate the stretchable parts intensively and other parts only lightly. It also removes repetitive sections using a greedy algorithm. However, it still produces noticeable TSP artifacts and the segment removal is based on heuristics and extensive blending. In contrast, our approach finds transitions between so-called bixels based on audio similarity and does not require to apply traditional stretching tools with error-leading high-scaling factors, which prevents from TSP artifacts such as distortion effects.

The most recent approaches to music retargeting are purely based on removing or repeating slices of the original piece of music. For example, Wenger et al. [WM11] synthesize a musical piece with the desired length and a minimal perceptual error by applying a heuristic best-first search. The method finds reasonable results for certain music genres but does not guarantee to reach the globally optimal solution of the proposed cost function. Hunt et al. [HB96] and Schwarz [Sch11] follow a different strategy by applying a global optimization [Vit67] in the contexts of concatenative speech synthesis and musical synthesis respectively. While they provide interesting results for particular audio signals, their method neither considers the rhythm nor the high-level structure of a piece of music, and thus cannot preserve these properties. The Echo Nest's Earworm application [Ear12] is based on timbre and pitch features. Various heuristics are used to extract a set of loops that are then selected in a greedy manner until a length close to the desired one is reached. Then the A* search algorithm [HNR68] is applied to connect the initial loop solution to the constrained start and end positions. Early tests showed that the algorithm often generates a solution far from the desired duration and sometimes fails to find a solution at all. The major restriction of the above approaches is their missing ability to preserve or enforce certain musical structures.

With a similar motivation to image texture synthesis [WL00, KSE*03], there have been some efforts to create audio textures (see [Sch11] for a recent review). For example, Parker et al. [PB04] adapt algorithms from image texture synthesis, namely image quilting and chaos mosaics. Lu et al. [LWZ04] compute transition probabilities based on a similarity measure which enables a probabilistic synthesis of audio textures. Interesting results can be obtained for repeating sounds (such as rain, fire and wind) and short ambient music but they fail for long sequences of structured music.

Some methods have been proposed to retarget music in a MIDI based representation. Simon et al. [SBSA05] apply concatenative synthesis to transform an audio recording with a known MIDI score into a new music with a target score. Lui et al. [LHS10] retarget musical style from audio recordings to MIDI. For this, a machine learning procedure is performed to obtain the style parameters that are then applied to a raw MIDI to generate a new audio file. Algorithmic composing poses a set of challenges outside the scope of this paper. We are more interested in maintaining a high level of similarity with the input musical piece rather than mimicking a certain music style.

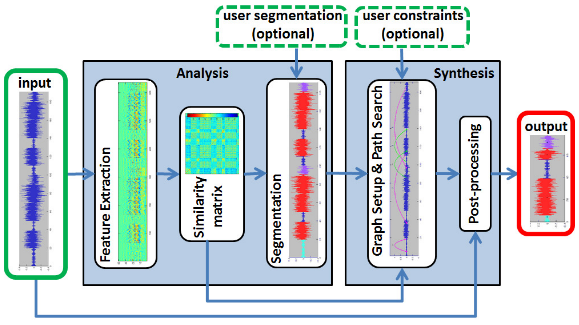


Figure 2: *Overview of the proposed algorithm.*

In the context of synchronization of music and video or animation, Yoon et al. [YLB09] describe a method to automatically cut a music video by analyzing and matching the brightness and the flow of music and video. Kim et al. [KPS03] generate dance motions of characters such that the motion beats match those of the accompanying music. These methods have in common that they do not modify the input musical piece. Yoon et al. [YLL06] synchronize a background music with a video by time-scaling the music. Recently a semi-automatic tool called UnderScore [RBM*12] has been proposed for adding musical underlays to spoken content. However it is limited to the alignment and dynamics (volume) adjustment of the input piece of music.

A different family of retargeting is music summarization, also known as music thumbnailing [LC00, CF02, BW05]. While this is an efficient tool to browse a large collection of music files or provide a compact representation for music retrieval systems, it is not well adapted to retarget a musical piece to a desired length.

3. System Overview

Our algorithm is illustrated in Figure 2. It accepts a piece of music selected by the user as input and returns a retargeted version with the desired length and structure while adhering to a set of optional constraints. The pipeline consists of two major components: an analysis and a synthesis phase.

The first phase analyzes the input musical piece by extracting a broad range of low to high-level information. For this, we extract a set of features (such as timbre and beats) to describe each part of the musical piece (Section 4.1). This permits to find pairs of perceptually similar parts in the input music. To obtain high-level information, we also perform segmentation using spectral clustering (Section 4.2). It automatically decomposes the music into different clusters and also captures its structure. Estimating this structure is important to maintain or edit it in the retargeted version.

The second phase is dedicated to the music synthesis (Section 5). Given a set of optional constraints, such as desired target length, structure, parts to remove or to include, start and end nodes, our algorithm searches for transition points inside music segments of the same cluster, and between segments of different clusters. The transition cost is based on the feature descriptors and beat-aligned similarity matrices. We represent the input music as a graph and mathematically formulate the problem as a constrained shortest path problem. The objective function is to obtain the most pleasing music in terms of audio continuity (smooth transitions) as well as beats, melody and higher-level structures. The main constraint is the desired duration of the music to synthesize. Additional constraints can encode various facts such as the importance of some parts of the input music (to promote their use in the retargeting) and which parts must be protected (not be modified) or not included in the output music. This is solved by a dynamic programming-based approach running at interactive rate, which allows the user to modify the constraints in real time and verify the new corresponding output piece of music.

4. Music Analysis

This section is dedicated to analysis of the input music. We show how to obtain both low and high-level information, respectively by feature extraction and segmentation.

4.1. Feature Extraction

To analyze and manipulate a music file automatically, it is important to extract relevant information from its audio signal. This process is known as feature extraction. In image processing, edge detection [FP02] and SIFT [Low04] are widely used techniques to extract and/or describe reliable features in images. A similar concept exists for music such as the Mel-Frequency Cepstral Coefficients (MFCCs) [Mer76]. MFCCs can be represented by a vector (typically 40 dimensions) and correspond to the short-term power spectrum of a sound, based on a linear cosine transform of a log power spectrum on a nonlinear mel scale of frequency [SVN37]. In practical terms, it has been shown that MFCCs are an appropriate feature descriptor for music analysis, description and retrieval algorithms [PMK10, Log00].

MFCCs are computed from a fast Fourier transform spectrum with a certain window size. Setting the window size to a fixed value and offset does not appropriately capture the rhythmic structure of the music. Instead, a dynamic kernel size based on the music beat can be used (e.g. [BW05] for audio thumbnailing). We propose to apply a similar approach for our scenario of music retargeting, which can prevent local changes in tempo and audible discontinuities during music synthesis. Beat is the basic unit of time in music [Ber76]. To automatically detect the beat positions, we apply BeatRoot [Dix07]. The length of beat segments generally varies between 250ms and 1.5s, depending mainly on the music genre (e.g., techno music tends to have short beat segments). As an analogy to the basic unit in image processing (i.e., pixels), we refer to these beat segments as “bixels” (contraction of the words “beat” and “pixel”) to avoid confusion with the segmentation terms of the next section.

Once the MFCCs are computed for each bixel, they can be used to compare the associated parts of the music and find perceptually similar parts. To compute the distance between all the MFCCs, we tested various distance measures.

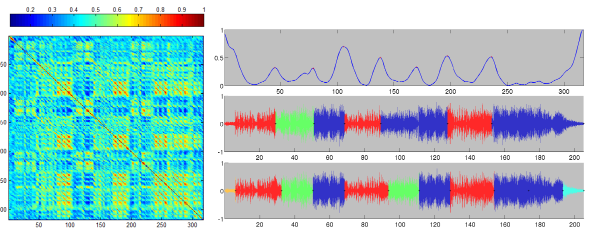

(a) (b)

Figure 3: *Music analysis for the song "Lucy in the sky with diamonds" by The Beatles. (a): beat-aligned self-similarity matrix (cf. Section 4.1). (b): segmentation into clusters (cf. Section 4.2). Top: novelty score computed from the similarity matrix. Middle: segmentation automatically obtained by our method. Bottom: ground truth segmentation for comparison.*

The most reliable results were obtained by Spearman's rank correlation [MW10] in terms of discriminative power and robustness. The distances are stored in a beat-aligned similarity matrix where the entry $S(i,j)$ contains the similarity measure between the i^{th} and j^{th} bixels of the input music. For completeness, a visual representation of S is shown in Fig 3-a. The resulting matrix captures the perceptual similarity between two bixels but does not contain any temporal context. To incorporate the dynamics of the music, we adopt the measure of Schödl et al. [SSSE00] originally developed for video similarity: we consider not only the current bixel but also the temporally adjacent bixels as:

$$S'(i,j) = \sum_{t=-m}^{m} w_t S(i+t, j+t). \tag{1}$$

The weights w_t are the normalized binomial coefficients and we set $m = 2$. A similar approach was used by Schwarz [Sch07] for audio concatenation. The value of $S'(i,j)$ is high when the i^{th} and j^{th} bixels are both perceptually and temporally consistent. S' is normalized to the interval $[0,1]$.

4.2. Segmentation

From the features described in Section 4.1, high-level structural information can now be obtained, which will permit maintenance or editing of the structure of the input musical piece during retargeting. We proceed in two steps: novelty score computation and then segmentation, following existing work in the music research community [PMK10,Foo00].

In the first step, we detect the similarity discontinuities because they tend to indicate the beginning/end of a structure element. This can be considered as an edge detection task in images [FP02] and a similar procedure can be performed for audio signals. We applied the weighted detection kernel proposed by Foote et al. [Foo00] along the diagonal of S' to get the novelty score $N(i)$ for each bixel i. We set the kernel size to 64 for all our experiments. Finally the local maxima of this novelty score correspond to similarity boundaries (edges) and the bixels inside the boundaries define *segments*. A typical result is illustrated in Fig 3-b. With the exception of one segment, the segmentation obtained by our method is in accordance with the ground truth segmentation (obtained manually) while running fully automatically.

The second step aims to cluster the different segments. We start by computing the mean of the MFCCs for each extracted segment to obtain their descriptor. Then given these descriptors, the segments are clustered using spectral clustering by the normalized cuts algorithm [SM00], widely used in the computer graphics and vision communities such as image and video segmentation. Compared to the popular K-means clustering [Mac67], spectral clustering does not depend on an initial solution and allows clusters to have different spatial extents and shapes. Therefore in practice, better segmentation results are obtained by spectral clustering. The number of clusters typically varies between 2 and 5, and can be interactively controlled by the user if necessary. Finally, each bixel of the input musical piece is linked to its corresponding cluster. Figure 3-b illustrates a typical example of novelty score and the final segmentation.

5. Music Retargeting and Synthesis

The previous section focused on analyzing the input music, especially how to extract low-level (feature descriptors) and high-level (music structure by segmentation) information. This section describes music synthesis for retargeting applications using the extracted information. It starts by defining the mathematical formulation of music retargeting as an optimization problem. Then we discuss the relevant optimization methods and present an efficient dynamic programming-based technique. Finally, we explain how to maintain or edit the structure of the input music using an approach inspired by motion graphs.

5.1. Mathematical Formulation

We mathematically formulate the retargeting as a labeling problem where a set of bixels of the input music is selected and combined with respect to a music-relevant cost function and constraints. In accordance to standard labeling notations, the i^{th} bixel of the input music is now represented by the label l_i. The audio perception consistency between two bixels l_i and l_j is measured by the similarity term $S'(l_i,l_j)$ defined in Eq. (1). Each bixel l_i is also associated with an importance term $I(l_i) \in [0,1]$ which permits to force or forbid the selection of some bixels, for example when the user particularly loves or dislikes a part of the input music. By default, the values $I(l_i)$ are set to uniform weights.

Let $\mathbf{l} = \{l_i | i = 1 \dots M\}$ define a bixel ordering that represents the retargeted musical piece of the desired length M. The objective function based on the importance and similarity metrics can then be written as:

$$E(\mathbf{l}) = \lambda \sum_{i=1}^{M} (1 - I(l_i)) + (1-\lambda) \sum_{i=1}^{M-1} D(l_i, l_{i+1}), \tag{2}$$

where the first term $1 - I(l_i)$ encodes the inverse importance of the bixel l_i. The second term represents the audio dissimilarity between two bixels and is defined as $D(l_i,l_j) = 1 - S'(l_i+1, l_j)$, i.e., temporally adjacent bixels in the input music have a dissimilarity cost of 0. The constant λ balances the influence of the two terms and we set $\lambda = 0.5$ for all the experiments shown in this paper.

5.2. Optimization Technique

We now present the approach applied to compute the labelling $\mathbf{l}$ minimizing Eq. (2) in a globally optimal manner. Multi-label optimization has been extensively studied in the computer vision and graphics communities [BVZ01, SZS03]. It is a challenging problem in general so existing methods rely on approximation, and thus are not globally optimal, or assume particular smoothness terms (e.g. symmetric cost). Contrary to general multi-label problems, two key observations are that the neighborhood is one-dimensional and the importance value $I(l_i)$ at Eq. (2) does not depend on the position of l_i in the returned solution. This allows us to merge the importance I and the dissimilarity D into a single cost term W defined as:

$$W(l_i,l_j) = \lambda\left(1 - I(l_j)\right) + (1-\lambda)D(l_i,l_j), \tag{3}$$

which represents the cost of placing the bixel l_i right before the bixel l_j. This permits the equivalent reformulation of Equation (2) as a complete directed graph whose N nodes are the bixels of the input music (see Section 4.1) and whose edge value between two nodes l_i and l_j corresponds to $W(l_i,l_j)$. Given such a graph, the goal is to select a path with the least cost and with the desired length M. We applied a dynamic programming (DP) similar to Viterbi [Vit67] to obtain this length-constrained path. The start and end nodes of this path are constrained to bixels l_s and l_e respectively (such as the starting and ending bixels of the input piece or any preferences of the user), i.e. $l_1 = l_s$ and $l_M = l_e$.

Let $C_s(i,k)$ be the cost of the minimum path from the start node l_s to the node l_i in k stops, which can be defined as:

$$C_s(i,k) = \min_{j\in\{1..N\}} \left(C_s(j,k-1) + W(l_j,l_i)\right). \tag{4}$$

The recursion is initialized with $C_s(i,1) = W(l_s,l_i)$. By applying DP, we finally obtain the globally optimal path of length M from l_s to l_t. We call a transition between two nodes (bixels) a *jump* if they are not temporally adjacent in the input music. Illustrations of jumps are shown in several figures of the paper, e.g. Fig 1. To synthesize the retargeted version, the bixels on the computed path are stitched together. A small local alignment of the signals is performed at jump positions to minimize the potential audio discontinuity. Our experiments showed that, thanks to the accuracy of our approach, cross-fading at jump positions was not necessary.

The described algorithm can achieve a duration precision up to the length of a beat, which is sufficient for most of applications. Some specific cases, like audio-video synchronization, need a precision up to single sample. Given our result obtained by DP, only a small scaling (less than 3% in our experiments) is required to achieve perfect synchronization. For such small scaling factors, TSP is appropriate (Section 2) and we applied the method of Verhelst et al. [VR93].

5.3. Structure Constraints

The retargeting algorithm of Section 5.2 is yet unaware of any high-level structures and thus cannot reproduce them. We now discuss how to generalize it for retargeting musical pieces whose overall structure must be reproduced.

Our modeling of high-level constraints is based on the concept of motion graphs in computer graphics. In the context of character animation, Kovar et al. [KGP02], Arikan et al. [AFO03] and Safonova et al. [SH08] find seamless transitions between different categories of character pose or action. For example, given a walking character as the start node and a kick action as the end node, the method aims to find a set of transitions between these nodes inside a corpus of motion data in such a way that a certain discontinuity measure of the animation is minimized. For our music retargeting application, the corpus corresponds to the entire input music and the categories of character poses/actions correspond to the clusters obtained in Section 4.2. A fundamental difference between the strategy of motion graphs and ours is that we enforce transition duration constraints (i.e., fix the output music length) instead of spatial constraints and do not allow data interpolation to avoid creating audio artifacts.

The method of Section 5.2 is modified as follows. The main idea is to constrain the bixel selection based on the clustering information. First, a dissimilarity matrix D_ν is defined for each cluster ν (see Section 4.2) and is computed from the bixels of cluster ν. This enforces that only bixels of cluster ν can be selected and thus prevents jumping to any clusters, which would alter the musical structure. A generalized cost matrix (Eq. (3)) is defined for each cluster ν:

$$W_\nu(l_i,l_j) = \lambda\left(1 - I(l_j)\right) + (1-\lambda)D_\nu(l_i,l_j). \tag{5}$$

In addition, we introduce a function $\psi()$ that allows us (i) controlling the duration of each cluster and (ii) maintaining or editing the structure of the input musical piece, that is to say control the cluster ordering in the final output path. Through an intuitive interface, we give the user the possibility to edit $\psi()$ according to his/her own preference in order to, for example, remove a cluster in the output (e.g., if its duration is considered too short or its music is unpleasant), edit the cluster ordering and control the duration of each cluster. By default $\psi()$ encodes the original label ordering and the duration of each label is proportionally scaled with respect to the retargeting scaling factor.

The recursive cost function of Eq. (4) is modified by the cost matrix W_ν depending on the function $\psi(k)$ which assigns a cluster to the bixel at position k in the output music:

$$C_s(i,k) = \min_{j\in\{1..N\}} \left(C_s(j,k-1) + W_{\psi(k)}(l_j,l_i)\right). \tag{6}$$

The DP approach is now applied on the modified version of C_s to find the globally optimal solution. An important aspect is that we do not process each cluster individually (local technique) but rather directly compute the entire path (global approach). Therefore intermediate nodes between clusters do not have to be specified: they are automatically selected from the consistency measure of Eq. (6). This approach returns an ordered set of jumps (i) within the same cluster to shorten/elongate the associated cluster and (ii) between different clusters to find transitions. These jumps form a path with the desired structure. A representative result is illustrated in Figure 1.

Thanks to label-dependent data and importance terms, a wide range of challenging retargeting applications is possible. For example, the user can define a new ordering of labels to perform structural reshuffling operations. It also permits completely removing certain parts of the input music so that they are not contained in the generated musical piece, and protecting some parts so that they are played entirely. Examples of such applications will be shown in Section 6.1.

6. Results

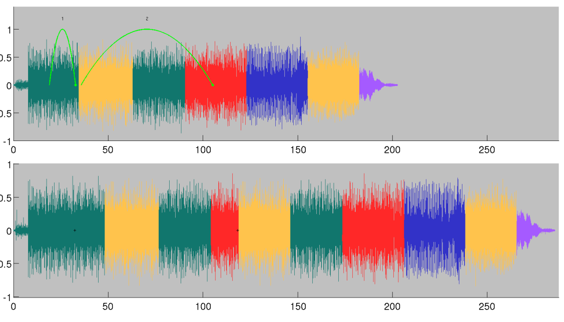
Our Matlab prototype runs on an Intel Core2 Quad Q6600 CPU with 8 GB of RAM. The pre-processing (see Section 4) of a 3-minute musical piece typically runs in less than one minute. The retargeting phase complexity (see Section 5) is linear in the desired output length. Typical execution time to resize a 3-minute musical piece to 6 minutes (about $N = 300$ nodes and $M = 600$ stops) by DP takes about 160ms with a non-optimized C++ code running on a single core CPU. Resizing a 3-minute musical piece to 1 minute (about $N = 300$ nodes and $M = 100$ stops) takes about 30ms. Therefore our method allows real-time music retargeting and user interactivity. Audio and video sequences, as well as additional results, are available in the **supplemental materials**.

6.1. Music Retargeting

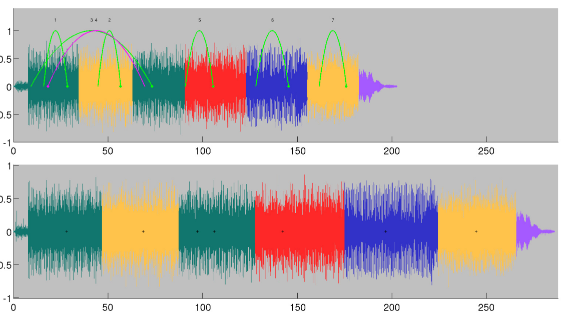
Music follows a certain dramaturgic plot and whose structure is important to interpret and appreciate a piece of music. In the following, we show that our approach can efficiently capture and take into account this global structure information to retarget a piece of music. Figure 1-left illustrates a typical example of segmentation (see Section 4.2) to capture the music structure. Figure 1-right represents the music automatically retargeted to the desired duration by the proposed algorithm. The associated jumps are shown in Figure 1-left. They are depicted as arcs with a dot at their start positions. Jumps that go forward in time are colored purple, backward jumps are colored green. One may note that the structure of the music is correctly preserved in the output. Another important observation is that only a limited number of jumps are needed, even with high scaling factors, which preserves as much of the input consistency as possible.

Figure 4 compares the retargeting results without and with structure-awareness. Figure 4-a demonstrates that, without structural information as in previous works (e.g. [WM11, HB96,Sch07]), the music structure is altered and undesirable repetitions of segments occur. On the contrary, Figure 4-b shows that the proposed structure-aware music retargeting algorithm (cf. Section 5.3) is able to correctly preserve the structure. This constitutes a key advantage of our approach.

Singing part removal: in some applications such as



(a)



(b)

Figure 4: *Retargeting of the song "Flood" by Tenpenny Joke to 150% of the original duration without (a) and with (b) structural constraints. See text for more details.*

background music generation for, e.g., advertisements, it is desirable to create instrumental-only versions of a song. Our algorithm can remove the singing parts and return a version that only consists of instrumental parts and with the desired duration. The input song is segmented into vocal and non-vocal clusters and then the optimization procedure is applied. The vocal parts can be extracted automatically by speech detection [RGS07] or karaoke files. A typical result where we removed the singing parts of the song "Trouble" by the band Coldplay is shown in Figure 5. An additional result obtained for the Russian song "Chucelo" by Nastya Yasnaya is available in the supplemental material.

Structural reshuffling: the structural segmentation (cf. Section 4.2) and the handling of the labelling constraints (cf. Section 5.3) permit reshuffling the input musical piece into a completely new structure. We refer to this application as *structural reshuffling* and a representative example is shown in Figure 6. The desired structure is defined interactively by the user. Such an assisted editing tool offers an efficient and interactive way to remix and transform existing music works into new creations.

6.2. Video Editing

During video post-production step, a video editor assembles the sequences shot or created during the production, in collaboration with the directors and producers. The music might be designed and recorded for specific parts and with a specific duration. If, at the end of post-production, the producer

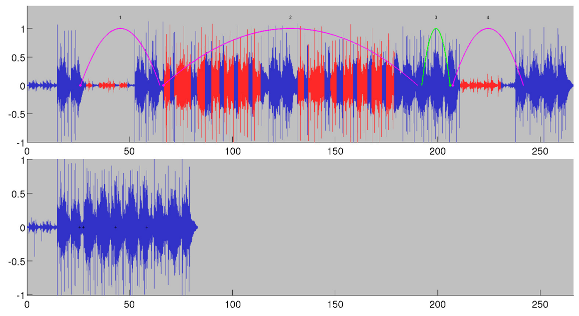


Figure 5: *Singing part removal for the song "Trouble" by the British alternative rock band Coldplay. Top: original song with singing parts (shown in red) automatically extracted by lyrics metadata, and the associated jumps by the proposed algorithm. Bottom: the retargeted music.*

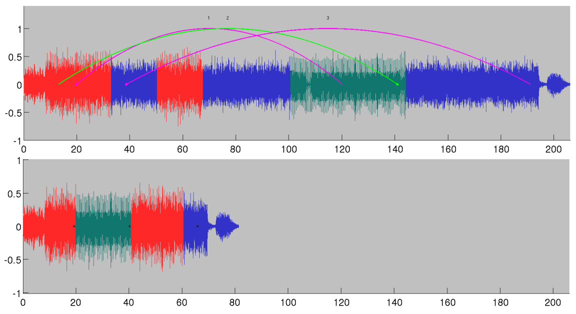
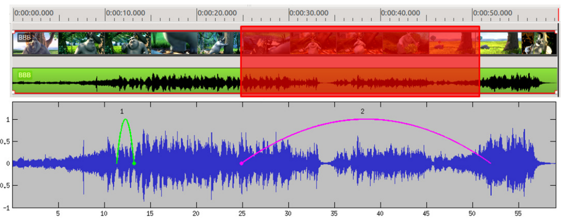


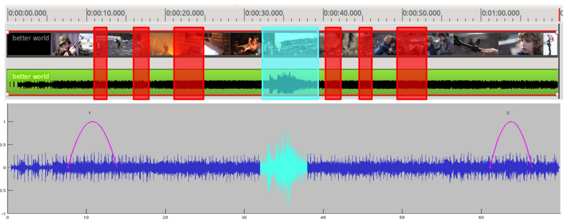
Figure 6: *Structural reshuffling of the song "Victory" by Alexander Blu. Top: input musical piece with its segmentation and the associated jumps obtained by the proposed algorithm. Bottom: the reshuffled version verifying the new structure desired by the user (red, green, red, blue).*

decides to delete some scenes, the soundtrack does not have the appropriate length and must be shortened. A simple approach is to cut the unwanted scenes and stitch the remaining video parts, along with their associated audio. To deal with audio discontinuity, blending is generally applied. We refer to this method as cut-and-blend. A similar case can occur during the development of animation sequences for high-end video games. In the following, we show some scenarios where scenes from a movie are removed and the audio is retargeted using our method. We compare each result to the "cut-and-blend" approach.

Action level control: in a part of the animation "Big Buck Bunny" (directed by Sacha Goedegebure, 2008), the action scenes are interleaved with humorous scenes. We demonstrate how those non-action scenes can be removed without compromising the consistency of audio and video. Since it changes the animation duration, we apply our method to retarget the music to the new necessary duration: from 60 to 33 seconds. The inputs are the original audio track and the segmentation (parts to keep and parts to remove). The transition results automatically obtained by our algorithm are shown in Fig 7(a). The complete retargeted music and a comparison to the cut-and-blend method are in the supplemental video.



(a)



(b)

Figure 7: *Examples of our music retargeting for advanced video editing. (a) Top: video editor time line of the animation movie "Big Buck Bunny". The video scenes to be removed are marked in red. Bottom: the resulting jumps to retarget the audio. (b) Music retargeting with synchronization constraint. The video editor timeline of the video clip "Better World" shows the scenes to remove marked in red and those to synchronize with the audio (bomb explosion) in blue.*

Synchronization: the audio track is often correlated with the video content: for example explosion sound effects when a bomb explodes. In such cases, when a movie is edited, it is important for these audio effects to stay synchronized with the associated video frames. Our algorithm can handle this synchronization constraint: the user simply needs to label the scenes to stay aligned with the audio. We demonstrate this feature on the music video "Better World" by the band Saturday's Tinitus (curtesy of Reto Troxler, 2010). We removed some scenes from the video and retargeted the soundtrack while maintaining the synchronization between the explosion sounds and the corresponding explosion visual effects. Figure 7(b) shows the explosion labelling, and the retargeted music is available in the supplemental video. The cut-and-blend approach provides a good synchronization but leads to very annoying discontinuities of the rhythm and melody at the cut positions. In contrast, our algorithm succeeds to align the explosions and creates high quality transitions. Moreover one may note that this task is usually very time consuming for an artist following a manual approach, even using professional tools, whereas our method runs fully automatically.

6.3. Retargeting Comparison

We conducted two user studies to measure the quality of our results. The first study is dedicated to the *local* consistency of the retargeted musical piece while the second one focuses on the *global* structure of the entire retargeted musical piece.

For the first user study, we evaluate the audio continuity and consistency of the jumps obtained by our algorithm. For this, we prepared a set of 32 10-second snippets taken

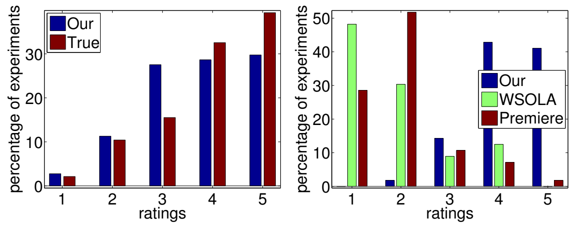


Figure 8: *Results of our user study. Left: comparison of the ratings for snippets (i) obtained by our retargeting algorithm and (ii) extracted without modification from the original musical pieces. Right: comparison of the ratings for rescaling entire musical pieces using our method, WSOLA, and Adobe Premiere. See details in Section 6.3.*

from the retargeted musical pieces obtained by the proposed method. Some of them correspond to the results obtained in the previous sections (from Section 6.1 to Section 6.2). All these snippets contain transitions within the same cluster or between different clusters (cf. Section 5.3). For comparison, we also considered snippets extracted directly from the original musical pieces (i.e. they do not contain any jumps) at random positions. We asked each of the 63 participants to grade the snippets from 1 (very annoying) to 5 (perfect, unnoticeable transitions). Results are shown in Fig 8-left. A similar number of retargeted and unmodified snippets (about 13%) was rated 1 and 2. The explanation is that some local changes of style are misinterpreted as transitions, even if they exist in the original musical piece. More than 85% of the retargeted snippets were rated acceptable (3), very satisfying (4) or perfect (5). Data analysis indicate a mean of 3.7 and 3.9, respectively for our retargeted and the unmodified snippets, with a similar standard deviation of 1.1 and a median of 4. It indicates that the results obtained by our method are statistically consistent with the original snippets.

We conducted a second user study about the global aspect of the entire retargeted musical pieces. The aim is to compare our algorithm to existing automatic retargeting methods. In early experiments, we applied *Paul's Extreme Sound Stretch* (PESS) and *Echo Nest Earworm* (cf. Section 2). However Earworm failed[†] creating a result for the majority of the musical pieces we tested. Both Earworm and PESS returned outputs violating the duration constraint, up to 53 seconds for a 2'12"-long piece. Moreover the audio quality of these outputs was obviously poor (strong artifacts and distortion) so we excluded them from a deeper user study. In contrast, our program returns a solution for all the tested musical pieces and always verifies the duration constraint. We presented to the participants a set of original musical pieces and their retargeted versions with various scaling factors (from 50% to 200% of the original duration) obtained by our content-aware algorithm presented in Section 5.3 and by two other relevant state-of-the-art approaches: (i) a TSP method called WSOLA [GL08] (cf. Section 2) similar to the feature included in the popular tools Ableton Live and Steinberg Cubase and (ii) the audio editor of the well-established and renowned commercial Adobe Premiere. We asked each of the participants to grade the entire musical pieces, in terms of global structure preservation, from 1 (strong artifacts) to 5 (perfect, unnoticeable modifications). The musical pieces have different properties (beats, structure) and correspond to various genres such as psychedelic folk, traditional flamenco and alternative rock. Results of the user study are displayed in Fig 8-right. It shows that our method clearly outperforms both WSOLA and Adobe Premiere. More than 80% of the musical pieces retargeted by our method were graded very satisfying (4) or perfect (5). Moreover analysis of the ratings shows that in every user response, our method was consistently rated better (respectively 92.8% and 91.0% of the tests) or equivalent to WSOLA and Adobe Premiere.

In addition to the evaluation of the above existing techniques, we showed our tool and results to professional video/audio editors and we received a very positive feedback from them. They particularly appreciated the numerous potential applications, the quality of our results and the fact that the user or artist can easily interact with the system and define his/her own preferences.

6.4. Discussion

The proposed method provides satisfying and reliable results for a wide range of instrumental musical genres from heavy metal to traditional flamenco. However, for audio pieces with no self-similar parts, the computed transitions might introduce discontinuities. Examples that fall into this category are musical pieces whose features are purely progressive (e.g. structure, loudness, melody or rhythm changes incrementally over time) such as the musical piece “Bohemian Rhapsody” by the band Queen. The quality of the generated musical pieces also depends on the reliability of the feature analysis (Section 4). For example, beat tracking tools generally assume that the musical piece has a reasonably regular beat, with no large discontinuities and silent parts [Dix07]. Thus if the input musical piece has such characteristics, its structure might not be preserved.

As demonstrated by the above experiments, our approach can reproduce a wide range of patterns thanks to the beat detection and structure-awareness. Nevertheless some repetitive patterns that are relatively far apart (e.g. more than 5-10 seconds) might be not captured. An example for the movie “Oceania” (directed by Harpreet Dehal, 2008) is in the supplemental material. While the jumps are barely noticeable, the beat is preserved and our results compare favorably to the cut-and-blend technique, music professionals might notice that the interval between the piano bass chords is slightly altered once. More generally, our current system does not capture medium-scale features, such as bars (also called measures), and thus they cannot be correctly preserved. A potential solution could be to add information from instrumental analysis [Mar06]. Note that our overall approach remains valid and can easily incorporate such additional features thanks to the generality of our system.

[†] memory error or Python segmentation fault

As stated above, our system is dedicated to instrumental musical pieces rather than radio songs containing lyrics. Our system can still be applied and synthesize such songs but the resulting jumps might alter the continuity or meaning of the lyrics. To solve this issue, we automatically detect the lyrics of a song from karaoke files and protect the associated audio parts (see Section 5.3): the retargeted version contains vocal parts and the lyrics are not altered. Alternative solutions to deal with lyrics could be voice detection [RGS07], speech recognition and techniques of Natural Language Processing to maintain meaningful lyrics.

7. Conclusions

We proposed an algorithm for converting existing music tracks into scalable music with varying length, capturing and maintaining the music's high-level structure and supporting additional constraints such as importance or position of specific parts. Our program runs in a fully automatic way and still offers the user a high degree of control over the final result through optional constraints. We built on ideas from image and video processing research and demonstrated that audio processing may benefit from similar concepts, enabling more effective techniques. Still, many questions are unique to audio processing and require dedicated features (e.g. MFCC and segmentation) and novel solutions (e.g. length-constrained cluster-based shortest path). We presented some challenging applications that would be difficult or time consuming with a manual approach or existing tools. Experimental results and user studies showed that our approach provides interesting improvements over existing commercial packages and algorithms. Finally, while this paper focused on retargeting of music, we believe that our underlying concept of global and structured content retargeting can be applicable to other graphics-related problems such as motion retargeting.

References

[AFO03] Arikan O., Forsyth D. A., O'Brien J. F.: Motion synthesis from annotations. In *SIGGRAPH* (2003). 5

[AS07] Avidan S., Shamir A.: Seam carving for content-aware image resizing. *SIGGRAPH* (2007). 1

[Ber76] Berry W.: *Structural functions in music*. Prentice-Hall, 1976. 3

[BLA12] Berthouzoz F., Li W., Agrawala M.: Tools for placing cuts and transitions in interview video. In *SIGGRAPH* (2012). 1

[BVZ01] Boykov Y., Veksler O., Zabih R.: Fast approximate energy minimization via graph cuts. *PAMI* (2001). 5

[BW05] Bartsch M. A., Wakefield G. H.: Audio thumbnailing of popular music using chroma-based representations. *IEEE Transactions on Multimedia* (2005). 3

[CBBJR03] Cardle M., Brooks S., Bar-Joseph Z., Robinson P.: Sound-by-numbers: motion-driven sound synthesis. In *SIGGRAPH/Eurographics Symposium on Computer Animation* (2003). 1

[CF02] Cooper M., Foote J.: Automatic music summarization via similarity analysis. In *International Conference on Music Information Retrieval* (2002). 3

[CJ11] Chadwick J. N., James D. L.: Animating fire with sound. *SIGGRAPH* (2011). 1

[Dix07] Dixon S.: Evaluation of the audio beat tracking system BeatRoot. *Journal of New Music Research* (2007). 3, 8

[Ear12] Earworm API by The Echo Nest: accessed from http://the.echonest.com/ (2012). 2

[Foo00] Foote J.: Automatic audio segmentation using a measure of audio novelty. In *IEEE International Conference on Multimedia and Expo* (2000). 4

[FP02] Forsyth D. A., Ponce J.: *Computer vision: a modern approach*. Prentice Hall, 2002. 3, 4

[GL08] Grofit S., Lavner Y.: Time-scale modification of audio signals using enhanced WSOLA with management of transients. *IEEE Transactions on Audio, Speech, and Language Processing* (2008). 2, 8

[HB96] Hunt A. J., Black A. W.: Unit selection in a concatenative speech synthesis system using a large speech database. In *IEEE International Conference on Acoustics, Speech, and Signal Processing* (1996). 2, 6

[HNR68] Hart P. E., Nilsson N. J., Raphael B.: A formal basis for the heuristic determination of minimum cost paths. *IEEE Trans. Systems Science and Cybernetics* (1968). 2

[KGP02] Kovar L., Gleicher M., Pighin F.: Motion graphs. In *SIGGRAPH* (2002). 1, 5

[KPS03] Kim T.-h., Park S. I., Shin S. Y.: Rhythmic-motion synthesis based on motion-beat analysis. *SIGGRAPH* (2003). 3

[KSE*03] Kwatra V., Schödl A., Essa I., Turk G., Bobick A.: Graphcut textures: Image and video synthesis using graph cuts. *SIGGRAPH* (2003). 2

[LC00] Logan B., Chu S.: Music summarization using key phrases. In *IEEE International Conference on Acoustics, Speech, and Signal Processing* (2000). 3

[LD99] Laroche J., Dolson M.: Improved phase vocoder time-scale modification of audio. *IEEE Transactions on Speech and Audio Processing* (1999). 2

[LHS10] Lui S., Horner A., So C.: Retargeting expressive musical style from classical music recordings using a support vector machine. *Journal of the Audio Engineering Society* (2010). 2

[Log00] Logan B.: Mel frequency cepstral coefficients for music modeling. In *International Symposium on Music Information Retrieval* (2000). 3

[Low04] Lowe D. G.: Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision* (2004). 3

[LWB*11] Liu Z., Wang C., Bai Y., Wang H., Wang J.: Musiz: a generic framework for music resizing with stretching and cropping. In *ACM International Conference on Multimedia* (2011). 2

[LWZ04] Lu L., Wenyin L., Zhang H.-J.: Audio textures: theory and applications. In *IEEE Transactions on Speech and Audio Processing* (2004). 2

[Mac67] MacQueen J. B.: Some methods for classification and analysis of multivariate observations. In *Berkeley Symposium on Mathematical Statistics and Probability* (1967). 4

[Mar06] Marolt M.: A mid-level melody-based representation for calculating audio similarity. In *International Conference on Music Information Retrieval* (2006). 8

[Mer76] Mermelstein P.: Distance measures for speech recognition: Psychological and instrumental. In *Pattern Recognition and Artificial Intelligence*. 1976. 3

[MW10] MYERS J. L., WELL A. D.: *Research Design and Statistical Analysis*. Lawrence Erlbaum Associates, New Jersey, 2010. 4

[Pau11] PAUL N. O.: Paul's Extreme Sound Stretch: accessed from http://hypermammut.sourceforge.net/ (2011). 2

[PB04] PARKER J., BEHM B.: Creating audio textures by example: tiling and stitching. In *IEEE International Conference on Acoustics, Speech, and Signal Processing* (2004). 2

[PMK10] PAULUS J., MÜLLER M., KLAPURI A.: Audio-based music structure analysis. In *International Conference on Music Information Retrieval* (2010). 3, 4

[RBM*12] RUBIN S., BERTHOUZOZ F., MYSORE G., LI W., AGRAWALA M.: UnderScore: musical underlays for audio stories. In *ACM Symposium on User Interface Software and Technology* (2012). 3

[RGS07] RAMIREZ J., GÓRRIZ J. M., SEGURA J. C.: Voice activity detection. fundamentals and speech recognition system robustness. *Robust Speech Recognition and Understanding* (2007). 6, 9

[RGSS10] RUBINSTEIN M., GUTIERREZ D., SORKINE O., SHAMIR A.: A comparative study of image retargeting. *SIGGRAPH Asia* (2010). 1

[RSA08] RUBINSTEIN M., SHAMIR A., AVIDAN S.: Improved seam carving for video retargeting. *SIGGRAPH* (2008). 1

[SBSA05] SIMON I., BASU S., SALESIN D. H., AGRAWALA M.: Audio analogies: Creating new music from an existing performance by concatenative synthesis. In *International Computer Music Conference* (2005). 2

[Sch07] SCHWARZ D.: Corpus-based concatenative synthesis. In *IEEE Signal Processing Magazine* (2007). 4, 6

[Sch11] SCHWARZ D.: State of the art in sound texture synthesis. In *International Conference on Digital Audio Effects* (2011). 2

[SH08] SAFONOVA A., HODGINS J. K.: Synthesizing human motion from intuitive constraints. In *Artificial Intelligence Techniques for Computer Graphics*. 2008. 5

[SM00] SHI J., MALIK J.: Normalized cuts and image segmentation. *PAMI* (2000). 4

[SSSE00] SCHÖDL A., SZELISKI R., SALESIN D. H., ESSA I.: Video textures. In *SIGGRAPH* (2000). 1, 4

[SVN37] STEVENS S. S., VOLKMANN J., NEWMAN E. B.: A scale for the measurement of the psychological magnitude pitch. *Journal of the Acoustical Society of America* (1937). 3

[SZS03] SUN J., ZHENG N., SHUM H.: Stereo matching using belief propagation. *PAMI* (2003). 5

[Vit67] VITERBI A. J.: Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. In *IEEE Transactions on Information Theory* (1967). 2, 5

[VR93] VERHELST W., ROELANDS M.: An overlap-add technique based on waveform similarity (WSOLA) for high quality time-scale modification of speech. In *IEEE International Conference on Acoustics, Speech, and Signal Processing* (1993). 5

[WL00] WEI L., LEVOY M.: Fast texture synthesis using tree-structured vector quantization. In *SIGGRAPH* (2000). 2

[WM11] WENGER S., MAGNOR M.: Constrained example-based audio synthesis. In *IEEE International Conference on Multimedia and Expo* (2011). 2, 6

[YLB09] YOON J.-C., LEE I.-K., BYUN S.: Automated music video generation using multi-level feature-based segmentation. *Multimedia Tools and Applications* (2009), 197–214. 3

[YLL06] YOON J.-C., LEE I.-K., LEE H.-C.: Feature-based synchronization of video and background music. In *International Workshop on Intelligent Computing in Pattern Analysis/Synthesis* (2006). 3

[ZJ10] ZHENG C., JAMES D. L.: Rigid-body fracture sound with precomputed soundbanks. *SIGGRAPH* (2010). 1