

VIDEO SCENE SEGMENTATION USING VIDEO AND AUDIO FEATURES

Hari Sundaram Shih-Fu Chang

Dept. Of Electrical Engineering, Columbia University,
New York, New York 10027.

Email: {sundaram, sfchang}@ctr.columbia.edu

ABSTRACT

In this paper we present a novel algorithm for video scene segmentation. We model a scene as a semantically consistent chunk of audio-visual data. Central to the segmentation framework is the idea of a finite-memory model. We separately segment the audio and video data into scenes, using data in the memory. The audio segmentation algorithm determines the correlations amongst the envelopes of audio features. The video segmentation algorithm determines the correlations amongst shot key-frames. The scene boundaries in both cases are determined using local correlation minima. Then, we fuse the resulting segments using a nearest neighbor algorithm that is further refined using a time-alignment distribution derived from the ground truth. The algorithm was tested on a difficult data set; the first hour of a commercial film with good results. It achieves a scene segmentation accuracy of 84%.

1. INTRODUCTION

This paper deals with the problem of segmenting video data into semantically coherent scenes using audio and video data. This is an important problem for several reasons: (a) the determination of semantically coherent scenes is the first step towards semantic understanding of the entire video (b) breaking up a long video into scenes will allow for non-linear navigation of video data. For example, scene segmentation will be useful for browsing news programs.

Prior work on video scene segmentation has focused on scene cut detection using image data alone [2,8]. In [8], the authors use scene transition graphs to determine scene boundaries. Their method assumes a repetitive shot structure within a scene. While this structure is present in particular scenes such as interviews, it can be absent from many scenes in commercial films. This can happen, for example, when the director relies on fast succession of shots to heighten suspense.

There has been prior work done dealing with the problem of audio segmentation [5,6,9]. In these papers, the authors use short-term (100 ms) changes in a few features (e.g. energy, cepstra) to classify the audio data into several predefined classes such as speech, music environmental sounds etc. They do not deal with notion of audio-scenes, which are necessarily long and exhibit long-term consistency. Audio data has been used for identifying important regions [1] or detecting events such as explosions [3] in video skims. These skims do not segment the video data into scenes; the objective is to obtain a compact representation.

In this paper, we develop a joint audio-visual framework for video scene segmentation using insights from the ground truth. We define a scene as a chunk of audio-visual data that possesses consistent, long-term audio and visual properties. Audio and video scenes are defined in a similar fashion. We develop a causal memory model based in part on the model in [2]. The model has two parameters: (a) an analysis window that stores the most recent data (the attention span) (b) the total amount of data (memory).

For segmenting the data into audio scenes, we compute correlations amongst the envelopes of the features in the attention-span with feature envelopes in the rest of the memory. The video data comprises shot key-frames. The key-frames in the attention span are compared to the rest of the data in the memory to determine a coherence value that is derived from a color-histogram dissimilarity. This comparison takes also into account the relative shot length and the time separation between the two shots. In both cases, we use a local minima for detecting a scene change. The audio and video scene boundaries are initially aligned using a simple time-constrained nearest neighbor approach. This alignment result is then refined using the distribution of time alignment differences yielding good results.

The rest of the paper is organized as follows. In the next section we formally define the characteristics of a scene. Sections 3 and 4 deal with audio and video scene segmentations respectively. Section 5 discusses ways to fuse results of the two segmentations. This is followed by a section on experimental results. Finally, we present our conclusions in section 7.

2. WHAT IS A SCENE?

In this section we present the overall framework for scene segmentation.

2.1 Insights from the Ground Truth

The ground truth data is obtained from the first one hour of a classic science fiction film: *Blade Runner*. The data is complex with non-trivial audio and video changes. Consider for example, a typical sequence of audio changes: ambient music → street sounds → conversation → sounds in a bar.

The first hour of the film was hand labeled into coherent, semantically consistent scenes in two ways: by looking at the video along with the audio (scenes) and by listening to the audio alone (audio scene). Table 1 shows a strong agreement (i.e. they can be cross-validated) between two kinds of labeled data.

Table 1: The audio scene breaks were labeled *without* watching the video while the scene breaks were obtained by watching the film with the audio.

| Type | No. |
|-----------------|-----|
| Video Scenes | 28 |
| Audio Scenes | 33 |
| Scene Agreement | 24 |

The labeled data (Table 1) seem to imply that there are 9 “extra” sound scenes. These are audio scene changes within the same video scene. They reflect a change of mood (or theme) within the same video scene. While there is a clear visual change in the semantic in the four video scenes that disagree, they are either parenthesized by silence or are transitory scenes with no break in the semantics of the audio.

2.2 Formalizing the idea of a Scene

We model the *audio-scene* as a collection of a few dominant sound sources. These sources are assumed to possess stationary properties that can be characterized using a few features. An audio-scene change is said to occur when the majority of the dominant sources in the sound change. We model the *video-scene* as a collection of shots that have a single, consistent (within the scene), underlying semantic. We further assume that the majority of shots in the scene associated with the semantic are chromatically *coherent*. A video-scene is said to occur when the majority of shots (that are chromatically consistent) change.

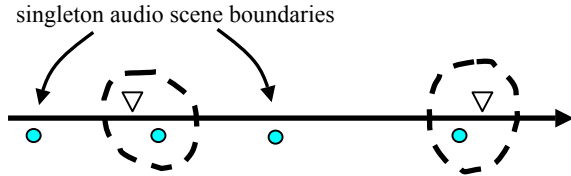


Figure 1: The figure shows video (triangles) and audio (solid circles) scene change locations. The dashed circles show audio/video scene boundaries which agree.

A *scene* is a contiguous segment of data having consistent long term audio and video characteristics. The *scene* boundaries are determined by aligning (with time constraints) audio-scene boundaries with video-scene boundaries. Audio and video boundaries that do not agree are labeled as *singleton* changes. The ground truth data implies that singleton audio scene changes are indicative of possible theme (mood) changes and hence such changes are denoted as “interesting” audio events within the scene. Singleton video changes appear unlikely (prob. 4/28).

2.3 The Memory Model

In order to segment data into scenes, we use a causal, first-in-first-out (FIFO) model of memory (figure 2). This model is derived in part from the idea of coherence [2]. In [2] the authors use a non-causal, infinite memory model. In our model of a listener, two parameters are of interest: (a) memory: this is the net amount of information (T_m) with the viewer and (b) attention span: it is the most recent data (T_{as}) in with the memory of the listener. The listener uses this data to compare against the contents of the memory to decide if a scene change has occurred.

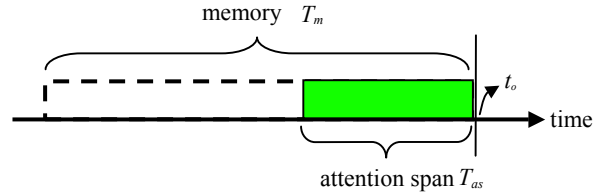


Figure 2: The attention span (T_{as}) is the most recent data in the buffer. The memory (T_m) is the size of the entire buffer. Clearly, $T_m \geq T_{as}$.

3. AUDIO SCENE CHANGE DETECTION

In this section we present our algorithm for audio-scene segmentation. A more detailed description of audio scene segmentation is to be found in [7].

3.1 Features and Envelope Models

We use ten different features [5,6,7,9] in our algorithm (a) cepstral-flux (b) multi-channel cochlear decomposition (c) cepstral vectors (d) low energy fraction (e) zero crossing rate (f) spectral flux (g) energy and (h) spectral roll off point. We also use the variance of the zero crossing rate and the variance of the energy as additional features. The cochlear decomposition was used because it was based on a psychophysical ear model. The cepstral features are known to be good discriminators [4]. All the other features were used for their ability to distinguish between speech and music [5] [6]. Features are extracted per *frame* (100ms. duration) for the duration of the analysis window.

Given a particular feature f and a finite time-sequence of values, we wish to determine the behavior of the envelope of the feature. The feature envelopes are force-fit into signals of the following types: constant, linear, quadratic, exponential, hyperbolic and sum of exponentials. All the envelope (save for the sum of exponentials case) fits are obtained using a robust curve fitting procedure. We pick the fit that minimizes the least median error. The envelope model analysis is only used for the scalar variables. The vector variables (cepstra and the cochlear output) and the aggregate variables (variance of the zero-crossing rate and the spectral roll off point) are used in the raw form.

3.2 Detecting a Scene Change

Let us examine the case where a scene change occurs just to the left of the listeners attention span. First, for each feature, we do the following:

1. Place an analysis window of length T_{as} (the attention-span length) at t_o and generate a sequence by computing a feature value for each frame (100 ms duration) in the window.
2. Determine the optimal envelope fit for these feature values.
3. Shift the analysis window back by Δt and repeat steps 1. and 2. till we have covered all data in the memory.

We then define a local correlation function per feature, using the sequence of envelope fits. The correlation function C_f for each feature f is then defined as follows:

$$C_f(m\delta) = 1 - d(f(t_o, t_o - t_{as}), f(t_o + m\delta, t_o + m\delta - t_{as})) \quad (1)$$

where, $f(t_i, t_j)$ represents the envelope fit for feature f for the duration $[t_1, t_2]$. $m \in [-N, 0]$, where $N \equiv (T_m - T_{as})/\delta$. The analysis window shifts back by δ and d is the Euclidean metric¹ on the envelopes. For the vector and the aggregate data, we do not compute the distance between the windows using envelope fits but use a L^2 metric on the raw data. In our experiments we use $\delta = 1$ sec.

We model [7] the correlation function as a decaying exponential: $C_i(t) = \exp(b_i t)$, $t < 0$ where C_i is the correlation function for feature i , and b_i is the exponential decay parameter. The audio-scene decision function $D(t_o)$ at any instant t_o is defined as follows: $D(t_o) = \sum_i b_i$.

The audio-scene change is detected using the local minima of the decision function. This is done by using a sliding window of length $2w_a + 1$ sec. to slide across the data. We then determine whether the minima in the window coincides with the center of the window. If it does, the location is labeled as an audio scene change location.

4. VIDEO SCENE CHANGE DETECTION

In this section, we shall describe the algorithm for video-scene segmentation. We also develop notions of *recall* and *coherence*.

4.1 Recall

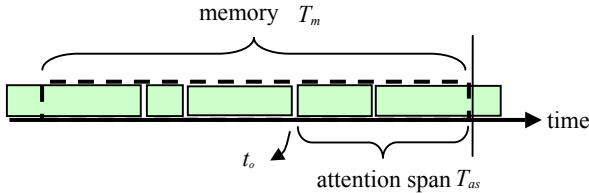


Figure 3: Each of the solid blocks represents a single shot. Note that the most recent shot and the oldest shot may be partially present in the buffer.

In our visual memory model, the data is in the form of key-frames of shots (figure 3). The model also allows for the most recent and the oldest shots to be partially present in the buffer. A point in time (t_o) is defined to be a scene transition boundary if the shots that come after that point in time, do not recall [2] the shots prior to that point. The idea of recall between two shots a and b is formalized as follows:

$$R(a, b) = (1 - d(a, b)) \cdot f_a \cdot f_b \cdot (1 - \Delta t / T_m), \quad (2)$$

where, $R(a, b)$ is the recall between the two shots a, b . $d(a, b)$ is a color-histogram distance between the key-frames corresponding to the two shots, f_i is the ratio of the length of shot i to the buffer size (T_m). Δt is the time difference between the two shots.

The formula for recall indicates that recall is proportional to the length of each of the shots. This is intuitive since if a shot is in memory for a long period of time it will be recalled more

¹ This metric is intuitive: it is a point-by-point comparison of the two envelopes.

easily. Again, the recall between the two shots should decrease if they are further apart in time.

4.2 Computing Coherence

Coherence is easily defined using the definition of recall:

$$C(t_o) = \left(\sum_{\forall a \in T_m, b \in \{T_m \setminus T_{as}\}} R(a, b) \right) / C_{\max}(t_o) \quad (3)$$

where, $C(t_o)$ is the coherence across the boundary at t_o and is just the sum of recall values between all pairs of shots across the boundary at t_o . $C_{\max}(t_o)$ is obtained by setting $d(a, b) = 0$ in the formula for recall. This normalization compensates for the different number of shots in the buffer at different instants of time. Our formulation simplifies the model found in [2].

We compute coherence at the boundary between every adjacent pair of key-frames. Then, similar to the procedure for audio scene detection, we determine the local minima. This is done by using a sliding window of length $2k + 1$ frames and determine if the minima in the window coincides with the center of the window. If it does, the location is labeled as a video scene change location.

5. MERGING THE AUDIO AND VIDEO SCENES

We generate correspondences between the audio and the video scene boundaries using a simple time-constrained nearest-neighbor algorithm. Let the list of video scene boundaries be $V, i \in \{1..N_v\}$. Let the list of audio scene boundaries be $A, i \in \{1..N_a\}$. The ambiguity window around each video scene is k frames wide. The ambiguity window width around each audio scene boundary is w_a sec long. Note that these sizes are the same size of the windows used for local minima location. For each video scene boundary, do the following:

- Determine a list of audio scene boundaries whose ambiguity windows intersect the ambiguity window of the current video scene boundary.
- If the intersection is non-null, pick the audio scene boundary closest to the current video scene boundary. Remove the this audio scene boundary from the list containing audio scene boundaries.
- If the intersection is null, add the current video scene boundary to the list of singleton video scene changes.

At the end of this procedure, if there are audio scene boundaries left, collect them and add them to the list of singleton audio scene changes. The nearest neighbor algorithm can be improved by determining the probability distribution of the lags of the audio scene changes with respect to the video scene changes. This distribution is determined using the ground-truth scenes that are in agreement. We use this lag distribution to assign a confidence score to the audio lag. This is done for each of the joint audio-video scene changes obtained using the nearest-neighbor rule. We eliminate all joint audio-video scenes that have a lag confidence score less than ϵ .

6. EXPERIMENTS

In this section we present experimental results on the data set using our audio and video scene change algorithms. The data set used to test our algorithms is complex; it is the first hour of a classic science fiction film: *Blade Runner*.

There are three parameters of interest in each scene change algorithm (i.e. audio and video). They are: (a) memory (T_m) (b) attention-span (T_{as}) (c) ambiguity-window size. For both audio and video scene change algorithms, the attention-span and the memory parameters follow intuition: results improve with a large attention-span and a large memory. Large windows have the property of smoothing the audio decision function [7] and the video coherence. The audio and video ambiguity parameters are used in the location of local minima in both scene change algorithms. The same parameters are used as time-constraints when aligning the two scene boundaries. Larger windows decrease the number of false alarms but also increase the number of misses.

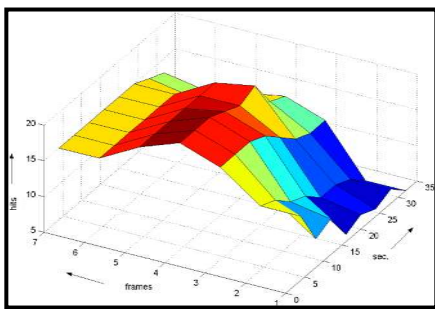


Figure 5: Plot of number of correct matches (hits) in the naïve case, against a variation in the video (in frames, x-axis) and audio ambiguity (in sec., y-axis) windows. The best result is 20/24 matches; video ambiguity: 4 frames, audio ambiguity: 4 sec.

In the figure 5, we show the number of correct matches against a variation in the ambiguity window sizes with other parameters fixed. The memory buffer parameters have been fixed: audio: $T_m=31$ sec. $T_{as}=16$ sec., video: $T_m=32$ sec., $T_{as}=16$ sec. The video ambiguity is units of frames, while the audio ambiguity window is in seconds. The maximum number of possible matches is 24. The plot shows the nearest-neighbor case. The best result is obtained for video ambiguity of 4 frames, audio ambiguity of 4 sec: 20/24 correctly matched. This gives a detection accuracy of 84%. Using the probability distribution refinement, for the same parameters, the accuracy drops slightly to 18/24. This is because we have small training set (size 24). A small value of epsilon ($\epsilon = 0.01$) caused the two correct matches to be missed. A much larger training set will reduce the probability of misses. There were 220 singleton audio events in both cases. The results can be improved, but these results seem all the better because the shot detection algorithm had misses and false alarms.

7. CONCLUSIONS

In this paper we have presented a framework of segmentation of audio-visual data into semantically consistent scenes. We begin by defining a scene to be a semantically consistent chunk of audio-visual data. This idea is used in conjunction with a memory model with two parameters: (a) the

attention span (b) total memory. We segment the audio and the video data separately and subsequently merge the results to determine scene boundaries.

In order to determine audio scene segments we first determine optimal envelope fits for each feature extracted in the memory buffer. We then determine the correlation amongst the envelope fits. The video segmentation algorithm determines the coherence amongst the key-frames of the shots in the memory. The coherence between two key-frames is proportional to the length of the each of the two shots as well as incorporates the time difference between the two shots. A local minima criterion is used to determine the video and audio segmentation boundaries. To determine the scene segmentation, the two segmentations are then merged using a simple time-constrained nearest neighbor scheme. This is further refined using a lag probability distribution.

The segmentation algorithm achieves an accuracy of 20/24 scenes (84%) in the naïve case and 18/24 (75%) using the refinement. There were 220 singleton audio events. The advantage of a scene model with singleton events is that in addition to browsing the video by scenes, navigation within a scene can take place using the audio events. While the results leave much scope for improvement, we believe that the results are very good when the complexity of the data set (a one hour segment of a film) is kept in mind.

There are additional improvements possible (a) instead of segmenting the audio and the video separately, an algorithm to directly segment the data using both the audio and video data simultaneously (b) a more sophisticated memory model (as opposed to a FIFO) that assigns different probabilities of removal to different shots.

8. REFERENCES

- [1] A. Hauptmann M. Witbrock *Story Segmentation and Detection of Commercials in Broadcast News Video* Advances in Digital Libraries Conference, ADL-98, Santa Barbara, CA., Apr. 22-24, 1998.
- [2] J.R. Kender B.L. Yeo, *Video Scene Segmentation Via Continuous Video Coherence*, CVPR '98, Santa Barbara CA, Jun. 1998.
- [3] R. Lienhart et. al. *Automatic Movie Abstracting*. Technical Report TR-97-003, Praktische Informatik IV, University of Mannheim, Jul. 1997
- [4] L. R. Rabiner B.H. Huang *Fundamentals of Speech Recognition*, Prentice-Hall 1993.
- [5] Eric Scheirer Malcom Slaney *Construction and Evaluation of a Robust Multifeature Speech/Music Discriminator* Proc. ICASSP '97, Munich, Germany Apr. 1997.
- [6] S. Subramaniam et. al. *Towards Robust Features for Classifying Audio in the CueVideo System*, Proc. ACM Multimedia '99, pp. 393-400, Orlando FL, Nov. 1999.
- [7] H. Sundaram S.F. Chang *Audio Scene Segmentation Using Multiple Features, Models And Time Scales*, to appear in ICASSP 2000, Istanbul Turkey, Jun. 2000.
- [8] M. Yeung B.L. Yeo *Time-Constrained Clustering for Segmentation of Video into Story Units*, Proc. Int. Conf. on Pattern Recognition, ICPR '96, Vol. C pp. 375-380, Vienna Austria, Aug. 1996.
- [9] T. Zhang C.C Jay Kuo *Heuristic Approach for Generic Audio Segmentation and Annotation*, Proc. ACM Multimedia '99, pp. 67-76, Orlando FL, Nov. 1999.