Automatic Audio Segmentation Using A Measure of Audio Novelty

Jonathan Foote

FX Palo Alto Laboratory, Inc.
3400 Hillview Avenue
Palo Alto, CA 94304
foote@pal.xerox.com

***Abstract* -- This paper describes methods for automatically locating points of significant change in music or audio, by analyzing local self-similarity. This method can find individual note boundaries or even natural segment boundaries such as verse/chorus or speech/music transitions, even in the absence of cues such as silence. This approach uses the signal to model itself, and thus does not rely on particular acoustic cues nor requires training. We present a wide variety of applications, including indexing, segmenting, and beat tracking of music and audio. The method works well on a wide variety of audio sources.**

I. Introduction

In video, the frame-to-frame difference is often used as a measure of novelty, which is highly useful for automatic segmentation and key frame extraction. A similar measure for audio would have many useful applications, however, computing audio novelty is significantly more difficult than video. Straightforward spectral differences are not often useful because they typically give too many false alarms. Typical speech and music spectra constantly fluctuate, and it is not a simple task to discriminate significant changes from ordinary variation.

We present methods of estimating the instantaneous audio novelty by analyzing self-similarity. At each instant, the self-similarity for past and future regions is computed, as well as the cross-similarity between the past and future. A significantly novel point will have high self-similarity in the past and future and low cross-similarity. The extent of the "past" and "future" can be varied to change the scale of the analysis. Using a short time extent allows individual notes to be found, while longer events, such as musical passages, can be found by considering wider windows. The result is a measure of how novel the source audio is at any time.

II. Technical Details

The audio is parameterized using standard spectral analysis. Each analysis frame is tapered with a Hamming window and transformed into the frequency domain using a fast Fourier transform (FFT). The logarithm of the magnitude of the FFT is used as an estimate of the power spectrum of the signal in the window. High frequency components above Fs/4 are discarded as they are not as useful for the similarity calculation. The resulting vector characterizes the spectral content of a window.

Note that the actual parametrization is not crucial as long as "similar" sounds yield similar parameters. Different parameterizations may be very useful for different applications; for example experiments have shown that the MFCC representation, which preserves the coarse spectral shape while discarding fine harmonic structure due to pitch, may be particularly appropriate for certain applications. Psychoacoustically motivated parameterizations, like those described by Slaney in [1], may be especially appropriate, particularly if they better reproduce the similarity judgments of human listeners.

A. Distance Matrix Embedding

Once the audio has been parameterized, it is then *embedded* in a 2-dimensional representation. The key is a measure D of the (dis)similarity between feature vectors $\boldsymbol{v}_i$ and $\boldsymbol{v}_j$ calculated between audio frames i and j. A simple distance measure is the Euclidean distance in the parameter space; another useful metric is the cosine of the angle between the parameter vectors.

$$D_C(i,j) \equiv \frac{\boldsymbol{v}_i \bullet \boldsymbol{v}_j}{\|\boldsymbol{v}_i\| \|\boldsymbol{v}_j\|}$$

This has the property that it can yield a large similarity score even if the vectors are small in magnitude. This is generally desirable so that similar regions with low energy will be judged highly similar. For most applications, subtracting the spectral mean from each vector produces a more pronounced similarity score.

The distance measure is a function of two frames, hence instants in the source signal. It is convenient to consider the similarity between all possible instants in a signal. This is done by *embedding* the distance measure in a two-dimensional representation. The matrix $\boldsymbol{S}$ contains the similarity metric calculated for all frame combinations, hence time indexes i and j such that the i, jth element of $\boldsymbol{S}$ is $D(i,j)$. In general, $\boldsymbol{S}$ will have maximum values on the diagonal (because every window will be maximally similar to itself); furthermore if D is symmetric then $\boldsymbol{S}$ will be symmetric as well. To simplify computation, the similarity can be represented in the "slanted" domain $\boldsymbol{L}(i,l)$ where $l = i - j$ is the lag. This is particularly true in for the applications presented later, where the similarity is only required for relatively small lags and not all combinations of i and j. Computing $\boldsymbol{L}$ only for small, non-negative values of l can result in substantial reductions in computation and storage over the full $\boldsymbol{S}$ matrix.

$\boldsymbol{S}$ can be visualized as a square image [2]. Each pixel i, j is colored with a gray scale value proportional to the similarity measure $D(i,j)$. These visualizations let us clearly see the structure of an audio file. Regions of high audio similarity, such as silence or long sustained notes, appear as bright squares on the diagonal. Repeated notes are visible as bright off-diagonal rectangles. If the music has a high degree of repetition, this will be visible as diagonal stripes or checkerboards, offset from the main diagonal by the repetition time.

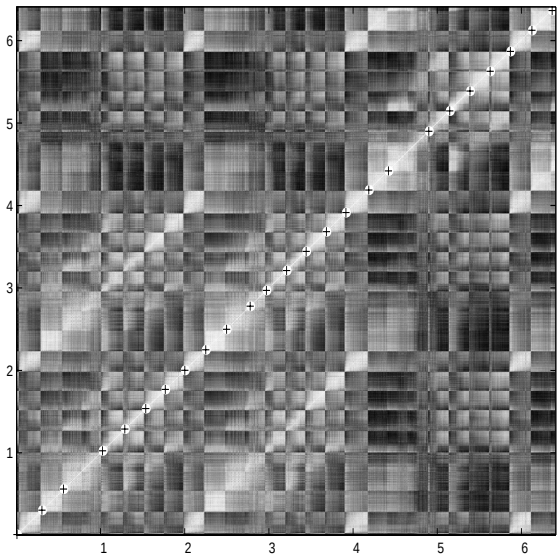

Figure 2. Gould performance showing note boundaries

Figure 2 shows the first seconds of Bach's *Prelude No. 1 in C Major*, from *The Well-Tempered Clavier*, BVW 846. The image visualizes a 1963 piano performance by Glenn Gould. (In this image, the origin is at the lower left and time increases both with height and to the right.) The musical structure is clear from the repetitive motifs; multiples of the repetition time can be seen in the off-diagonal stripes parallel to the main diagonal. Figure 1 shows the first few bars of the score: the repetitive nature of the piece should be clear even to those unfamiliar with musical notation. The visualization makes both the structure of the piece and details of performance visible. 34 notes can be seen as squares along the diagonal. The repetition time is visible in the off-diagonal stripes parallel to the main diagonal, as well as the initial C note which is repeated at 2, 4, and 6 seconds.

B. Kernel Correlation

The structure of ***S*** is key to the novelty measure. Consider a simple "song" having two successive notes of different pitch, for example a cuckoo call. When visualized, ***S*** for this example will look like a 2 x 2 checkerboard. White squares on the diagonal correspond to the notes, which have high self-similarity; black squares on the off-diagonals correspond to regions of low cross-similarity. Using the cosine metric, similar regions will be close to 1 while dissimilar regions will be closer to -1. Finding the instant when the notes change is as simple as finding the crux of the checkerboard. This can be done by correlating ***S*** with a kernel that itself looks like a checkerboard. (For obvious reasons, we call this class "checkerboard" kernels.) Perhaps the simplest is the 2x2 unit kernel which can be decomposed into "coherence" and "anti-coherence" kernels as shown

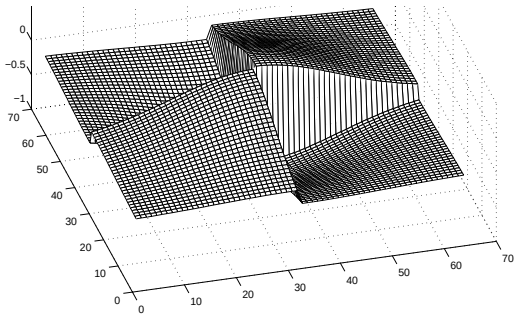

Figure 3. 64 x 64 checkerboard kernel with Gaussian taper

$$C = \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} - \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$$

The first term measures the self-similarity on either side of the center point; this will be high when both regions are self-similar. The second term measures the *cross-similarity* between the two regions; this will be high when the regions are substantially similar, thus with little difference across the center point. The difference of the two values estimates the *novelty* of the signal at the center point; this will have a high value when the two regions are self-similar but different from each other. Larger kernels are easily constructed by forming the Kronecker product of C with a matrix of ones, for example,.

$$\begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix} \otimes \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 1 & -1 & -1 \\ 1 & 1 & -1 & -1 \\ -1 & -1 & 1 & 1 \\ -1 & -1 & 1 & 1 \end{bmatrix}$$

Kernels can be smoothed to avoid edge effects using windows that taper towards zero at the edges. For the experiments presented here, a radially-symmetric Gaussian function is used. Figure 3 shows a 3-dimensional plot of a 64 x 64 checkerboard kernel with a radial Gaussian taper having $\delta = 32$.

Correlating a checkerboard kernel with the similarity matrix ***S*** results in a measure of novelty. Imagine sliding *C* along the diagonal of our example, and summing the element-by-element product of *C* and ***S***. When *C* is over a relatively uniform region the

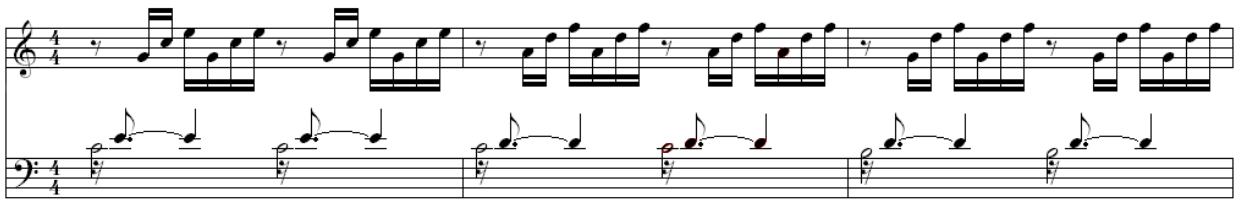

Figure 1. First bars of Bach's *Prelude No. 1 in C Major*, BVW 846, from *The Well-Tempered Clavier*

positive and negative regions will tend to sum to zero. Conversely, when C is positioned exactly at the crux of the checkerboard, the negative regions will multiply the negative regions of low cross-similarity, and the overall sum will be large. Thus calculating this correlation along the diagonal of $\boldsymbol{S}$ gives a time-aligned measure of audio novelty $N(i)$, where i is the frame number, hence time index corresponding to the original source audio.

$$N(i) = \sum_{m=-L/2}^{L/2} \sum_{n=-L/2}^{L/2} C(m,n) S(i+m, i+n)$$

By convention, the kernel has a width (lag) of L and is centered on 0,0. For computation, $\boldsymbol{S}$ can be zero-padded to avoid undefined values, or, as in the present examples, only computed for the interior of the signal where the kernel overlaps $\boldsymbol{S}$ completely. Note that only regions of $\boldsymbol{S}$ with a lag of L or smaller are used; thus the slant representation is particularly helpful. Also, only one-half of the values under the double summation (those for $m \geq n$) need be computed because typically both $\boldsymbol{S}$ and C are symmetric.

The width of the kernel L directly affects the properties of the novelty measure. A small kernel detects novelty on a short time scale. Larger kernels average over short-time novelty and detect longer structure, such as musical transitions, key modulations, or symphonic movements. Figure 5 shows the novelty score N, computed on the similarity matrix of Figure 2. Results using two kernel widths are shown (the 2S kernel plot was offset slightly upwards for clarity). The shorter kernel clearly picks out the individual note events, though some notes are slurred and are not as distinct. This method clearly identifies the onset of each note, without analyzing explicit features such as pitch, energy, or silence. The the longer kernel yields peaks at the boundaries of 8-note phrases (at 2, 4, and 6 seconds). Each peak occurs exactly at the downbeat of the first note in each phrase. Note that this method has no *a-priori* knowledge of pitch, notes, or musical phrases, but is finding perceptually and musically significant points.

C. Extracting Segment Boundaries

As mentioned above, extrema in the novelty score correspond to large changes in the audio. These points often serve as good boundaries for segmenting the audio. Finding the segment boundaries is a simple matter of finding the peaks in the novelty score. A simple approach is to find points where the score exceeds a

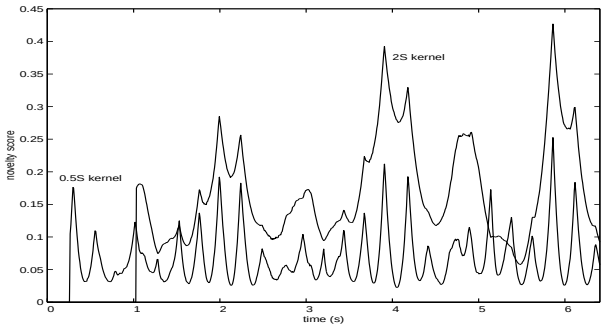

Figure 4. Novelty score for Gould performance and different kernel widths

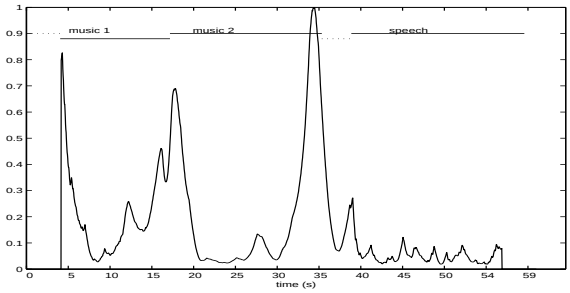

Figure 5. Novelty score for video soundtrack

local or global threshold. The approach used here is to locate the maximum wherever the score exceeds the threshold for maximum time precision. The score is normalized such that the maximum score is one and the minimum is zero.

A useful way of organizing the index points is in a binary tree structure constructed by ranking all the index points by novelty score. Figure 2 shows the segment boundaries extracted from the 1/2 second kernel results of on the first 10 seconds of the Gould performance. Boundaries are indicated by $\oplus$. Individual notes are clearly isolated, save for the fourth note which has been slurred with the third.

D. Speech and Music Segmentation

Besides music, these methods work for segmenting audio into speech and music regions. Figure 5 shows the audio novelty for the first minute of *Animals Have Young* (video V14 from the MPEG-7 content set [13]). This segment contains 4 seconds of introductory silence, followed by a short musical segment with the production logo. At 17 seconds the titles start, and very different theme music commences. At 35 seconds, this fades into a short silence, followed by female speech over attenuated background music for the remainder of the segment. Figure 5 shows the similarity score computed over a 8-second window, which is long enough to average out the short-time spectral differences in speech. The largest peak occurs directly on the speech/music transition at 35 seconds. The two other major peaks occur at the transitions between silence and music at 4 seconds and between the introduction and theme music at 17 seconds.

Segmentation by novelty score works well for musical notes and phases, as well as spoken phrases. Though difficult to qualitatively evaluate (or present in a short paper), experiments using a variety of audio, such as TV dramas, yielded subjectively satisfying segmentation results. Even the soundtrack of a Warner Brothers "Looney Tunes" cartoon, containing an near-pathological sequence of orchestral sounds, sound effects, and Mel Blanc vocalizations, was segmented with reasonable success. This segmentation method can't, however, be expected to segment speech into words unless they are spectrally different. This is because words are often not well-delineated acoustically, for example, the phrase "that's Steven" would be segmented into "that's-S" and "teven" because there is little acoustic differences in the "s" sounds of the two words, even with a glottal stop. Note that this would likely be the segmentation that a non-English speaker would choose.

III. Applications

The ability to reliably segment audio has a large number of useful applications. Note that this approach can be used for *any* time-dependent media, such as video, where some measure of point-to-point similarity can be determined.

Audio segmentation and indexing

As demonstrated above, these methods give good estimates of audio segment boundaries. This is useful for applications where one might wish to access only a portion of an audio file. For example, in an audio editing tool, selection operations can be constrained to segment boundaries so that the selection region does not contain fractional notes or spoken phrases. This would be similar to "smart cut and paste" in a text editor that constrains selection regions to entire units such as words or sentences. The segment size can be adjusted to the degree of zoom so that the appropriate time resolution is available. When zoomed in, higher resolution (from smaller kernels) would allow note-by-note selection, while a zoomed-out view would allow selection by phrase or section. Similarly, segmenting audio greatly facilitates audio browsing: a "jump-to-next-segment" function allows audio to be browsed more rapidly than real-time. Because segments will be reasonably self-similar, listening to a small portion will give a good idea of the entire segment. This would be especially appropriate when combined with a video shot detection system: shots having to have a significant audio novelty as well as video difference are more likely to be meaningful transitions. Another application might be to play back an audio piece synchronized with unpredictably timed events (such as progress through a video game). Longer segments could be associated with particular stages, such as a game level or virtual environment location. As long as the user stayed at that stage, the segment would be looped. Moving to a different stage would cause another segment to start playing.

Audio summarization and gisting

This approach can be extended to automatic audio summarization, for example, by playing only the start of each segment as in the "scan" feature on a CD player. In fact, segments can be clustered so that only significantly novel segments are included in the summary. Segments too similar to a segment already in the summary could be skipped without losing too much information. For example, when summarizing a popular song, repeated instances of the chorus could be excluded from the summary as they would be redundant. Reliably segmenting music by note or phrase allows substantial compression. For example, a repeated series of notes can be represented by the first note and the repetition times. The MPEG-4 structured audio standard supports exactly this kind of representation, but heretofore there have been few reliable methods to analyze the structure of existing audio.

IV. Related Work

Much prior work in audio segmentation has been based on detecting significant silences [3]. Though this works satisfactorily for clean speech, much common audio, such as popular music or reverberant sources, may contain no silences at all. The difference between a running spectral average and a new spectral window is used to find "audio cuts" in [4], though no results are presented to indicate how well this works. Another approach uses speaker identification to segment audio by speaker turns [5]. Though the latter approach could be used to segment music, it relies on statistical models that must be trained from a corpus of labeled data, or estimated by clustering audio segments [6]. Another approach segments audio using heuristically-determined thresholds on features such as zero-crossing rate [8,9]. A more robust approach trains Gaussian models on particular acoustic features [10,11]. The approach presented here is significantly different as it works for any audio source regardless of complexity, does not rely on particular acoustic features such as silence, pitch, or zero-crossing rate, and needs no clustering, modeling, or training.

References

[1] Slaney, M, "Auditory toolbox," Technical Report #1998-010, Interval Research Corporation, Palo Alto, CA, 1998

[2] Foote, J. (1999). "Visualizing music and audio using self-similarity." In *Proc. ACM Multimedia 99*, Orlando, FL, pp 77-80.

[3] Arons, B. "SpeechSkimmer: A system for interactively skimming recorded speech." In *ACM Trans. on Computer Human Interaction*, **4**(1):3-38, March 1997.

[4] Lienhart, R., Pfeiffer, S., and Effelsberg., W. "Scene determination based on video and audio features." In *Proc. IEEE ICMCS*, Vol. I, June 1999

[5] Kimber, D., and L. Wilcox, L., "Acoustic segmentation for audio browsers," in *Proc. Interface Conference*. Sydney, Australia, 1996.

[6] Gish et al., "Segregation of speakers for speech recognition and speaker identification," *Proc. ICASSP*, May 1991, vol. 2 pp. 873-876.

[7] Scheirer, E. D.,"Pulse tracking with a pitch tracker." *Proc. 1997 Workshop on Applications of Signal Processing to Audio and Acoustics*, Mohonk, NY, IEEE.

[8] Saunders, J. "Real-time discrimination of broadcast speech/ music." In *Proc. ICASSP '96*, volume II, pp. 993-996, Atlanta, May 1996. IEEE.

[9] Zhang, T., and C.-C. Jay Kuo, "Heuristic approach for generic audio data segmentation and annotation," In *Proc. ACM Multimedia 99*, Orlando, FL, pp 67-76.

[10] Wold, E., Blum, T., Keslar, D., and Wheaton, J., "Content-based classification, search, and retrieval of audio." *IEEE Multimedia,* pages 27-36, Fall 1996.

[11] Scheirer, E. and Slaney, M. "Construction and evaluation of a robust multifeature music/speech discriminator." In *Proc. ICASSP 97*, volume II, pages 1331-1334. IEEE, April 1997.

[12] Foote, J. (1997). "Content-based retrieval of music and audio." In *Multimedia Storage and Archiving Systems II, Proc. SPIE,* Vol. 3229, Dallas, TX.

[13] MPEG Requirements Group, "Description of MPEG-7 content set," Doc. ISO/MPEG N2467, MPEG Atlantic City Meeting, October 1998