# Automatic Segmentation of Home Videos

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

Temporal video segmentation is one of the fundamental and essential tasks in video processing, understanding and management. In this paper, we present an automatic method for segmenting the home videos into temporal logical units. We have developed a statistical framework using Markov chain Monte Carlo (MCMC) technique. The temporal scene boundaries are detected by maximizing the posterior probability of the model parameters. The model parameters contain the number of the scenes and the boundary locations of the scenes. The proposed method has been demonstrated on several home videos, and high accuracy has been obtained.

## 1. Introduction

Home video is a broad term that refers to the videos with a "free-style", e.g., family videos and videos taken by amateur videographers. They are acquired from a variety of sources that are recording some environments or activities. They come in different forms. Some are with high resolutions, while some others have low quality. Some have full field of view. Some may be recorded by cameras hidden in the bags, like spy cameras, so part of their field of view is blocked by the carrier. Some example key-frames of the home videos are shown in Figure 1. The videos are generally composed of the shots, that are created by camera operations, e.g., on/off or switching between cameras. Temporal scene segmentation is a process of clustering adjacent shots into groups, such that shots in each group correspond to a particular physical location or an on-going action.

Several temporal segmentation methods have been developed for different types of videos. Hanjalic *et al.* [2] proposed a method for detecting the boundaries of the story units in movies. In this work, inter-shot similarity is computed based on the block matching of the key-frames. Similar shots are linked, and the segmentation process is performed by connecting the overlapping links. Rasheed *et al.* [4] proposed a two-pass segmentation algorithm for feature films and TV shows. First, the potential video scene boundaries are detected based on the color-similarity feature among the shots. Over-segmented scenes from the first pass are then merged in the second pass, based on the analysis of the motion content in the scenes. These methods are based on the "filem/TV grammars", which is a set of production rules of how the movies/TV shows are generated. For instance, in action scenes, the shots are generally short, and their motion content is high. On the other hand, the shots are long and the visual appearance is smooth in drama scenes. However, this heuristic is not applicable in home videos, since they are usually recorded in a relatively "free" style without obvious format or patterns. Furthermore, some of the above mentioned methods are based on the pre-defined thresholds [4] [2]. It may not be suitable in home videos, since the feature plots sometimes are not distinctive across scenes. Therefore, these techniques create either over-segmentation or under-segmentation due to the difficulty of selecting the thresholds. There are also methods on the story segmentation of the news broadcast videos. Hsu *et al.* [3] proposed an approach based on discriminative models. The authors have developed the *BoostME*, which uses the Maximum Entropy classifiers and the associated confidence scores in each boosting iteration. These methods were developed based on the unique characteristics of news videos and require the special care on the anchor person shots, which are not relevant in the home videos.

In this paper, we propose a framework for the temporal segmentation on home videos using Markov chain Monte Carlo (MCMC) technique. The proposed method finds scene boundaries by maximizing the posterior probability of the model parameters using Markov chain. The Markov chain contains three types of updates with corresponding transition probabilities. Visual features are used for the likelihood computation. The final output of boundary locations is collected from multiple Markov chains. The rest of the paper is organized as follows: Section 2 describes the proposed framework in details; Section 3 presents the experimental results; finally, Section 4 concludes our work.

## 2. Proposed Framework

By the problem definition, given the video shots, scene segmentation of the videos is a process to group the shots into temporal clusters. In each scene, the shots are related by the consistent color patterns due to the similar environmental settings. We formulate scene segmentation as a change-point problem. In a change-point problem, the random pro-

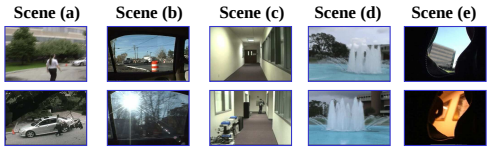


Figure 1: Five example home videos with the key-frames of some shots in each scene. (a,d) are outdoor scenes taken by hand-held cameras; (b) Outdoor scene taken by a camera in a car; (c) Indoor scene taken by a hand-held camera; (e) Outdoor scene taken by a spy camera.

cess has different controlling parameters over times. In our application, the scene boundaries are the change-points of the environment settings. MCMC has been demonstrated as an efficient method solving the change-point problems. We use a hierarchical Bayesian model in our MCMC process. Let $k$ be the number of video scenes, $y$ be the video data and $\theta_k$ be the vector containing the $k-1$ scene boundary locations. By Bayes rule, the posterior probability of the parameters $k$ and $\theta_k$ is,

$$p(k, \theta_k|y) = p(y|k, \theta_k)p(\theta_k|k)p(k), \quad (1)$$

where $p(k)$ is the model prior on $k$, $p(\theta_k|k)$ is the conditional prior on $\theta_k$ and $p(y|k, \theta_k)$ is the data likelihood. Since vector $\theta_k$ implicitly determines $k$, the likelihood can be written as $\mathbb{L}(y|\theta_k) = p(y|k, \theta_k)$. Furthermore, for simplicity purpose, we use $\pi(x)$ to denote $p(k, \theta_k|y)$, where $x$ represents the parameters $k$ and $\theta_k$.

The general MCMC algorithm [1] is well suited for our application, where the structure of the parameter vector may change during the process. The algorithm is described as follows:

- Initialize the model parameters $x_0$.
- At each iteration $i$, perform following actions:
  1. Create a new parameter $x'_{i-1} = x_{i-1} + \Delta x$, where $\Delta x$ is randomly drawn from some trial distribution $T(x_{i-1})$, based on $x_{i-1}$.
  2. Calculate the ratio $\alpha(x_{i-1}, x'_{i-1})$ as, $\alpha(x_{i-1}, x'_{i-1}) = min\Big\{1, \frac{\pi(x'_{i-1})q(x'_{i-1}, x_{i-1})}{\pi(x_{i-1})q(x_{i-1}, x'_{i-1})}\Big\}$.
  3. Update $x_i = x'_{i-1}$, if $\alpha > AcceptRatio$. Otherwise, set $x_i = x_{i-1}$.

In this algorithm, $q(x, x')$ is the transition probability from $x$ to $x'$, and its computation depends on the type of the updates. It should satisfy the reversibility, i.e., if $q(x, x')$ exists, then $q(x', x)$ should also exist. Therefore, the proposed updates should also be reversible to ensure this property. Also, it should be noted that the (+) operation in step (1) does not necessarily refer to the arithmetic addition. The arithmetic operation is only meaningful when the structure of the parameter vector does not change during the updates. In the situation where the vector structure has been modified, the (+) operator represents the process of obtaining $x'$ with a change $\Delta x$ that is based on $x$.

Since the Poisson distribution models the distribution on numbers of incidents in a time interval, we model the number of scenes, $k$, by a Poisson with mean $\lambda$, and the model prior is computed as $p(k) = e^{-\lambda}\frac{\lambda^k}{k!}$. If there are $k$ scenes in the video, then there are $k-1$ scene boundaries. Given the video with total $T$ shots, the conditional prior $p(\theta_k|k)$ is the probability of selecting a subset with size $k-1$ from $T-1$ shots. Thus, it can be defined in terms of combinations, $p(\theta_k|k) = \frac{1}{C_{k-1}^{T-1}} = \frac{(k-1)!(T-k)!}{(T-1)!}$.

Based on our formulation, the scenes are recorded independently from each other. Therefore, given the total $L$ scenes, the overall likelihood $\mathbb{L}(y|\theta_k)$ can be computed from the individual likelihood of each scene,

$$\mathbb{L}(y|\theta_k) = \Big(\prod_{m=1}^{L} \mathbb{L}(y_m|f_m)\Big)^{\frac{1}{L}}. \quad (2)$$

The geometric mean is taken here for the normalization purpose, so that the ratio test in the algorithm is meaningful. We discuss the computation of the individual likelihood in the following sections.

## 2.1. Feature Selection

Many features have been exploited in the field of video scene segmentation, including color, motion, shot length, etc. Since home videos are taken in a "free style", the motion content and the shot length are not distinctive across the scenes. Therefore, we have focused our effort on the analysis of the color statistics of the video. We use 3D color histograms in RGB space to represent the color information of the video frames, 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,

$$HistInter(f_i, f_j) = \sum_{b \in Allbins} min(h_i^b,\ h_j^b), \quad (3)$$

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 shots. However, for the shots with long durations and with high activity content, multiple key-frames form better representation. Suppose for shot $s$, there are total $n$ frames, the procedure for selecting the key-frames is described as follows [4],

- Include the middle frame into the key-frame set $\mathbb{K}_s$ as the first key-frame $\kappa_s^1$;
- For $i = 1 : n$, do
  If $max(HistInter(f_i, \kappa_s^j)) < Th, \forall \kappa_s^j \in \mathbb{K}_s$
  Include $f_i$ into $\mathbb{K}_s$ as a new key-frame.

Here, we use the color histograms for the key-frames.

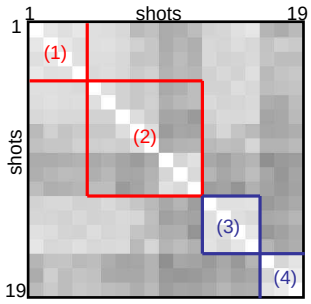


Figure 2: Visual similarity map of a testing video. The brighter cell represents higher similarity. The shots in the same scene possess higher similarity comparing across scenes. The bright blocks on the diagonal give the temporal scenes. The figure shows the intermediate results for one iteration, where the red scenes (1,2) are not matched with correct boundaries, and the blue scenes (3,4) are the correct detections.

## 2.2. Likelihood Computation

For the home videos, usually the shots in one temporal scene are coherent to the same environment. There are visual similarities exist among these shots. On the other hand, the shots from different scenes should be 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:

$$Sim(s_i, s_j) = max(\mathbb{C} - d_B(\kappa_{s_i}^m, \kappa_{s_j}^n)), \qquad (4)$$

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. After computing the visual similarity between all pairs of shots in the video, a similarity map is generated. One such map is shown in Figure 2. 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, $\mathbb{L}(y_m|f_m) = avg\big(\mathbb{M}(a:b)\big)$, i.e., the average similarity value of the diagonal sub-block in the similarity map $\mathbb{M}$. It is intuitive that the correct segmentation of the video gives the diagonal blocks to reach the maximum likelihood. To compute the overall likelihood, substitute $\mathbb{L}(y_m|f_m)$ into Eq.2. Up to this point, the overall likelihood $\mathbb{L}(y|\theta_k)$, the conditional prior $p(\theta_k|k)$ and the model prior $p(k)$ are determined. We describe the transition probabilities for each type of updates in next section.

## 2.3. Proposal Updates

The updating process contains two parts: diffuse and jumps. Diffuse is defined as the update without changing the structure of the parameter vector $x$, while the jumps do change the structure. In our case, the diffuse is the shifting of the boundaries between adjacent scenes. There are two types of jumps: merging of two adjacent scenes and splitting of an existing scene. The updates are randomly selected to be performed. Let $S_m$ be the $m$-th scene with shots $\{s_m^1, s_m^2, ..., s_m^{n_m}\}$, where $n_m$ is the number of shots in scene $S_m$.

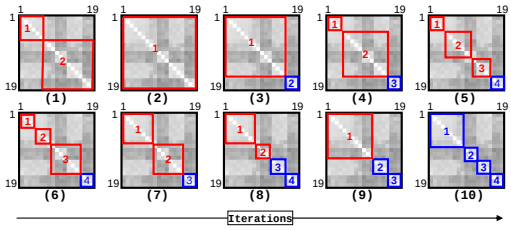


Figure 3: Ten updates from one single run. The red boxes are the detected scene that do not match with true boundaries, while the blue ones are the correctly detected scenes.

- *SHIFT*: Scene $S_m$ is randomly selected from $[S_1, S_{L-1}]$, such that the boundary between $S_m$ and $S_{m+1}$ is updated. The new boundary $s^t$ is randomly drawn from a normal distribution centering at the original boundary $s_{m+1}^1$ in the range of $[s_m^1,\ s_{m+1}^{n_{m+1}}]$. The updated scene $S'_m$ contains shots of $\{s_m^1, ..., s^{t-1}\}$, and the updated scene $S'_{m+1}$ contains $\{s^t, ..., s_{m+1}^{n_{m+1}}\}$.
- *MERGE*: A number $m$ is randomly drawn from the uniform distribution $[1, L-1]$, such that $S_{m+1}$ is merged with $S_m$. The new scene $S'_m$ now contains shots $\{s_m^1, ..., s_m^{n_m}, s_{m+1}^1, ..., s_{m+1}^{n_{m+1}}\}$.
- *SPLIT*: One target scene $S_m$ is randomly selected from $\{S_1, S_L\}$. The potential split is chosen uniformly from range $[1, n_m]$, let $i$ denote this split. The shots of scene $S_m$ are $\{s_m^1, ..., s_m^{i-1}\}$, and a new scene $S'$ is created with shots $\{s_m^i, ..., s_m^{n_m}\}$. This new scene is inserted in the scene list right after $S_m$.

A simple sequence of the updates is shown in Figure 3. In the ratio test, the individual $q(x, x')$ and $q(x', x)$ are not necessarily computed. Only the ratio $q(x', x)/q(x, x')$ is needed. For the shift update, the probability of selecting $S_m$ is $1/(L-1)$, and the probability of selecting the new boundary $s^t$ from the original boundary $s^{\hat{t}}$ is $p(\Delta t) = N(\Delta t; \sigma)$, $\Delta t = t - \hat{t}$. Due to the symmetry property of the normal distribution, the probability of shifting back to $s^{\hat{t}}$ from $s^t$ is same, since $p(-\Delta t) = p(\Delta t)$. Therefore, for the shift updates, the ratio $q(x', x)/q(x, x') = 1$.

The transition during a merge is related with the transition for a split, since merge and split are a pair of reversible updates. For the merge action, the transition $q(x, x')$ is straightforward, $q(x, x') = 1/(L-1)$. Suppose the number of scenes is reduced from $L$ to $L-1$ by merging $S_m$ and $S_{m+1}$ into a new scene $S'_m$, the transition probability is $1/(L-1)$ for randomly selecting $S_m$ from $\{S_1, ..., S_{L-1}\}$. To recover $L$ scenes back from $L-1$ scenes, a split has to be performed on the merged scene. First, the merged

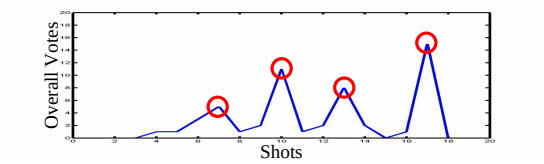


Figure 4: The overall votes of the shots to be declared as the scene boundaries after all runs. The red circles represent final output of the scene boundaries, which corresponds to the local maxima.

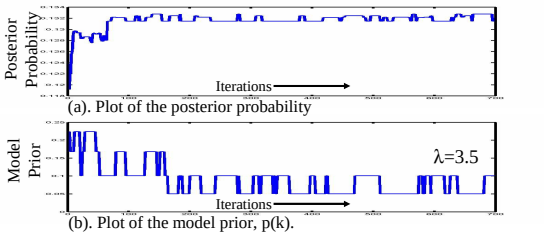


Figure 5: (a) Plot of the posterior during a single run. As demonstrated in the figure, after certain iterations, the posterior reached a "confidence" level and stays there with only minor fluctuations. (b) Plot of the model prior, $p(k)$.

scene $S'_m$ is selected with probability $1/(L-1)$. Then, the split boundary is selected with probability $1/(n'_m)$, where $n'_m$=$n_m$ + $n_{m+1}$. Therefore, $q(x,x')$ and $q(x',x)$ for the merge operation are, $q(x,x') = \frac{1}{L-1}$ and $q(x',x) = \frac{1}{(L-1)\times(n_m+n_{m+1})}$. For the splits, the transition probability, $q(x,x')$, and the probability $q(x',x)$ of its reverse operation, a merge of the split scenes, can be derived in a similar fashion, $q(x,x') = \frac{1}{L\times n_m}$ and $q(x',x) = \frac{1}{L}$. Replace $q(x,x')$ and $q(x',x)$ in the ratio test, in addition with the computed posteriors, the acceptance can be determined.

# 3. Experimental Results

The proposed method has been tested on four home videos. The scenes were recorded with various environment settings (Figure 1). Since the iteration in each run is carried on the fixed numbers, it is possible that the last update may not result in the accurate scene boundaries. Sometimes, it may result in the neighborhood of the true boundary. To overcome this problem, we have multiple Markov chains (runs) executed independently. The result from each individual chain provides the votes to the corresponding shots. After certain runs, the shots that have the locally highest votes represent the final scene boundary output. Figure 5 shows the overall votes of the scene shots being declared as scene boundaries. Even though one single chain may not hit the correct result, there is an issue of the posterior probability reaching a "confidence" level. As shown in Figure 5, after certain iterations, the posterior probability reaches a level and stays there with only minor fluctuations. It should be noted that the time to reach the confidence level is rapid, if the data size is small.

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 |

The matching between the ground truth data and the segmented scenes are based on the matching of the 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 $\{s_1, s_2, ..., s_L\}$ denote the starting shots of the detected scenes. $t_i$ is declared as matched if one or more of the detected boundaries $s_j$ falls in its evaluation interval. In our experiment, we allow a window of 1 shot on each side of the reference boundary. Two accuracy measures are used as the system performance: precision and recall, $Precision = X/A$ and $Recall = X/B$, where $X$ is the number of correct matches; $A$ is the total number of system detections; $B$ is the total number of ground truth boundaries. The detailed measures are shown in Table 1.

# 4. Conclusions

In this paper, we have presented a statistical framework for the temporal scene segmentation on the home videos. We segmented the video sequences by achieving the maximum posterior probability applying the Markov chain Monte Carlo (MCMC) technique. The model parameters consist of the number of the scenes and their corresponding boundaries. The posterior probability is computed based on the model priors and the data likelihood. The method has been applied to several home videos, and high accuracy measure have been obtained.

# References

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

[2] A. Hanjalic, R.L. Lagendijk, and J. Biemond, "Automated High-Level Movie Segmentation for Advanced Video-Retrieval Systems", *CSVT*, Vol.9, Issue.4, 1999.

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

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

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