A General Framework for Temporal Video Scene Segmentation

Yun Zhai
School of Computer Science
University of Central Florida
yzhai@cs.ucf.edu

Mubarak Shah
School of Computer Science
University of Central Florida
shah@cs.ucf.edu

Abstract

Videos are composed of many shots caused by different camera operations, e.g., on/off operations and switching between cameras. One important goal in video analysis is to group the shots into temporal scenes, such that all the shots in a single scene are related to a particular physical setting, an on-going action or a theme. In this paper, we present a general framework for temporal scene segmentation for various video types. The proposed method is formulated in a statistical fashion and uses the Markov chain Monte Carlo (MCMC) technique to determine the boundaries between video scenes. In this approach, an arbitrary number of scene boundaries are randomly initialized and automatically updated using two types of updates: diffuse and jumps. The posterior probability on the number of scenes and their boundary locations is computed based on the model priors and the data likelihood. The updates of the model parameters are controlled by the hypothesis ratio test in the MCMC process. The proposed framework has been experimented on two types of videos, home videos and feature films, and accurate results have been obtained.

1. Introduction

Temporal scene segmentation is an important and fundamental problem in video processing and understanding and has many applications in various domains. In feature films, segmentation provides the chapters of the movie. In TV programs, segmentation can be used to separate the commercials from the regular TV shows. In news broadcast, it provides the news story topics, which can be further used for the story summarization. Videos are usually composed of multiple shots caused by camera operations, e.g., turning the cameras on/off, the switching between cameras, and other production techniques. The goal is to cluster the video shots into temporal scenes, such that the shots in each scene are related in terms of similar concepts, like the same environmental settings, the same on-going actions, or the coherent themes. The temporal segmentation is a prerequisite to the further analysis and understanding of the video content.

Several temporal segmentation methods have been developed for different types of videos. Rasheed *et al.* [5] proposed a two-pass algorithm for the scene segmentation in feature films and TV shows. The potential video scene boundaries are initially detected based on the color-similarity feature, *Backward Shot Coherence (BSC)*. Over-segmented scenes from the first pass are then merged in the second pass, based on the motion similarity constraint. Sundaram *et al.* [6] proposed a segmentation method for movies using the audio-visual features. First, audio scenes and video scenes are detected separately. The correspondences between these two types of scenes are then determined using a time-constrained nearest-neighbor algorithm. Yeung *et al.* [7] was one of the first to propose a graph-based representation of the video by constructing a Shot Connectivity Graph (SCG). The graph is split into several sub-graphs using the complete-link method of hierarchical clustering, such that each sub-graph satisfies a color similarity constraint. These methods are based on the "film/TV grammars", which are a set of production rules of how the movies/TV shows should be composed. However, this heuristic is not applicable to the other types of videos. For instance, home videos are recorded in a completely "free" style. Shooters are not trained in recording techniques, and there is no obvious format or patterns exist in the video. Furthermore, since the rules in the productions of films and TV shows are different, the methods for these two types of videos cannot be used interchangeably. There is also a particular interest in the story segmentation of the news broadcast videos. Hsu *et al.* [4] proposed a statistical approach based on *BoostME*, which uses the Maximum Entropy classifiers and the associated confidence scores in each boosting iteration. Chaisorn *et al.* [1] used HMM to find the story boundaries. The video shots are first classified into different categories. The HMM contains four states and is trained on three features: type of the shot, whether location changes and whether speaker changes. These methods were developed based on the unique characteristics of the news videos and often involve the special treatment of the anchor shots, which exist only in news videos.

In this paper, we propose a general framework for the temporal video segmentation by using Markov chain Monte Carlo (MCMC) technique. The segmentation parameters, including the number of the scenes and the scene locations are evaluated by an iterative process. The scene boundaries are randomly initialized and modified through a series of updates of the Markov chain: shifting of boundaries, merging of two adjacent scenes and the splitting of one scene into two scenes. These updates are capable of jumping between different parameters spaces as well as diffusing inside the same space. Visual features are used for the likelihood computation for two applications, and the final output of the scene boundary locations is collected from multiple independent Markov chains, such that the possible misdetections by a single chain are avoided. We have tested our framework on two domains, home videos and feature films, and high accurate results have been obtained. The rest of this paper is organized as follows: Section 2 describes the MCMC algorithm and the computations of its transition probability and the posterior probability; Sections 3 and 4 explain the applications of the general framework on the segmentation of home videos and feature films, respectively, and demonstrates their system performance; finally, Section 5 provides the conclusions of the proposed work.

2. Proposed Framework

Given the shots in the video, scene segmentation of the videos is a process of grouping the related shots into clusters. Within each scene, the shots are related to each other by the *central concept*. The *central concept* can be the environmental settings (home videos), the coherence of some topics (news program), or the sub-themes (feature films). Different scenes are distinguished by their difference in terms of the *central concept*, and the scene boundaries are the locations where the intrinsic properties of the *central concept* change. Thus, finding the scene boundaries in scene segmentation process can be thought as solving a change-point problem. In a typical change-point problem, the random process has different distribution parameters over times. The goal is to find the points where these parameters change. In video scene segmentation, the change-points are the scene boundary locations. Markov chain Monte Carlo (MCMC) has been demonstrated as an efficient technique solving the change-point problems. In our MCMC process, the posterior probability of the model parameters is computed based on the model priors and the data likelihood of the video. In this section, we first introduce the general MCMC algorithm. Then, a detailed description of the proposal updates is presented. Finally, we describe the computations of the transition probability and the posterior probability for the updates.

2.1. General MCMC Algorithm

We use a hierarchical Bayesian model in the MCMC process. We assume that the model set $\{M_k, k \in \Phi\}$ is a countable set, where k is the number of the detected scenes, and $\Phi = \{1, 2, \cdots\}$ is a set of all the possible scene numbers. Model M_k has a parameter vector θ_k, which contains the $k-1$ scene boundary locations. Let y denote the video features selected for the data likelihood computation. Based on the Bayes rule, the posterior probability of the parameter k and θ_k given y is:

$$p(k, \theta_k|y) \propto p(y|k, \theta_k)p(\theta_k|k)p(k), \tag{1}$$

where $p(k)$ is the prior probability for the number of scenes, $p(\theta_k|k)$ is the conditional prior for the boundary locations θ_k given k, and $p(y|k, \theta_k)$ is the likelihood of the data given the parameters k and θ_k. Since θ_k implicitly determines k, the above equation can be further simplified as,

$$p(k, \theta_k|y) \propto p(y|\theta_k)p(\theta_k|k)p(k). \tag{2}$$

In the rest of this paper, we use the shorter term $\pi(x) = p(k, \theta_k|y)$ to denote this target posterior, with $x = \{k, \theta_k\}$ considered as a combined parameter vector of k and θ_k.

The general Metropolis-Hasting-Green algorithm [2] is well suited for our task, where the dimension of the parameter vector, x, may change during the updates. It is described as follows:

- Initialize the model parameters x_0.
- At each iteration i, perform the following actions:
  1. Generate Th_α from $Uni[0, 1]$.
  2. Create a new parameter x'_{i-1} from some trial distribution based only on x_{i-1} with a proposal transition (diffusion or jump).
  3. Calculate the ratio $\alpha(x_{i-1}, x'_{i-1})$ as, $\alpha(x_{i-1}, x'_{i-1}) = min\left\{1, \frac{\pi(x'_{i-1})q(x'_{i-1}, x_{i-1})}{\pi(x_{i-1})q(x_{i-1}, x'_{i-1})}\right\}$.
  4. Set $x_i = x'_{i-1}$, if $\alpha > Th_\alpha$. Otherwise, $x_i = x_{i-1}$.

In this algorithm, $q(x, x')$ is the transition probability from x to x'. The transition probabilities from state to state depend on the types of the updates. They should satisfy the detailed balance, and the proposed updates should be reversible to ensure this property.

Before going into the detailed description of the updating process, we first present the notations for the variables. Let k be the current number of detected scenes, T be the total number of shots in the video, S_m be the m-th scene with shots $\{s_m^1, \cdots, s_m^{n_m}\}$, where n_m is the number of shots in S_m. Let S'_m be the m-th scene after update, $\mathbb{L}(y|\theta_k)$ be the data likelihood of the entire video, $\mathbb{L}(y_m|f_m)$ be the likelihood of scene S_m given the corresponding features f_m, and k_{max} is the maximum number of the scenes allowed.

The proposal updates contain two parts: diffusion and jumps. Diffusion is defined as the update without changing the structure of the parameter vector x. On the other hand, jumps do change the structure and traverse across different sub-spaces. In our case, the diffusion is the shifting of the boundaries between adjacent scenes. There are two types of jumps: the merging of two adjacent scenes and the splitting of an existing scene. In many applications ([3],[2]), two more updates were proposed: diffusion on the segment model parameter(s) and the change of the segment models. The segment parameters are the ones that control the generation of the data. In our application, based on the assumption that each segment is coherent to its *central concept*, there is often only one scene model for a single video domain. Thus, changing between models is not needed in this case. Furthermore, in some cases like home videos, the data size (number of shots in our case) is small. The maximum likelihood estimator is adequately effective to compute the parameter(s). Therefore, the model parameter diffusion steps can also be dropped.

2.2. Stochastic Diffusions

The diffusions involve the shifts of the scene boundaries between adjacent video scenes. The update is as follows:

- A number m is randomly drawn from the discrete uniform distribution $[1, k-1]$, such that the boundary between S_m and S_{m+1} is updated.
- The new boundary s^t is drawn from a 1-D normal distribution with the mean at the original boundary s^1_{m+1} in the range of $[s^1_m,\ s^{n_{m+1}}_{m+1}]$. The updated scene S'_m contains shots of $\{s^1_m, \cdots, s^{t-1}\}$, and the updated scene S'_{m+1} contains $\{s^t, \cdots, s^{n_{m+1}}_{m+1}\}$.

Assume the number of the current scenes is k and the current parameter vector is $x = \{k, \theta_k\}$. Then, the probability for selecting scene S_m is $1/(k-1)$. Since the potential shift is drawn from a normal distribution around the original scene boundary $\hat{t}$, this drawing probability for the new boundary t is computed as,

$$p(t) = \frac{1}{\sqrt{2\pi\sigma^2}} exp(-\frac{\Delta t^2}{2\sigma^2}) \Big(I_{[s^1_m, s^{n_{m+1}}_{m+1}]}(t) \Big), \qquad (3)$$

where $\Delta t = t - \hat{t}$, and σ is the standard deviation of the movement (in our experiment, $\sigma = 2$). The indicator function $I(t)$ controls the shift, such that the new boundary is within the correct range. The normal distribution is assumed since the new boundary is not expected to deviate from the old boundary too far. The forward transition probability for the shift update is $q(x, x') = (\frac{1}{k-1})p(t)$.

During this entire update, the total number of scenes, k, is not changed, and the new boundary remains in the original range $[s^1_m,\ s^{n_{m+1}}_{m+1}]$. The reverse transition is the process of shifting from the new boundary t back to the original boundary $\hat{t}$. Thus, the relationship between $q(x, x')$ and its reverse version $q(x', x)$ is equal due to the symmetrical property of the normal distribution.

2.3. Reversible Jumps: Merge and Split

For the jump updates, the merge transition is related to the transition of a split, since merge and split are a pair of reversed updates. Let us consider the splits first. The number of scenes is increased by 1 by splitting a scene $S_m = \{s^1_m, \cdots, s^{n_m}_m\}$ into two new scenes $S'_m = \{s^1_m, \cdots, t-1\}$ and $S'_{m+1} = \{t, \cdots, s^{n_m}_m\}$, where t is the new boundary. The process contains two portions: selecting a scene S_m and selecting a new boundary between its old boundaries. The selection of the new boundary in the split process can be performed assuming the uniform distributions [2]. However, to achieve better performance, the data-driven technique is often used ([3]) to propose the jump transitions. We assume the uniform probability for selecting scene S_m. The new boundary t is chosen, such that it provides the maximum likelihoods for the two new scenes, $t = arg\ max\Big(\mathbb{L}(S'_m|f'_m) + \mathbb{L}(S'_{m+1}|f'_{m+1})\Big)$, where $\mathbb{L}(S'_m|f'_m)$ and $\mathbb{L}(S'_{m+1}|f'_{m+1})$ are the likelihoods of the new scenes S'_m and S'_{m+1} given their corresponding features. If we consider the video scenes are independent events in the time series, the proposal probability for a split can be expressed in the following form,

$$q(x, x') = \frac{1}{k}\mathbb{L}(S'_m|f'_m)\mathbb{L}(S'_{m+1}|f'_{m+1}). \qquad (4)$$

The reversed update of the split is the merging of two scenes into one. The construction of the proposal probability for the merge can be carried out similarly to the one for the split. Again, we assume the uniform distribution for selecting scene S_m, such that scenes S_m and S_{m+1} are merged into S'_m. The proposal probability for the merge transition is constructed as, $q(x, x') = \frac{1}{k-1}\mathbb{L}(S'_m|f'_m)$.

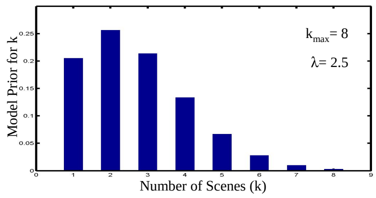


Figure 1. Prior distribution (Poisson) of the model parameter k, the number of scenes in the video. The mean of the distribution, λ, is pre-assigned as 2.5, and k_{max} is 8.

2.4. Posterior Probability

Since Poisson distribution models the number of incidents happening in a unit time interval, we assume the number of scenes, k, is drawn from a such distribution. The prior on k is computed as, $p(k) = e^{-\lambda}\frac{\lambda^k}{k!}I_{[1,k_{max}]}(k)$. A plot of the prior distribution is shown in Fig.1.

If there are k segments (scenes) in the video, then there are $k-1$ scene boundaries, since the boundary for the first scene is always the beginning of the video. The probability of $p(\theta_k|k)$ is the same as the probability of selecting a subset with size $k-1$ from the remaining $T-1$ shots. Therefore, the conditional prior can be defined in terms of the combinations, $p(\theta_k|k) = \frac{1}{C^{T-1}_{k-1}} = \frac{(k-1)!(T-k)!}{(T-1)!}$.

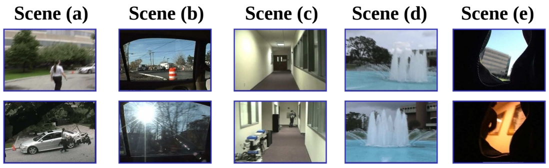


Figure 2. Key-frames of five home video scenes. They cover both indoor and outdoor scenes, and they are recorded by difference sources.

The last term to be computed is the likelihood. Let $\mathbb{L}(y|\theta_k) = p(y|\theta_k)$ denote the global likelihood of the video data y given the parameter vector θ_k. As discussed in Section 2, each scene possesses a different *central concept*. It is meaningful to make an assumption that scenes are independently recorded from each others. Therefore, the overall likelihood can be expressed as,

$$\mathbb{L}(y|\theta_k) = \left(\prod_{m=1}^{L} \mathbb{L}(y_m|f_m) \right)^{\frac{1}{L}}, \tag{5}$$

where $\mathbb{L}(y_m|f_m)$ is the individual likelihood of data y_m in scene S_m, based on the feature values f_m. The geometric mean of the individual likelihoods is considered for the normalization purpose. In order to make the ratio test meaningful, the likelihood should be scaled to the same level during each iteration. The definition of the *central concept* is different across domains. Therefore, the features selected to compute the likelihoods are different for the different types of videos. Here, $\mathbb{L}(y|\theta_k)$ is a general representation of the likelihood rather than a specific computation.

The target posterior probability is proportional to the product of the model prior $p(k)$, the conditional prior $p(\theta_k|k)$, and the data likelihood $\mathbb{L}(y|\theta_k)$ (Eq.2). To determine if the proposed update in the parameter space is accepted or rejected, we compute the ratio of the two terms: $\pi(x')q(x', x)$ and $\pi(x)q(x, x')$. If the ratio, $\alpha(x, x')$, satisfies the stochastically generated threshold, the proposed update is accepted; otherwise, the model parameters are kept the same as in the previous iteration.

3. Application on Home Videos

In this section and the following section, we demonstrate the applications of the MCMC algorithm on two types of videos, home videos and feature films, and present the system performance in each of the applications. In this paper, the video shots are assumed to be available.

Home video is a broad term that refers to the videos with a “free-style”, e.g., family videos. They are recorded from various sources and have different appearances. Some example key-frames of the home videos are shown in Figure 2. Temporal scene segmentation in home videos provides the logical units of the interested locations or actions, and the output can be used for further analysis and processing on the videos. Applying fixed thresholds is likely to create either under-segmentation or over-segmentation, and it is not practical to train the system for the threshold selection. On the other hand, MCMC process provides the solution by analyzing the samples on the parameters and is able to detect both the weak and strong boundaries.

3.1. Feature Selection

In the context of temporal scene segmentation, a variety of features have been exploited. The commonly used features include color, motion content, shot length, etc. In home videos, we have focused on the analysis of the color information in the shots. We use the color histograms to represent the color information of the video frames. The color histogram for each frame is the 3-dimensional histogram for RGB channels with 8 bins in each dimension. Let h_i be the histogram for frame f_i. Furthermore, we define the histogram intersection between frames f_i and f_j as, $HistInter(f_i, f_j) = \sum_{b \in Allbins} min(h_i^b, h_j^b)$, where b is the individual bin in the histogram.

Instead of using all the frames in the shot, we extract the key-frames as the representation of the shot, and further analysis is based on the key-frames only. Commonly there is one key-frame selected for each shot. However, for the shots with long durations and with high activity content, multiple key-frames form better representation. For shot s, the key-frame set $\mathbb{K}_s = \{\kappa_s^1, \cdots, \kappa_s^m\}$ is extracted using [5]. We use the histograms for the keyframe representation.

3.2. Likelihood Computation

For home videos, usually the shots in one scene are coherent with respect to the same environment. There are visual similarities that exist among these shots. On the other hand, the shots from different scenes are visually distinctive. We define the visual similarity between two shots in terms of the Bhattacharya distance. The Bhattacharya distance between two histograms h_1 and h_2 is defined as $d_B(h_1, h_2) = -ln\Big(\sum_{b \in allbins} \sqrt{h_1^b h_2^b} \Big)$. The visual similarity between shots s_i and s_j is as follows:

$$Sim(s_i, s_j) = max(\mathbb{C} - d_B(\kappa_{s_i}^m, \kappa_{s_j}^n)), \tag{6}$$

where $\kappa_{s_i}^m \in \mathbb{K}_{s_i}$, $\kappa_{s_j}^n \in \mathbb{K}_{s_j}$, and $\mathbb{C}$ is a constant. A similarity map is generated by computing the similarities between all shot pairs (Figure 3). In this map, the brighter cell represents higher similarity value. The shots that are in the same temporal scene form a bright block along the diagonal. If the shots $[s_a, ..., s_b]$ are clustered into scene S_m, the likelihood for this scene is computed as:

$$\mathbb{L}(y_m|f_m) = avg\Big(\mathbb{M}(a:b, a:b)\Big). \tag{7}$$

It is intuitive that the correct segmentation of the video gives the diagonal blocks to reach the maximum likelihood. The overall likelihood can be computed by substituting Eq.7 into Eq.5. The overall likelihood $\mathbb{L}(y|\theta_k)$, the conditional prior $p(\theta_k|k)$ and the model prior $p(k)$ are now determined.

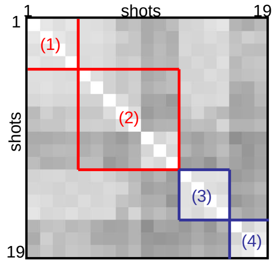


Figure 3. Visual similarity map of a testing video. The brighter cell represents higher similarity. The shots in the same scene possess higher similarity comparing across scenes. The diagonal blocks correspond to the video scenes.

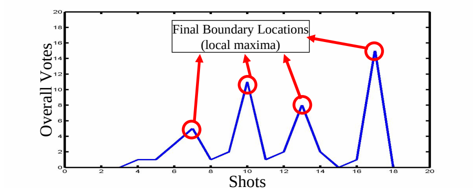


Figure 4. The overall votes of the shots declared as the scene boundaries. The red circles represent the shots that are declared as the final boundaries, which correspond to the local maxima in the plot.

Therefore, the acceptance for the proposal updates can be decided by the ratio test described in the MCMC algorithm.

3.3. System Performance

The proposed method has been tested on four home videos. Each scene is composed of several shots. They cover both of the indoor scenes and outdoor scenes, and they were taken in different ways, like hand-held, pin-hole and vehicle-mounted cameras (Figure 2).

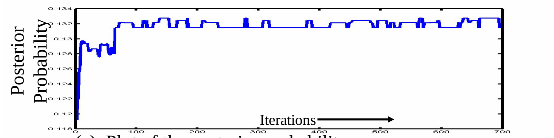
It is well known that a single Markov chain may not result in the true boundary. To overcome this problem, we independently executed multiple Markov chains. The results from each individual chain provides the votes of the shots that have been declared as the scene boundaries. After certain runs, the shots that have the locally highest votes represent the final scene boundary. Figure 4 shows the overall votes of the shots being declared as scene boundaries from all runs. Even though one single chain may not provide the correct result, there is an issue of how quickly it converges. This is referred as the “burn-in” period. As shown in Figure 5, after certain iterations, the posterior probability reaches a level and starts mixing with the target distribution. For this particular testing video, the “burn-in” time is short due to the small size of the data (number of shots).

The match between the ground truth data and the segmented scenes are based on the match of their starting boundaries. For a given home video with n scenes, $\{t_1, t_2, ..., t_n\}$ are the starting shots of the reference scenes, and $\{s_1, s_2, ..., s_k\}$ denote the starting shots of the detected scenes. t_i is declared as matched if one or more of the detected scene boundaries s_j falls in its evaluation interval. In the experiments, we allow a window of 1 shot on each side of the reference boundary in the matching process.

Two accuracy measures are used as the system performance, precision and recall:

$$Precision = X/A, \quad Recall = X/B, \tag{8}$$

where X is the number of correctly matches between system detections and the ground truth data; A is the total number of the system detections; B is the total number of the ground truth references. The detailed precision/recall measures are shown in Table 1. Treating every match equally important in all the videos, the overall precision and recall are 0.840 and 0.913, respectively.



(a). Plot of the posterior probability

Figure 5. (a). The plot of the posterior probability for the model parameter estimation during a Markov chain. After certain iterations, the process starts converging. It should be noted that if the data size (number of shots in our application) is small, this “burn-in” process is short.

Table 1. Accuracy measures of four home videos.

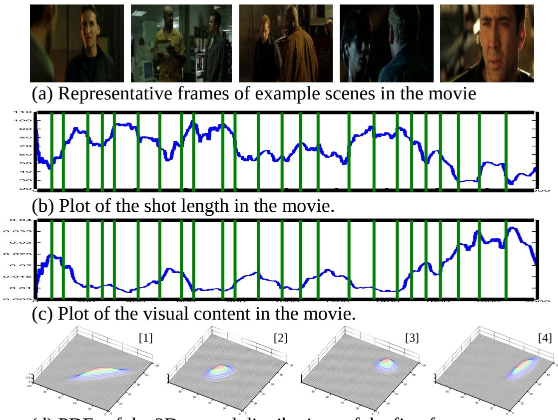
Measures	*clip1*	*clip2*	*clip3*	*clip4*
Length	12:42	06:53	07:31	17:53
Num. of Shot	47	16	19	25
Num. of Scenes	8	5	5	5
Detected Scenes	8	5	5	7
Match	7	5	5	4
Precision	0.875	1.000	1.000	0.571
Recall	0.875	1.000	1.000	0.800

4. Application on Feature Films

To demonstrate the generality of the proposed framework, we have also tested our system on several films.

4.1. Feature Selection and Likelihood Computation

The movies are composed according to the *film grammar*, i.e., the rules about how the movies are produced. We often observe different patterns in different types of movie scenes. For example, in action scenes, the shots are generally short in length, and the visual content of the shots changes rapidly. On the other hand, in drama scenes, the shots are longer, and the visual content is relatively consistent. We used two features computed from movies: shot



(a) Representative frames of example scenes in the movie

(b) Plot of the shot length in the movie.

(c) Plot of the visual content in the movie.

(d) PDFs of the 2D normal distributions of the first four scenes.

Figure 6. (a). Representative frames of some example scenes in movie *Gone In 60 Seconds*; (b). Plot of the shot length variable; (c). Plot of the visual content; (d). PDF plots on the 2D normals of the first four scenes in the movie.

length and visual content. Let l_s denote the length of shot s, and v_s be the visual content in that shot. The visual content is defined as, $v_s = \frac{1}{N_s}\sum_{i=1}^{N_s}(1 - HistInter(f_i, f_{i+1}))$, where $HistInter(f_i, f_{i+1})$ is the color histogram intersection between the i-th and $(i+1)$-th frames, and N_s is the number of frames in shot s. The plots of shot length and the visual content are shown in Figure 6. These two features are used in the construction of the likelihood.

In the film production, the patterns for different features are related to each other. For instance, in action scenes, the short shots are accompanied with high degree of visual content. Therefore, the features l_s and v_s should not be considered independently to each other. We use a two-dimensional normal distribution to model the features in a scene S_m, $N(g_s, m) = \frac{1}{\sqrt{2\pi|G|}} exp\Big(-\frac{(g_s-\hat{g_m})^T G^{-1}(g_s-\hat{g_m})}{2}\Big)$, where g_s is the feature vector $[l_s\ v_s]^T$. The vector $\hat{g_m}$ is computed as the maximum likelihood estimator for scene S_m, and G is the covariance matrix with determinant $|G|$. Again, by considering the shots to be recorded independently, the likelihood for scene S_m is, $\mathbb{L}(y_m|f_m) = \Big(\prod_{s=1}^{n_m} N(g_s, m)\Big)^{\frac{1}{n_m}}$.

4.2. System Performance

We have experimented our approach on three feature-length films: *Gone in 60 Seconds*, *Dr. No - 007* and *Mummy Returns*. The matching follows similar procedure as used in Section 3.3. In movies, usually there is not a concrete boundary between two adjacent scenes due to the editing effects. Movie chapters sometime are changed with smooth transitions. Therefore, matching based on the boundaries is not meaningful. Instead, we used a "recovery" method. Suppose there are reference scenes $\{T_1, T_2, ..., T_n\}$ and detected scenes $\{S_1, S_2, ..., S_k\}$. A reference scene T_m is said

Table 2. **Accuracy measures for three movies: *Movie1: Gone in 60 Seconds*, *Movie2: Dr. No - 007*, and *Movie3: Mummy Returns***

Measures	*Movie1*	*Movie2*	*Movie3*
Length	01:46:09	01:30:55	01:45:33
Num. of Shot	2237	677	1600
Num. of Scenes	29	17	19
Detected Scenes	25	20	19
Match	24	14	15
Precision	0.960	0.700	0.790
Recall	0.828	0.824	0.790

to be "recovered", if its major part ($> 50\%$) overlaps with one of the detected scenes. The "recovery" is a one-to-one correspondence, i.e., one reference scene can only be matched with at most one detected scene, and vice versa. We use the precision and recall measures defined in Section 3.3. Detailed results for movies are shown in Table 2.

5. Conclusions

In this paper, we have presented a general framework for the temporal scene segmentation of various types of videos. We segmented the video sequence by iteratively determining the boundary locations through a series of updates in the Markov chain. The posterior probability is computed based on the priors and the data likelihood. The method was applied to several home videos and three feature films, and high accuracy measures have been obtained (Tables 1 & 2).

Acknowledgement

This material is based upon the work funded in part by the U.S. Government. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the U.S. Government.

References

[1] L. Chaisorn, T-S. Chua and C-H. Lee, "The Segmentation of News Video Into Story Units", *ICME*, 2002.

[2] P. Green, "Reversible Jump Markov Chain Monte Carlo Computation and Bayesian Model Determination", *Biometrika*, 82, 711-732, 1995.

[3] F. Han, Z.W. Tu and S.C. Zhu, "Range Image Segmentation by an Effective Jump-Diffusion Method", *PAMI*, 2004.

[4] W. Hsu and S.F. Chang, "Generative, Discriminative, and Ensemble Learning on Multi-Model Perceptual Fusion Toward News Video Story Segmentation", *ICME*, 2004.

[5] Z. Rasheed, M. Shah, "Scene Detection In Hollywood Movies and TV Shows", *CVPR*, 2003.

[6] H. Sundaram and S.F. Chang, "Video Scene Segmentation Using Video and Audio Features", *ICME*, 2000.

[7] M. Yeung, B. Yeo, and B. Liu, "Segmentation of Videos by Clustering and Graph Analysis", *CVIU*, 1998.