COMPUTER VISION AND IMAGE UNDERSTANDING
Vol. 71, No. 1, July, pp. 94–109, 1998
ARTICLE NO. IV970628

Segmentation of Video by Clustering and Graph Analysis*

Minerva Yeung† and Boon-Lock Yeo‡

Intel Research Labs, Santa Clara, California 95052

and

Bede Liu§

Department of Electrical Engineering, Princeton University, Princeton, New Jersey 08544

Received March 20, 1996; accepted March 19, 1997

Many video programs have story structures that can be recognized through the clustering of video contents based on low-level visual primitives and the analysis of high-level structures imposed by temporal arrangement of composing elements. In this paper we propose techniques and formulations to match and cluster video shots of similar visual contents, taking into account the visual characteristics and temporal dynamics of video. In addition, we extend the *Scene Transition Graph* representation for the analysis of temporal structures extracted from video. The analyses lead to automatic segmentation of scenes and story units that cannot be achieved with existing shot boundary detection schemes and the building of a compact representation of video contents. Furthermore, the segmentation can be performed on a much reduced data set extracted from compressed video and works well on a wide variety of video programming types. Hence, we are able to decompose video into meaningful hierarchies and compact representations that reflect the flow of the story. This offers a mean for the efficient browsing and organization of video.

Key Words: **hierarchical decomposition of video; nonlinear access; scene identification; scene transition graph; matching and clustering of video shots; key frame selection; video browsing; video representation; video analysis.**

> Cause and effect are mutually dependent, forwards and retrospectively. One begets the other by an inexorably ordained necessity, which would be fatal for us if we were able to discover all of the connections at once. The link of cause and effect, in other words the transition from one state to another, is also the form in which time exists, the means whereby it is materialized, in day to day practice... The point is to pick out and join together the bits of sequential fact, knowing, seeing and hearing precisely what lies between them and what kind of chain holds them together. That is cinema.
>
> —Andrey Tarkovsky, Sculpting in Time—Reflections on the Cinema, translated by K. Hunter-Blair.

1. INTRODUCTION

Methods and techniques to automatically analyze video documents based on contents and build structures that facilitate both hierarchical organization and semantical understanding of video are important for applications such as browsing, navigation, and search in video databases. Often the goal of such analyses is to provide fast and meaningful nonlinear access to relevant materials in video.

Existing content-based analyses of video focus on detecting shot boundaries (commonly known as *scene change*[1] detection; see, for example, [3–5]) based on objective visual primitives such as color, image correlations, and sometimes motion parameters. By detecting the shot boundaries, individual shots are rediscovered and they represent the fundamental units of video.

This is an important step to condense the visual information presented. An image, usually the first frame, is chosen to represent the shot, and the collection of such images gives a summary of the video sequence on which the users can browse. This presentation format relieves the user from the need to watch the entire video during browsing. However, in many video programs, there are hundreds of shots (the number can be thousands in modern action movies).[2] In addition, a frame taken from a shot often fails to represent the dynamic and time varying contents within the shot. In such cases, presenting shots in a one dimensional image array does not offer the users an efficient way to

* Part of this paper was presented at International Conference on Multimedia Computing and Systems 1996 [1] and International Conference on Pattern Recognition 1996 [2]. This work was conducted when the first two authors were at Princeton University.

† E-mail: Minerva-Yeung@ccm.sc.intel.com.
‡ E-mail: Boon-Lock-Yeo@ccm.sc.intel.com.
§ E-mail: liu@ee.princeton.edu.

[1] This is a misnomer. These techniques detect the boundary when a camera shot transits to another. In film production, a scene is composed of many different shots unified by a common locale or event. Throughout the paper, we call such operations "shot boundary detection" to distinguish them from the actual segmentation of scenes. The word *scene* will be used as in film production.

[2] For example, 500 is not uncommon in typical films [6].

1077-3142/98 $25.00

browse, navigate, and search for any particular video segments. This presents a greater challenge for the user if the user has never watched the video and has no idea where to initiate the search in the image array.

Another form of content analysis of video is to differentiate the contents presented in the individual frames of video documents with predefined context and extract the more "important" frames for presentation and browsing purposes. This can be achieved by extracting highlights [7] of sports events, detecting captions [8, 9], and key-words spotting [9]. Detection of these highlights allows skimming of long sequences of video and cut significantly the time of browsing the entire sequence.

Model-based parsing of video sequences has been proposed for news broadcasts [10, 11] to extract specific semantic elements of the sequences based on specific models of news broadcasts. Generic models, on the other hand, may be difficult to build for different sequences of various programming formats such as sitcoms, dramas, and movies. This may limit the applications to certain types of video programs.

In reality, video is a medium of presentation, a means to convey messages and in fact, a form of document. For many video documents, there are underlying story lines which are reflected by the visual contents and the temporal organization of the material presented. For example, repetition of shots are common in motion picture production. The recurring shots very often can be recognized from visual similarities. In addition, the temporal arrangements and dynamics of compositing shots in motion picture presentation reflect the structures and the flow of the story. In view of such underlying structures, it is desirable to automatically identify both visual and temporal relationships in video and extract a compact representation of the story. This can be useful in applications such as video browsing and navigation: a user can first identify the scenes taking place at a particular location using the visual information, select the scene desired using temporal information, and similarly navigate through various shots in the scene to locate the desired point in video by both visual and temporal information. In addition, the compact representation serves as a summary of the content and structure in video such that a user, by recognizing the underlying story structure, is able to get a quick overview of the story without viewing the entire video sequence.

In this paper we propose a general framework of clustering of video shots based on both visual similarities [12] and temporal localities of the shots which we call *time-constrained clustering*. We use the *Scene Transition Graph* (STG), a directed graph representation proposed by Yeung *et al*. [13], and extend the formulations to accommodate our analysis of story components extracted by the graph. We will show that time-constrained clustering and scene transition graph together provide the basis on which analysis can be performed to extract story structures that reflect the flow of underlying story and to meaningfully segment individual units from the story. Each such unit will closely approximate a scene, and such segmentation cannot be achieved by any of the existing shot boundary detection schemes.

The measurement of visual similarity takes into account the temporal variations within individual video shots. In addition, reduced data sets are extensively used to achieve fast analysis and processing of video. The reduction in data comes both from the spatially reduced image sequence extracted directly from compressed video and the temporally subsampled collection of representative frames through a key frame selection scheme. Furthermore, story structure and programming type of the video are not required—the structure is discovered in the process of the analysis. This allows long video sequences of many frames to be telescoped into a compact representation featuring the visual contents and temporal structures and effectively provides a *visual summary* of the underlying story. Consequently, video sequences can be automatically parsed to achieve a hierarchical decomposition based on contents, which in turn offers better organization of video and provides some high level structures to assist semantic understanding for browsing and navigation applications in digital video libraries.

The techniques proposed in this paper to break a video into large meaningful units based on the interactions between shots can be seen to be analogous to the classical approaches to segmentation of still images. In the segmentation of an image, one seeks to identify regions in the image that are uniform and homogeneous with respect to some characteristics. Pixels are grouped together into clusters in the spatial domains according to the desired characteristics. In our approach to the segmentation of video into large meaningful units like the scenes, we want to partition the video into units, each of which represents a sequential collections of interrelated shots that are unified by a common locale or dramatic event. In other words, such sequences of shots compose a meaningful unit in the story. The common characteristics of shots we seek are the mutual interactions in terms of visual similarities and temporal locality. We then achieve the segmentation through the use of graph property in the scene transition graph constructed.

The organization of the paper is as follows: In Section 2, we examine the role of "time" in the composition of video programs and the characteristics of motion picture presentation; we then provide a general framework of matching and clustering of video shots based on visual similarities across image frames represented in the shots and temporal variations within individual shots in Section 3. We extend this framework to take into account the temporal locality of shots by incorporating time constraints in the clustering process in Section 4. In Section 5, we review the *scene transition graph*, propose new formulations, define the *story unit*, and explain the segmentation of video into story units by the analysis of STG coupled with time-constrained clustering. The block diagram for the segmentation of story units and the building of the STG, and the details of implementation in various steps of the analysis, are presented in Section 6. Section 7 shows how the story units segmented can be further analyzed to approximate the actual scene boundaries. The experimental results are discussed in Section 8.

2. DISSECTING VIDEO

We refer to a complete moving image document as a video sequence in the paper. The fundamental unit of the production of video is the shot, which captures continuous action from a camera. A shot reflects a fragment of the story. A scene is "usually composed of a small number of interrelated shots that are unified by location or dramatic incident" [14]. At the higher level, several scenes can be composed into an act. The act-scene-shot decomposition forms an hierarchy; the story line then chains the shots into scenes, the scenes into acts, and together they are linked to tell the story. The story structure and organization are commonly found in motion pictures, TV movies, sitcoms, etc., and are not confined to such. In many structured programs like news, documentaries, TV magazines, and talk shows, each program is a composition of distinctive segments, each of which represents a distinct unit with underlying messages. The segments are then linked together by some establishing shots like the shot of the anchor room or the host.

It is advantageous to be able to segment meaningful units from a video sequence without specific knowledge of the particular program nature of the sequence. Instead, the segmentation will be based on the presentation of the featured video sequence along the time line and the visual contents. Consequently, long sequences of video can be effectively broken down into meaningful segments, and the story can be further condensed and presented in a hierarchical fashion for efficient browsing applications. To do this, we first must examine two characteristics of video presentation that are common in a variety of programs.

The most essential characteristic of motion picture presentation is the sequential montage. Montage "refers to the editing of the film, the cutting and piecing together of exposed film in a manner that best conveys the intent of the work" [15]. Montage was studied by Eisenstein [16] and other great Soviet directors. As described in [6], "One of the binding and immutable conditions of cinema is that actions on the screen have to develop sequentially, regardless of the fact of being conceived as simultaneous or retrospective... In order to present two or more processes as simultaneous or parallel you have necessarily to show them one after the other, they have to be in sequential montage." For example, in Hollywood the use of alternating close-up shots of two characters is common to convey conversations, innuendos, and reactions [15]. Repeating shots of the same person or same settings, alternating or interleaving with other shots, are often seen in TV programs, news broadcasts, interviews, talk shows, etc. to convey parallel events in a scene.

In addition, contents in a motion picture and many video materials are localized in time. For example, given two shots of the same person, if the two shots are juxtaposed together, they are more likely to be of the same scene than if they are placed far apart in time from each other. Often, a scene is made up of temporally adjacent shots indicating their interrelationships. On the other hand, two visually similar shots, if occurring far apart in the time line, may potentially represent different contents or locale in different scenes.

The sequential montage has produced a distinctive consequence at the shot level: repetitive shots of similar contents interleaved by other shots with different contents. These shots may be the shots of the same person, taken from the same camera, at the same location, or about the same event. Most often, the similarities of contents are shown through similar visual characteristics of the compositing frames in the shots. In addition, the similarity of individual video shots must reflect the time-varying nature of the contents. This form of similarity measurement is independent of the featured video program and does not require any specific knowledge of the underlying story. Hence, we are able to classify the shots by grouping shots of similar visual contents into special classes of distinct labels to further condense the information presented.

The clustering of video shots according to visual similarities offers another level of organization beyond the detection of individual shots. The resulting clusters of video shots, however, may suffer from the lack of "context" because the temporal information is lost. Visual similarity of shots alone may not be sufficient to differentiate the context and contents of individual shots as each shot is itself a distinct unit of *time* in the featured video presentation. For long programs, there is little justification for grouping shots into the same class based only visual contents without regard to the timing of the context. On the contrary, such grouping may lead to shots from different scenes clustered together, which may not only complicate the understanding, but also make the segmentation of distinct scenes very difficult, if not impossible.

In the following sections, we shall discuss how video shots can be clustered together based on visual similarities of image contents and the temporal dynamics shown in the visual contents within individual shots. We shall show that using a selected set of representative frames for each shot, the visual content variations are captured and the similarities of video shots are better reflected. We will then show how to incorporate time constraints that reflect temporal locality of contents in the clustering process. We call this time-constrained clustering of video shots. This clustering framework, when incorporated into the scene transition graph, is capable of capturing the flow of the story and allows the segmentation of video into *story units*, each of which closely approximates a scene.

3. MATCHING AND CLUSTERING OF VIDEO SHOTS

Throughout this paper, we shall treat a *shot*, defined as a continuous sequence of actions, as a fundamental unit in a video sequence.[3]

To organize and retrieve video shots, we need to analyze, compare, and process the shots based on their contents. Because of the huge amount of data presented by the many frames of a

[3] Further subdivisions of shots into smaller units are possible and can be easily accommodated by the framework defined below.

video shot, processing is normally done on reduced data. One common practice reported in existing literature is to use one image (*a key frame*), typically the first frame of the shot, to represent the shot. Processing operations are performed on these key frames, e.g., in [17] the similarity of two shots are measured by the similarity of the two key frames.

Generally speaking, in a video shot where object and camera motions are prominent, a representative image is not sufficient for the faithful analysis of the image set it represents. Yet it is computationally burdensome to include every frame in the shot for processing purposes. We need a good representation of video shots for faithful analyses. The representation must be able to optimize the trade-off between preserving the visual contents and temporal dynamics and minimizing the size of data needed for efficient computing purposes.

In Sections 3.1 and 3.2, we shall look at the visual similarity measurements of video shots proposed in [12]. We also show how a reduced representation can be extracted which effectively captures the dynamics in the video shots on which the matching and clustering processes are performed. We proceed to formulate the problem of clustering of video shots in Section 3.3.

3.1. Similarity of Video Shots

Here we assume that the individual video shots have been segmented out from a video sequence by temporally detecting the shot boundaries. Since each video shot is represented by its collection of frames it is reasonable to assume that there are no abrupt changes or incorporation of unrelated images present.

Similarity between two video shots must give sufficient indication of the same events, persons, or locations. When the camera follows the flow of the event to different locations or to include new elements, the temporal variations are all captured within a video shot. To match the contents in a way that agrees with human perception of similar contents, similarity of video shots are more suitably measured in terms of the intersection of the flow of imagery paths or camera trajectories.

We define similarity of two video shots as follows.

DEFINITION 1. Given two video shots $S_i = \{f_k\}_{k=b_i}^{e_i}$ and $S_j = \{f_k\}_{k=b_j}^{e_j}$, S_i and S_j are similar if there exists a pair of similar frames f_m, f_n with $b_i \le m \le e_i; b_j \le n \le e_j$.

DEFINITION 2. The dissimilarity index of S_i and S_j is defined as

$$d(S_i, S_j) = \min_{b_i \le l \le e_i, b_j \le k \le e_j} D(f_l, f_k). \tag{1}$$

Here $D(.,.)$ measures the dissimilarity between two image frames. More general definitions can be set to require the number of similar images in the two sets to be greater than a threshold z; $d(S_i, S_j)$ will then take the value of the zth smallest value of dissimilarity indices measured from the two sets of frames.

Various criteria can be used to measure the dissimilarity $D(f_i, f_j)$ between two images f_i and f_k. We use two measures of dissimilarity indices of video shots based on color and luminance information as proposed in [12]. $D(\ )$ is normalized to take on values between 0 and 1. The readers are referred to [12] for the details of shot matching schemes and the effectiveness. It should be noted that other measures of image similarity such as the quadratic distance between color histogram [18] or distance based on low-order moments of histogram [19] can also be used.

3.2. Reduced Data Sets for Video Shots

The processing of video shots usually involves computation on large data sets. It is desirable to perform the processing on reduced data sets instead of the original full-frame video sequence. We reduce the size of original video by temporal and spatial subsampling.

The dissimilarity measure of two shots is defined in (1). The definition requires the computation of the dissimilarity indices for all image pairs in the two shots, and thus is computationally expensive. On the other hand, due to the inherent size of the shot, it is tempting to use one frame to represent each shot. As we have shown previously, however, such representative image often does not capture the dynamics of contents. In reality, we observe that for a given shot, consecutive frames often present related information which is redundant. To reduce the computation required in various processing steps and to represent the contents of a video shot concisely with minimal size, we propose that each shot be represented by its representative image set—a good but nevertheless greatly subsampled collection of $\{f_k\}$ which can effectively capture temporal variations due to camera operations or object movements. For a video sequence which has little or no variation, one such representative image (e.g., the first image) is sufficient. In a long video shot or a shot with many variations, we choose more images. The selection of the representative set is achieved by nonlinear temporal sampling, which measures the dissimilarity index between the last selected frame and the remaining frames in a given shot and selects frames with considerable variations to be the representative frames.

That is, given a video shot $S_A = \{f_b, f_{b+1}, \ldots\}$, let R_A denote the set of representative frames selected for S_A. Initially $R_A = \{f_b\}$, that is, the first frame is always selected. Let f^* be the last selected frame and m be its frame index, thus we initialize $f^* = f_b$ and $m = b$. We then proceed as follows:

1. Select smallest $l > m$ such $D(f^*, f_l) > \epsilon$.
   Set $f^* \leftarrow f_l$. Set $m \leftarrow l$.
2. Update R_A: $R_A \leftarrow R_A \cup f_l$.
   Repeat 1 and 2 until the end of the shot.

The threshold ϵ is a predefined error threshold taking values in [0, 1]. The frame selection process is equivalent to a nonuniform sampling in time. Compared with uniform subsampling, it has the advantage of taking in relatively few frames in a still shot even when the shot is long and captures the frames which exhibit mores variations.

By using only the representative set of frames $R_A \subset S_A$ for shot S_A, we are essentially approximating the value of (1)

$$d(S_i, S_j) \approx \min_{f_l \in R_i,\, f_m \in R_j} D(f_l,\, f_m). \tag{2}$$

The error threshold ϵ, together with the variations in visual contents within each shot, will determine the number of frames selected to represent the shot. Given the fact that no prior knowledge of the video contents is available in most test sequences, ϵ is the parameter in selecting the set of frames to represent a video shot and can be predetermined in practice. When $\epsilon = 0$, the representative set R consists of every frame in the shot. When $\epsilon = 1$, R consists of only the first frame.

We have found that the proposed representation leads to good matching results when compared to those using original sequences, but uses on the average only 2 to 5% of the number of frames. In our segmentation experiments, we use about 5% of the frames for shot matching for satisfactory performances. Using a higher percentage has little impact on the clustering results but increases the computational complexity. To select less than 5% of the frames, we use ϵ in the range [0.02, 0.04] as recommended in [12].

In addition to temporal subsampling of data via the selection of representative frames, spatial subsampling is used to achieve another level of reduction of data and associated computations. In our processing steps like matching and clustering of video shots, and the building of graph representation later, we use the DC or DC + 2AC images extracted from compressed MPEG or Motion-JPEG video proposed by Yeo [20]. The reduced images are extracted with significantly fewer computations and without full decompression of the compressed data. Although some details are lost, the global visual characteristics such as color and shape are well preserved. In addition, the reduced image sequence is also sufficient to capture temporal variations within a shot.

3.3. Clustering of Video Shots

We assume a dissimilarity measure $d(S_i, S_j)$ defined previously between two shots S_i and S_j. We want to group shots that are similar together into a cluster. In addition, different clusters should have sufficiently different characteristics. Denoting by C_i the ith cluster, we define the following criterion of clustering

[**Criterion I**] Define

$$\epsilon_i = \max_{y,z \in C_i} d(y, z). \tag{3}$$

Given $x \in C_i$, then

$$d(x, w) > \epsilon_i, \quad \text{for all } w \in C_j,\ j \neq i.$$

Criterion I implies a strong condition on the similarity of shots in a cluster. It requires each shot to be similar to every other shot in a cluster. In this case, with each cluster C_i is an associated maximum dissimilarity ϵ_i defined by (3). It measures the maximum dissimilarity between any two shots in the cluster. In addition, any other shot outside of the cluster must have a dissimilarity greater than ϵ_i relative to any shot in the cluster.

Hierarchical clustering methods based on *complete-link*, described in [21] and the references therein, generate clusters that satisfy the condition listed in Criterion I. Before describing the algorithms, we define the dissimilarity between two clusters C_i and C_j as

$$d_{\max}(C_i, C_j) = \max_{x \in C_i, y \in C_j} d(x, y). \tag{4}$$

The algorithm proceeds as follows:

[Algorithm Complete-link method]

1. Initially, there are N clusters $\{S_1\}, \{S_2\}, \ldots, \{S_N\}$. Each cluster contains a shot.
   Set *NumCluster* $\leftarrow N$.
2. Stop when
   $$d_{\max}(A, B) > \delta, \text{ for all clusters } A \neq B.$$
   or *NumCluster* $= 1$.
3. Find the least dissimilar pair of clusters, R and S, according to (4), i.e., find R and S such that
   $$d_{\max}(R, S) \leq d_{\max}(A, B), \text{ for all clusters } A, B.$$
4. Merge R and S into a new cluster.
   Set *NumCluster* $\leftarrow$ *NumCluster* $- 1$.
5. Go to step (2).

The parameter δ controls when the iteration stops. It is normalized to take on values from 0 and 1 and defines the minimum separation between any two resulting clusters. *NumCluster* in the algorithm maintains the number of disjoint clusters formed. At each step, the algorithm merges two most similar clusters into a new cluster based on cluster dissimilarity (4). It is easy to see that at each step of the process, the clustering criterion I is always satisfied.

In the implementation, the dissimilarities of the newly merged clusters to other clusters must be updated accordingly at each iteration. One algorithm that implements such a clustering method is based on the proximity matrix $\mathcal{D}$ described in [21] and the references therein. An $N \times N$ proximity matrix $\mathcal{D} = [d(S_i, S_j)]$ has as entries the dissimilarity between two shots. At each step, the matrix is updated by deleting rows and columns corresponding to cluster R and S and adding a new row and column corresponding to the newly formed cluster $R \cup S$.

Initial generation of the matrix $\mathcal{D}$ requires

$$(N-1) + (N-2) + \cdots + 1 = \frac{N(N-1)}{2}$$

computation of $d(.,.)$.

The clustering parameter δ reflects the maximal visual dissimilarities allowed across different shots within each cluster. It acts as a threshold to give a corresponding partition of video shot

clusters. In practice, δ can be predefined, or selected interactively by an user looking into the dendrogram built in the clustering process. It can also be automatically selected by detecting sudden big jumps in the proximity value when two clusters merge. We will include the results on segmentation of video using various δ's later in Section 8.2.

4. TIME-CONSTRAINED CLUSTERING OF VIDEO SHOTS

In order to segregate one scene from another, we take into account *temporal locality of contents* in video. It means that for any two shots that are far apart in time, even if they share similar visual contents, they potentially represent different contents or occur at different scenes. Time-constrained clustering further imposes a time-window parameter T that prevents two shots that are far apart in time but similar to be clustered together. In this section, we will show that the addition of temporal constraints incurs minimal changes to the clustering algorithms but significantly reduces the computational complexity.

To proceed, we define the temporal distance between S_i and S_j to be

$$d_t(S_i, S_j) = \begin{cases} \min(|b_j - e_i|, |b_i - e_j|), & i \neq j; \\ 0, & i = j. \end{cases} \quad (5)$$

It is the distance in number of frames from the end of the earlier shot to the beginning of the latter one. We propose to use a time-window parameter T (T is the elapsed number of frames) as a constraint on the clustering process: shots can be clustered together if they fall within the time window. Formally, we modify Criterion I as follows:

[**Criterion I′**] For all $x \in C_i$,

1.
$$\max_{y \in C_i} d_t(x, y) \leq T$$

2.
$$\text{Define } \epsilon_i = \max_{y,z \in C_i} d(y, z), \quad \text{then}$$
$$d(x, w) > \epsilon_i, \quad \text{or}$$
$$d_t(x, w) > T, \quad \text{for all } w \in C_j,\ j \neq i.$$

In the modified criterion, the temporal distance threshold T is used on top of the visual dissimilarity to separate shots in a cluster from the next. The additional criterion is that in each cluster, we require that no two shots can be separated by more than T frames apart. In this modified Criterion I′, we can use the same clustering algorithm previously described with only one slight modification. Instead of $d(.,.)$ (and hence $d_{\max}$), we define

$$\hat{d}(S_i, S_j) = \begin{cases} d(S_i, S_j), & \text{if } d_t(S_i, S_j) \leq T; \\ \infty, & \text{otherwise.} \end{cases} \quad (6)$$

and similarly

$$\hat{d}_{\max}(C_i, C_j) = \max_{x \in C_i, y \in C_j} \hat{d}(x, y). \quad (7)$$

It is straightforward to see that using $\hat{d}$ and $\hat{d}_{\max}$, the same clustering algorithm generates a hierarchy of clusters, and clusters at each stage satisfy Criterion I′.

In addition to the capability of segregating shots that are not in the same scene, time-constrained clustering has an important implication in complexity. Denote by L the maximum number of shots that any shot is temporally close to, i.e.,

$$L = \max_i |\{S_j : d_t(S_i, S_j) \leq T\}|,$$

then instead of $(N(N-1))/2$ shot comparisons, we need most $N(L-1)$ shot comparisons since the remaining ones violate the temporal constraints imposed. For L much smaller than N, this is a significant reduction in computational complexity. In the modified proximity matrix $\hat{\mathcal{D}}$, only the entries near the main diagonal can be finite.

5. SCENE TRANSITION GRAPH AND THE ANALYSIS

The formulation of scene transition graph was first proposed in [13] as compact representation of the underlying story for browsing purposes. The graph structure can be extracted automatically using visual contents and temporal information without specific knowledge of the video content and structure. The nodes are clusters of visually similar shots and the edges indicate the temporal flow of the story. We first review the definition of STG in Section 5.1 and discuss how it can be used for the analysis of video in the context of time-constrained clustering of video shots in Section 5.2.

5.1. Definition of Scene Transition Graph

The definitions of STG presented below are part of a more general model of Hierarchical Scene Transition Graph (HSTG) proposed in [13].

A scene transition graph is a directed graph with the property $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{F})$, where $\mathcal{V} = \{V_i\}$ is the node set and $\mathcal{E}$ is the edge set. $\mathcal{F}$ is a mapping that partitions the set of shots $\{S_i\}$ into $V_1, V_2, \ldots$, the members of $\mathcal{V}$. For given U, W in $\mathcal{V}$, (U, W) is a member of $\mathcal{E}$ if there exists some S_l in U and S_m in W such that $m = l + 1$. In the discussion of shots S_i, we assume the following ordering of shots: if $i < j$, then shot S_i occurs before shot S_j in time.

Thus, the collection of shots is partitioned to form nodes of $\mathcal{G}$; each node represents a cluster of shots. $\mathcal{F}$ partitions $\{S_i\}$ into $V_1, V_2, \ldots$ such that nodes in each V_i are *sufficiently similar*, according to some dissimilarity metrics. In the framework of time-constrained clustering, $d(.,.)$, parameters δ and T uniquely define the mapping $\mathcal{F}$. A directed edge is drawn from node U to

W if there is a shot represented by node U that immediately precedes any shots represented by node W. Relations between clusters are governed by temporal ordering of shots within the two clusters. A simple example would be a scene of conversation between two persons; the camera alternates between shots of each person. The graph $\mathcal{G}$ consists of two nodes V_1 and V_2; (V_1, V_2) and (V_2, V_1) are both members of $\mathcal{E}$.

STG is able to represent compactly the structures of shots and the temporal flow of the story for many video programs. Unfortunately, clustering of visually similar shots without the differentiation of context (as in [13]) can render the STG very complicated, and consequently adversely affect the analysis of the resulting graph and the segmentation of video into meaningful units beyond the shots. Figure 1 illustrates such an example.

In Fig. 1a, a sample video with nine shots is shown. It consists of three scenes: two taken place at a meeting and one at the office. The meeting scenes are interleaved by the office scene. The STG, shown in Fig. 1c, consists of 5 nodes, formed by condensing visually similar shots into distinct clusters. Note that while we have condensed the video into a more compact structure of STG, it does not explicitly depict the flow of story from meeting to office and back to meeting. This is due to the grouping of visually similar shots in scenes 1 and 3 without the realization of the different context of these shots.

5.2. New Attributes of the STG for Analysis Purposes

To enable the analysis of video through the analysis of STG, new attributes and formulations are defined. We first define the attributes of a STG $\mathcal{G}$

$$FS(\mathcal{G}) = \text{index of the first shot represented by } \mathcal{G}; \quad (8)$$

$$LS(\mathcal{G}) = \text{index of the last shot represented by } \mathcal{G}. \quad (9)$$

Thus, if a STG $\mathcal{G}$ is constructed from shots $S_l, S_{l+1}, \ldots, S_m$, then $FS(\mathcal{G}) = l$ and $LS(\mathcal{G}) = m$.

We also want to associate with each STG $\mathcal{G}$ an undirected graph $\hat{\mathcal{G}} = (\mathcal{V}, \hat{\mathcal{E}})$. $\mathcal{G}$ and $\hat{\mathcal{G}}$ have the same node set; in addition,

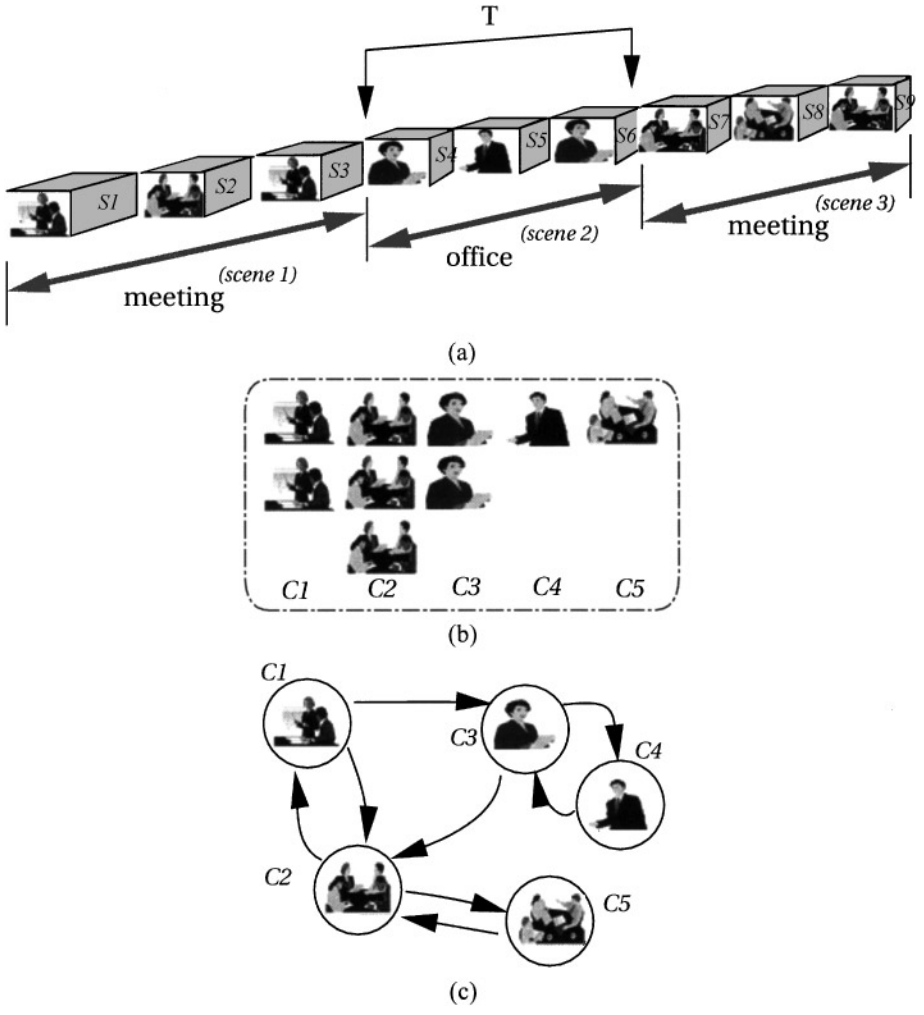

FIG. 1. Sample video sequence with repeating scenes: (a) 3 scenes of 9 shots, (b) sample clustering results, (c) the Scene Transition Graph.

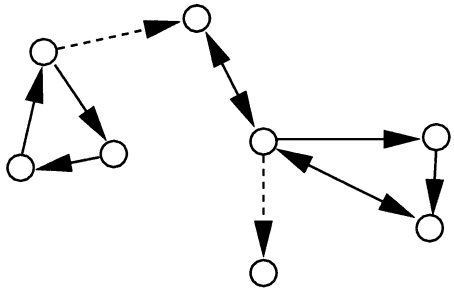

FIG. 2. A directed graph. Its cut edges are drawn in dashed line.

for each edge $\vec{e}$ of $\mathcal{G}$, there is a corresponding edge e in $\hat{\mathcal{G}}$ with the same ends. One important feature in the undirected graph $\hat{\mathcal{G}}$ is a *cut edge* defined as follows: an edge is a cut edge in an undirected graph, if when is removed, results in two *disconnected graphs* [22]. The set of cut edges in $\hat{\mathcal{G}}$ partitions $\hat{\mathcal{G}}$ into disjoint connected subgraphs $\hat{\mathcal{G}}_1, \hat{\mathcal{G}}_2, \ldots, \hat{\mathcal{G}}_n$, where each $\hat{\mathcal{G}}_n = (\mathcal{V}_n, \hat{\mathcal{E}}_n)$. Correspondingly, the cut edges induce the same partition on $\mathcal{G}$ such that there are now n disjoint STG's, $\mathcal{G}_1, \mathcal{G}_2, \ldots, \mathcal{G}_n$, where $\mathcal{G}_n = (\mathcal{V}_n, \mathcal{E}_n, \mathcal{F})$. For the mapping $\mathcal{F}$ we work with in this paper, $\mathcal{F}$ from $\mathcal{G}$ is preserved in each $\mathcal{G}_i$ after the partitioning of the graph $\mathcal{G}$.

Without loss of ambiguity, we shall refer to the directed edge $\vec{e}$ as a cut edge in $\mathcal{G}$ when the corresponding edge e in $\hat{\mathcal{G}}$ is a cut edge of $\hat{\mathcal{G}}$. Figure 2 shows an example directed graph; the cut edges are drawn using dashed line.

Each partitioned STG $\mathcal{G}_i$ represents the interactions of shots in a story unit. Formally, a *story unit* represented by $\mathcal{G}_i$ is the sequence of shots $S_{l_i}, S_{l_i+1}, \ldots, S_{m_i}$, where $l_i = FS(\mathcal{G}_i)$ and $m_i = LS(\mathcal{G}_i)$. Thus, it represents a contiguous segment of video frames; the shots from this segment interact in an intense and meaningful fashion. The collection of all cut edges of the STG $\mathcal{G}$ then represents all the transitions from one story unit to the next. Such transitions reflect the natural flow of story in the video. As we will see in later sections, a *story unit* closely approximates a scene.

5.3. STG Analysis and Time-Constrained Clustering

The *montage* presentation in video gives rise to a distinct form of *temporal interactions* in a given scene. A scene is composed of related shots in a particular setting. In a particular setting, there is often coexistence of multiple elements (e.g., multiple casts). Thus, in a scene we see the shots of these elements juxtaposed and linked together. We say two shots *interact* if they are juxtaposed side by side, that is, temporally adjacent to each other. In the STG, every interaction is shown by the presence of an edge connecting the nodes containing the respective shots.

In most scenes, there are recurring shots of the same element. Such recurrence can be recognized by similarities of visual contents. The clustering process then groups these similar shots into corresponding clusters based on the visual indices. The STG built subsequently contains cycles and nodes with a high degree of incoming and outgoing edges, indicating the interactions of the shots presented in these clusters. In time-constrained clustering, given a sufficiently good time window T, any shots from different scenes cannot be grouped into the same category (cluster). This means that the groupings of shots based on visual similarity are localized to a scene, therefore, the interactions are confined to clusters of the same scene, except at the transition of two scenes. More precisely, shots from different scenes cannot be juxtaposed, except at the transition of one scene to the next.

In a STG, a *cut edge* connects two disjoint connected subgraphs, each of which is a *story unit*. We thus use a story unit to approximate a scene for the segmentation purposes. In a STG representation of video, the transition of one *story unit* to the next will always occur on a *cut edge*. This is because a cut edge represents a unique point of transition from a shot from an earlier story unit to the ensuing shot of the next story unit. The collection of cut edges then partitions the STG into disjoint connected subgraphs and represents all the transitions from one unit to the next. By detecting the cut edges, individual story units can be segmented out. The segmented story units are semantically meaningful because they reflect to a certain degree the actual scenes depicted and the composition and interactions of events in different scenes.

The time-window T optimally should be chosen to reflect the minimum temporal distance between two recurring instances of the same scene, such that shots in these two instances are not grouped together (we shall call such instance a scene by itself, or more precisely, a story unit). In the examples illustrated in Fig. 1a, T should be set to reflect the temporal duration of the interleaved scene, such that the shots in scene 3 (e.g., S7), will not be in the same clusters that contains shots in scene 1, even if the visual characteristics may be quite similar.

However, such knowledge is not readily available. In practice, T can be chosen to be a sufficiently large fixed number. Here we assume that there are sufficient differences in the characteristics of one scene and those of the next scene, and the differences exhibited are good enough to segregate the shots from these adjacent but distinct scenes. (This is often the case because scene changes, unlike shot changes, often involve a change of locale and setting, different casts, etc. which are well reflected by changes in visual characteristics.) A scene usually consists of several shots, thus lasts longer.

These assumptions of video characteristics and the subsequent segmentation procedures are not dependent on any model of the specific nature of individual programs, and are applicable to a fairly wide variety of programming types. We will show later that story units can be successfully extracted from sitcoms, movies, cartoons, and documentaries.

The Scene Transition Graph generated with temporal constraint of the sample sequence in Fig. 1a is shown in Fig. 3. In contrast to the STG shown in Fig. 1c, this STG explicitly depicts the flow of story. Cut edges that separate story units are marked e_1 and e_2.

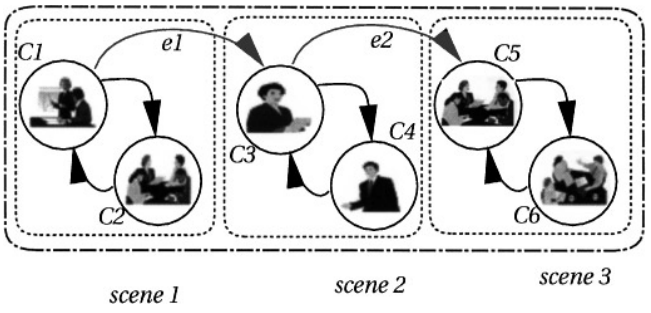

FIG. 3. Scene transition graph with temporal constraint.

6. SEGMENTATION OF VIDEO

In this section we will show a realization of the concepts of time-constrained clustering and scene transition graph presented in the previous sections to automatically parse a video program, extract story structures, and identify story units. The resulting graph representation presents not only the story structures, but also the flow of the story, as well as a clear view of meaningful units of the story segmented.

Given a video sequence, it has many shots and a few scenes. To access segments of interest of the video effectively and nonlinearly, there is the need to rediscover the composing shots and scenes, to find the structures of the document, to summarize the contents, and to present the structures and the contents to the users. More importantly, it is desirable to do most of the operations automatically.

The block diagram of the implementation is shown in Fig. 4. Here are the key steps listed in the diagram for analyzing the video programs and extracting meaningful story units. For illustration purposes, we provide a sample video sequence (R1) that consists of three scenes and nine shots and explain the results alongside the analysis steps. Individual components are discussed in the following sections. Note that in the diagram each shot is represented by an image. In the analysis process, however, the dynamics of the content variations in the shot are represented by a collection of *key frames* instead.

1. Shot segmentation. Shot transitions (shot boundaries) are detected to segment individual video shots. The algorithm detects the boundaries of shots based on the changes of visual characteristics and gives a list of the video shots (R2) of the sequence. For the sample sequence, nine shots (S1 to S9) are segmented out.

2. Time-constrained clustering of video shots. Video shots are partitioned into distinct clusters based on visual similarities and temporal constraints. Given the list of video shots (R2), shots are matched and clustered based on their visual contents and temporal localities to further condense repetitive shots of similar contents. For example, the nine shots are matched and grouped to give 6 clusters in R3 (C1 to C6).

3. Building of scene transition graph. A scene transition graph is built from the clustering results and the temporal relationships of the shots in the clusters. A node represents a cluster and an edge shows the flow of story from one node to the next. The STG of the sample video is shown as R4.

4. Scene segmentation. Story units (or scenes) are extracted by finding the *cut edges* of the STG. Each unit is a connected subgraph by itself. In R4, two cut edges (e_1 and e_2) are found, thus segmenting the video sequence into three story units. Each story unit indeed corresponds to the events taking place at a specific locale, that is, a *scene* in the film terms.

The analysis steps do not require knowledge of video programming type and have been tested on a variety of programs with promising results. The steps are generic to both compressed and uncompressed videos; however, for compressed video, reduced image sequences are extracted directly from the compressed data stream upon which the subsequent analyses are based. The ability to carry out the analyses on compressed video is important because many long programs are captured and stored directly in compressed formats like MPEG due to their inherent size of data. In addition, using reduced data for processing reduces computation time.

We use the algorithms in [20] for the extraction of reduced images from MPEG compressed video. The reduced images capture well the important global image features. They then form the basis for the different steps described above. In particular, the detection of different shot boundaries can be efficiently and effectively carried out on reduced images [5]. The individual shots, once detected, are then used for shot matching and clusterings.

7. REFINED ANALYSIS OF SEGMENTED STORY UNITS

The segmentation of story units is a step toward achieving accurate segmentation of different scenes for story organization beyond the shots. Given human recognition capability, and in the absence of understanding of the video, two scenes can be differentiated clearly from each other if they exhibit somewhat distinct visual characteristics, for example, an indoor scene followed by an outdoor scene. The human recognition capability is the upper limit of what algorithmic and systematic analysis can achieve. So it is reasonable to assume that two consecutive scenes in a video presentation do not share significant similarities in visual (perceptual) qualities.

The ability of the segmented story units to reflect closely the actual scenes relies on the ability of the time-constrained clustering to differenciate video shots, such that shots from different scenes are not grouped in the same cluster. In other words, the cluster labels given to the shots in a particular scene must reflect well both the visual similarity and temporal locality within the scene. In practice, the labeling is a difficult problem. There are two parameters in the time-constrained clustering: δ, which determines the clusters (the shots in each cluster and the number of clusters) based on visual dissimilarity, and T, which is the time-window parameter. In the clustering process, the values of these two parameters are predetermined. A fixed δ may reflect the

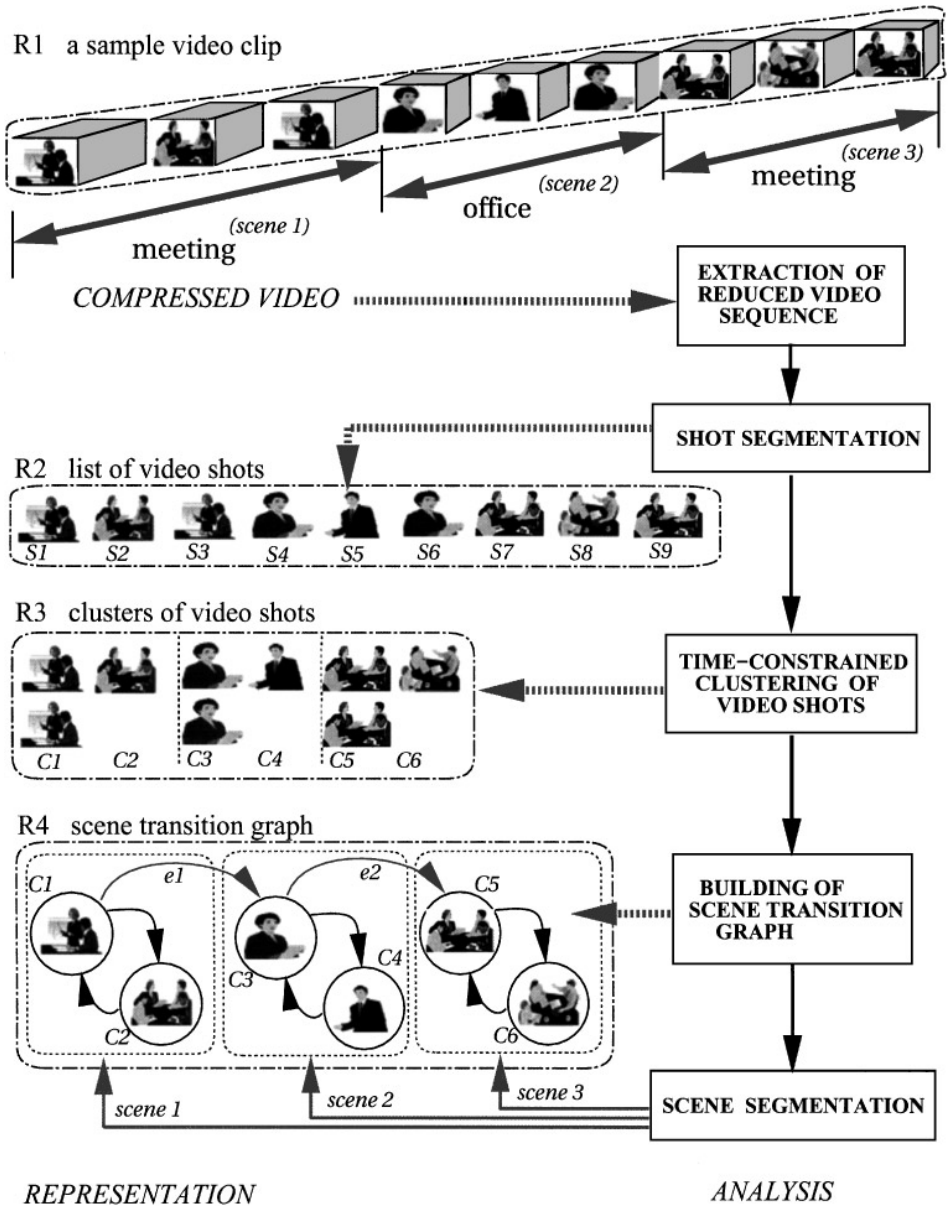

FIG. 4. Block diagram of proposed framework.

similarity of visual characteristics uniformly across various shots from different sequences to a large degree; any two shots with dissimilarity value above the threshold cannot be in the same cluster. This somewhat resembles the human perceptual system. When someone is asked to determine if two images are similar or not, he or she must set some subjective thresholds, though the thresholds may not be strictly based on visual characteristics like color and shape, but rather the semantic understanding of the images. On the other hand, without the knowledge of how long each individual scene lasts, T cannot be approximated well. A T too large can render shots from different scenes to cluster together, while a T too small can cause similar shots in the same scene to have different labels. In the segmentation of the story units, the former leads to the classification of two distinct scenes into one story unit, while the latter potentially means that a scene is broken down into several story units.

We want the story units segmented to closely approximate actual scenes. On the other hand, we believe that there may never be any automatic techniques that can achieve 100% accuracy in scene segmentation for a variety of video sequences. This is why given inaccurate segmentation, we prefer over-segmentation rather than under-segmentation. In other words, for many applications like browsing, it is less detrimental to have several story units represent a scene, and present all these units, than to have one story unit represent several scenes—these scenes cannot be recovered in subsequent analysis. This is why

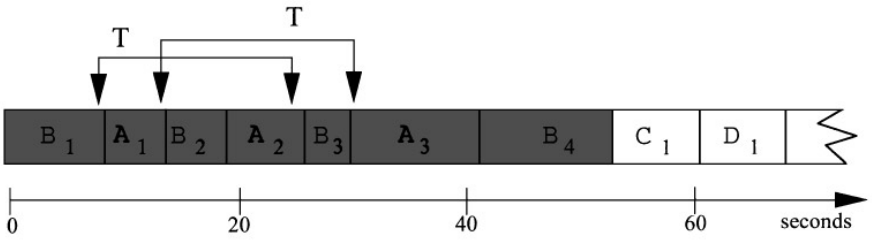

FIG. 5. A sequence of video shots and time-window parameter T.

given no knowledge on the length of individual scenes, we set T to a fixed value that reflects a reasonably short duration of time prior to the analysis. In our experiments, we set the time-window T to be approximately 100 s. A shot on average lasts 4–5 s, thus this means that the clustering is local to about 20 to 25 shots on average. This in general is shorter than the duration of a scene, and thus we may potentially have a few contiguous story units extracted from the initial segmentation representing the dramatic incidents in the same scene.

An illustration of a sequence of shots of a dialogue scene between Alice (A) and Bob (B) is shown in Fig. 5. We use the notation A_i to indicate the ith occurrence of a shot featuring Alice and B_i to indicate the ith occurrence of a shot of Bob. C's and D's are shots in another scene. The following example are based on this diagram.

Assume that all shots of the same person are clustered together by the hierarchical clustering algorithm in Section 3.3 with a predetermined visual dissimilarity threshold $\delta = \delta^*$ when there are no time-constraints, i.e.,

$d(A_i, A_j) \leq \delta^*$ and $d(B_i, B_j) \leq \delta^*$ for possible pairs (i, j). In the example the time-window $T = T^* = 20$ s. The shot duration is proportional to length in time scale; e.g., $d_t(B_1, B_3) > T^*$.

EXAMPLE 1. Suppose B_1 and B_2 have the least dissimilarity, i.e., $d(B_1, B_2) = \min d(B_i, B_j)$. This means that B_1 and B_2 are merged into one cluster in the first clustering step.

The clustering results are $\{B_1, B_2\}, \{A_1, A_2, A_3\}, \{B_3, B_4\}, \{C_1\}, \{D_1\}$. $\{B_i\}$ are not clustered into one cluster because there are at least a pair of shots, one from each cluster, that has a temporal distance $d_t > T^*$. There are three story units, $\{B_1, A_1, B_2, A_2, B_3, A_3, B_4\}, \{C_1\}, \{D_1\}$. In this case, even when the clustering does not group the shots of Bob into one cluster, the dialogue scene is still segmented out as one story unit.

EXAMPLE 2. Suppose B_2 and B_3 have the least dissimilarity, i.e., $d(B_2, B_3) = \min d(B_i, B_j)$. This means that B_2 and B_3 are merged into one cluster in the first clustering step. Thus, B_1 and B_4 cannot be grouped into the same cluster because each of them has a temporal distance $d_t > T^*$ from a shot (B_3 and B_2, respectively) in the cluster.

The clustering results are $\{B_1\}, \{B_2, B_3\}, \{A_1, A_2, A_3\}, \{B_4\}, \{C_1\}, \{D_1\}$. There are five story units, $\{B_1\}, \{A_1, B_2, A_2, B_3, A_3\}, \{B_4\}, \{C_1\}, \{D_1\}$. In this case, the dialogue scene is segmented out as three contiguous story units.

To achieve a better approximation of scene boundaries, we proceed to analyze the contiguous story units after the initial segmentation and check whether they reflect the dramatic elements in the same scene. We compute the duration of each story unit and use this information to improve the time-window parameter. In other words, T can be adjusted to reflect the durations of the story units—it can be made *elastic* to adapt to scene characteristics after the initial segmentation of the story units. This is a multilevel refinement of the segmentation process. In addition to merging contiguous story units that reflect the same dramatic incident, we can also achieve a better clustering of the shots in each story unit by integrating clusters of similar shots into one and relaxing the temporal constraints within the story unit. The goal is to push the limits of detecting story units that truly resemble the actual scenes and to obtain a compact representation that describes succinctly yet meaningfully the story contents.

This refined step of analysis is described as follows: Given the initial segmentation of a video into K story units $\{\mathcal{U}_i\}_{i=1}^{K}$. Let $\mathcal{U}_i$ be the ith story unit represented by the subgraph $\mathcal{G}_i$. It is a collection of contiguous shots, i.e.,

$$\mathcal{U}_i = \bigcup_{j=FS(\mathcal{G}_i)}^{LS(\mathcal{G}_i)} \{S_j\}, \quad i \in \{1, 2, \ldots, K\}.$$

Denote the duration of $\mathcal{U}_i$ by $\tau(\mathcal{U}_i)$. Assume at the initial analysis stage that the clustering threshold $\delta = \delta^*$ and the time-window parameter T is preset to some fixed vaue. Let i be the index of initial story units, m be the new index, and $\mathcal{U}'$ denote the refined story unit; i.e., $\mathcal{U}'_m$ is the mth refined story unit.

[**Refined Analysis of Story Units**]

1. Set $i \leftarrow 1$. Set $m \leftarrow 1$.
   Relax $T : T \leftarrow \tau(\mathcal{U}_1)$.
   Do a clustering on all the shots $S_j \in \mathcal{U}_1$ with (T, δ^*).
   $\mathcal{U}'_1 \leftarrow \mathcal{U}_1$.
2. $i \leftarrow i + 1$.
   If $i \leq K$
   (a) Compute the duration $\tau(\mathcal{U}'_m)$ and $\tau(\mathcal{U}_i)$.

Relax $T: T \leftarrow \tau(\mathcal{U}'_m) + \tau(\mathcal{U}_i)$.
Do a clustering on all the shots $S_j \in \mathcal{U}'_m \cup \mathcal{U}_i$ with (T, δ^*).

(b) If there is one cluster containing a shot from $\mathcal{U}'_m$ and a shot from $\mathcal{U}_i$.
$\mathcal{U}'_m \leftarrow \mathcal{U}'_m \cup \mathcal{U}_i$. Goto (2).

(c) Otherwise, relax $T: T \leftarrow \tau(\mathcal{U}_i)$.
Do a clustering on all the shots $S_j \in \mathcal{U}_i$ with (T, δ^*).
$m \leftarrow m + 1$.
$\mathcal{U}'_m \leftarrow \mathcal{U}_i$. Goto (2).

3. Return the new story units $\{\mathcal{U}'_1, \mathcal{U}'_2, \ldots, \mathcal{U}'_m\}$.

Given a story unit, the refinement involves the examination of the next story unit by relaxing the temporal window and reclustering the shots in these two units with the original visual dissimilarity threshold. If there exists at least one new cluster that contains shots from the two units, this means that the new subgraph constructed does not have any cut edges and the two story units are merged into one. The merging of story units continuous until the consecutive unit does not exhibit this property. Step 2(a) effectively reclusters the shots in the new story unit which is now a conglomerate of several old units, and Step 2(c) reclusters the shots in a single old unit with T set to be the duration of the story unit. The reclustering Steps 1 and 2(c) guarantee that even in cases where the boundaries of a story unit do not change, similar shots are clustered together rather than in several clusters because of fixed time constraints, and hence achieve a more compact structure.

In Example 1, the refined clustering results are

$$\{B_1, B_2, B_3, B_4\}, \{A_1, A_2, A_3\}, \{C_1\}, \{D_1\}.$$

The three story units $\{B_1, A_1, B_2, A_2, B_3, A_3, B_4\}, \{C_1\}, \{D_1\}$ remain the same after the refinement.

In Example 2, the refined clusters are

$$\{B_1, B_2, B_3, B_4\}, \{A_1, A_2, A_3\}, \{B_4\}, \{C_1\}, \{D_1\}.$$

The five story units $\{B_1\}, \{A_1, B_2, A_2, B_3, A_3\}, \{C_1\}, \{D_1\}$ now become three units with the first three merged into one,

$$\{B_1, A_1, B_2, A_2, B_3, A_3, B_4\}, \{C_1\}, \{D_1\}.$$

After the analysis, the dialogue scene is segmented out as the first story unit.

Compared to time-constrained clustering, refinement analysis involves reclustering of shots across adjacent story units and thus is a form of time-constrained clustering with larger time-window T. In practice, the duration of a story unit is much larger than the parameter T used in time-constrained clustering. The complexity will be between that of time-constrained clustering and of the clustering of all shots without any time constraints at all. For a sequence with N shots, the complexity will be on the order of N^2/K shot comparisons on the average. The fraction $1/K$ is the reduction factor from the full all-pair shot comparison without any time constraints.

8. RESULTS AND DISCUSSION

This section describes experimental results of the analyses applied to several test programs. The results of the segmentation of video sequences into story units using time-constrained clustering and STG analysis are presented in Section 8.1. In Section 8.2, we investigate the two parameters, T and δ, used in the time-constrained clustering process. We present results on the refinement of segmentation results in Section 8.3.

In the experiments, we use test sequences drawn from a variety of TV programs and movies, such as talk shows, sitcoms, documentaries, news, as well as segments of different movies. It should be noted that the commercials (in TV programs) have been edited out prior to the analysis.

8.1. Segmentation of Video

Figure 6 shows the STG[4] constructed from an episode of the sitcom "Friends." It is constructed using $T = 2000$ and $\delta = 0.3$. There are 35575 frames, each at a spatial resolution of 320×240. In this episode, there are 313 shots. From Fig. 6, we can see clearly the individual story units. Other graphs constructed for several test sequences are illustrated in Figs. 7 and 8.

The examples illustrate that time-constrained clustering of video shots is able to identify individual story units. Clustering procedures separate shots that are visually dissimilar into different clusters. The time constraints prevent the grouping of visually similar shots across different scenes into the same cluster. This way, recurring scenes can be segregated. Each highlighted edge joins the subgraphs together and presents the flow of units in sequential order. In doing so, it chains the story units, linking one unit to the next as the story flows. The story structure is presented concisely and meaningfully this way. The segmentation of contents achieved corresponds well to the perceived story segments of the program.

In addition, the resulting STG permits rapid nonlinear browsing of long video programs. The half-hour episodes represented in Figs. 6 and 7 are each succinctly condensed to a graph which can be displayed in a screen. A user can quickly identify and zoom into segments of interests without having to *linearly* view through the entire program.

A summary of the results is shown in Table 1. The tested sequences were digitized at 30 frames/s, with the exception of the "Democratic Convention 1992" which was digitized at 15 frames/s. The sitcoms are half-hour episodes (the commercials have been edited out). For the results presented in Table 1, the time window duration is the same for all test sequences: we use 100-s duration as the parameter. This translates to using

[4] Unfortunately, because of copyright considerations, we cannot show the images associated with each node. The correspondences between story units identified by the graph and the scenes for this sequence are shown in Table 3.

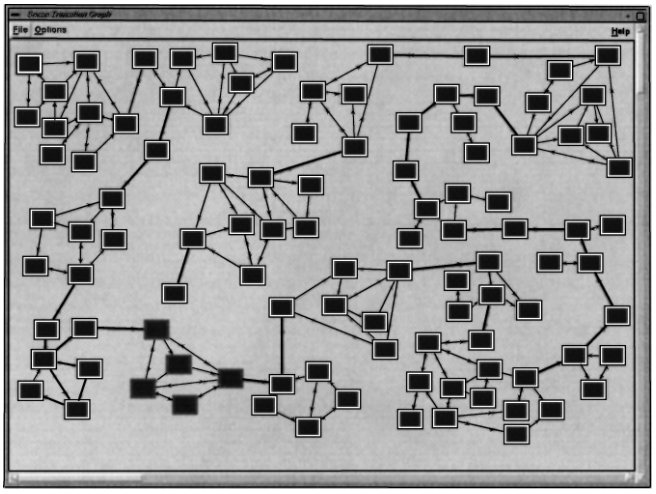

FIG. 6. STG with time-constrained clustering for "Friends" (sitcom). A selected story unit and the cut edges are highlighted. The image associated with each node cannot be shown because of copyright considerations.

$T = 3000$ for sequences digitized at 30 frames/s, and $T = 1500$ for sequences digitized at 15 frames/s. The clustering parameter $\delta = 0.3$. The results are used to illustrate the hierarchical decomposition of the video programs. Such an hierarchy, from frames, shots to story units, allows multilevel organization of video.

8.2. Variation of Clustering Parameters

We study the effect of varying the two parameters, T and δ, in the time-constrained clustering algorithm on the test sequence "Friends." The accuracy of segmentation and the number of story units are evaluated.

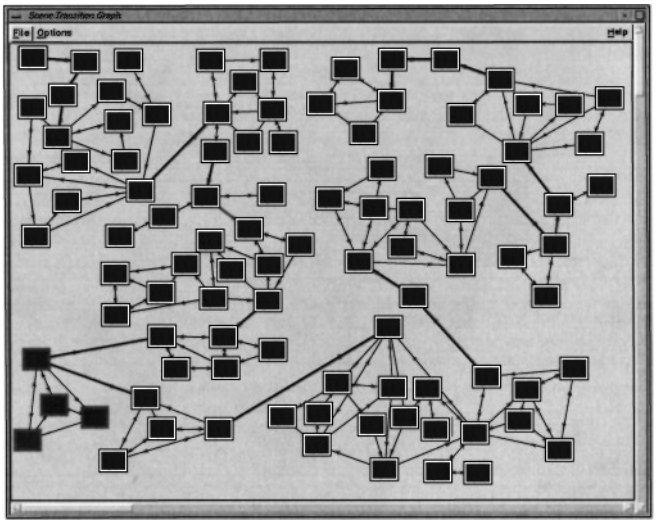

FIG. 7. STG with time-constrained clustering for "Frasier" (sitcom). A selected story unit and the cut edge are highlighted. The image associated with each node cannot be shown because of copyright considerations.

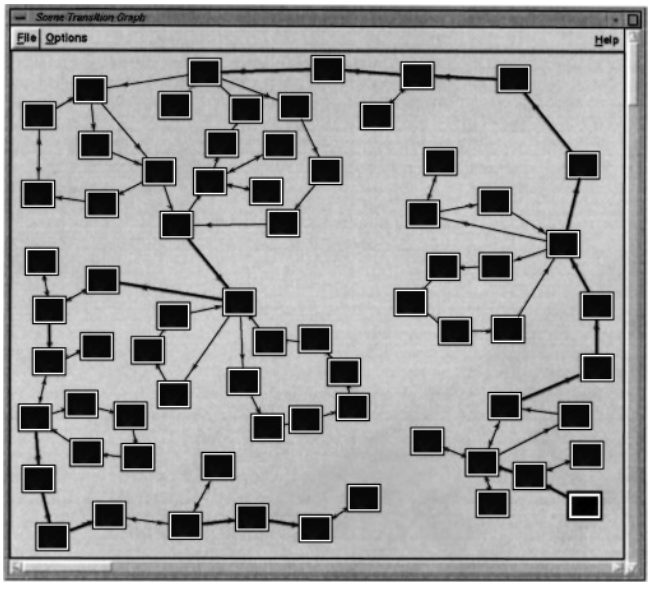

FIG. 8. STG with time-constrained clustering for "Dances with Wolves" (movie segment). The image associated with each node cannot be shown because of copyright considerations.

Table 2 tabulates the number of story units of "Friends" using the time-constrained clustering and cut-edge detection, with $T = 1000$, 2000, and 3000 and $\delta = 0.3$ and 0.5. The first column lists the major scenes as perceived by the authors. The second column lists the corresponding locations where each of the scene takes place. There are occasions when a scene takes place in more than one location (e.g., Scene 6). Locations "Street" are transition shots of the streets that lead into the ensuing location.

Columns 3 through 8 record the segmented story units. The notation is as follows: U1 means story unit 1. Thus, when $T = 1000$ and $\delta = 0.5$, the first scene is made up of two story units (U1 and U2), the second scene is made up of one story unit (U3) and so on. The entries marked (*) indicate those scenes which the algorithm fails to segment. For example, when $T = 3000$, $\delta = 0.3$, the story unit (U3) found spans Scenes 3 and 4.

TABLE 1
Results of Sample Test Sequences

Type	Sequence name	Number of frames	Number of shots	Number of clusters	Number of story units
Sitcom	"Friends"	35575	313	97	18
Sitcom	"Frasier"	37427	318	99	19
Cartoon segment	"Aladdin"	23251	188	78	22
Movie segment	"Dances with Wolves"	21400	140	70	19
Documentary	Democratic convention '92	14398	68	38	19

TABLE 2
Table Illustrating the Relationships of Segmented Story Units to Actual Story for "Friends—The One with the List," Using Different Values of T and δ

Scene	Location	$T = 1000$		$T = 2000$		$T = 3000$	
		$\delta = 0.3$	$\delta = 0.5$	$\delta = 0.3$	$\delta = 0.5$	$\delta = 0.3$	$\delta = 0.5$
1	Central Perk Cafe	U1–U2	U1–U2	U1–U2	U1–U2	U1	U1
2	Office	U3	U3	U3–U4	U3	U2	U2
3	Monica's apt.	U4–U6	U4	U5–U7	U4	U3	U3
4	Chandler's apt.	U7–U10	U5–U7	U8	U5	U3*	U3*
5	Monica's apt.	U11	U8	U9	U6	U4	U4
6	Chandler's apt.	U12–U19	U9–U13	U10–U12	U7–U8	U5	U5
	At the door	U20	U14	U13			
7	Central Perk Cafe	U21–U23	U15	U14–U16	U9	U6–U8	U5*
8	Street	U24	U16	U17		U9	
	Monica's apt.	U25–U31	U17–U20	U18–U23	U10	U10–U14	U5*
9	Street	U32		U24		U15	
	Office	U33–U35	U21–U22	U25	U10*	U16	U6
10	Street						
	Monica's and Ross's apts. (alternating)	U36–U40	U23–U25	U26–27	U11–U12	U17–U18	U6*–U7

* Scenes which the algorithm failed to segment.

Smaller δ values result in more clusters and thus more story units in general, while large values may cause shots to be clustered together even when the visual contents are significantly different. Similarly, as the time parameter T decreases, the algorithm is effective at segmenting story units even when δ varies to some degree. The trade-off is that we tend to obtain more story units. For example, when $\delta = 0.3$, there are 40 story units when $T = 1000$, 27 units when $T = 2000$, and 18 units when $T = 3000$. In browsing applications, it is often more devastating to fail to segment distinct scenes than to over-segment a scene into smaller units. If we consider the failure of the algorithm as the failure to segment distinct scenes, then choosing a smaller T will ensure the performance of satisfactory segmentation over a wide range of δ's. If the performance also takes into account over-segmentation, then a combination of right ranges of the two parameters are necessary to achieve satisfactory results. We found that $T = 2000$ with $\delta \leq 0.5$, and, $T = 3000$ with $\delta \leq 0.3$ work well for a variety of video programs to segment out the major story units. In addition, the choice of parameters does not depend on the specific programs tested.

In Table 2, for $\delta = 0.3$, there is single story unit found when $T = 1000$ and $T = 3000$, but two story units found when $T = 2000$. Such anomaly does not occur frequently in practice, but is rather of a consequence of the hierarchical clustering procedure based on Complete-link Method. This anomaly can be explained by the interactions of three visually similar shots S_i, S_j, and S_k, in increasing order of time. Furthermore, the relation $d_t(S_i, S_j) < 1000$, $1000 < d_t(S_j, S_k) < 2000$, and $d(S_j, S_k) < d(S_i, S_j) < d(S_l, S_m)$ for all other pairs of shots (S_l, S_m). When $T = 2000$, S_j is clustered together with S_k, and S_i is not clustered with any other shots. Thus, two story units result. When $T = 1000$, S_i and S_j are clustered together instead, and the interactions among other shots result in a single story unit. Our observation is that this anomaly is rare in practice, and generally, increasing T will result in fewer story units. On the other hand, it is easy to show that when T is fixed, increasing δ will always result in fewer or same number of story units.

Many of the story units obtained from the segmentation process reflect story contents of the actual scenes. We can compute the duration of each story unit and this offers further insight into the segmentation results. Table 3 lists the durations of the actual scenes and extracted story units for the test sequence "Friends" using $T = 2000$ and $\delta = 0.3$. The segmentation gives story units of a wide range of durations, from a segment of less than 2 s (e.g., U5 lasts 49 frames, 1.6 s) to segments of more than 2 min (e.g., U12 lasts 4180 frames, 2 min 20 s). Story units of short durations very often consist of a single shot which is commonly found to be an establishing shot of a particular locale or is used as a lead-in to the next scene. On the other hand, longer scenes may be cut to several story units because of the limitations posted by preset time-window parameter.

8.3. Refining the Segmentation Results

The preset time-window T, may lead to the over-segmentation of the story because similar shots may not be grouped into same clusters. Multilevel refinements on the analysis steps described in Section 7 are performed on contiguous story units by computing the duration of each story unit after the initial segmentation, relaxing the time-window parameter T to adopt to the durations of the units, and reclustering the shots. Table 4 shows the results of story unit segmentation for the test sequence "Friends" after the initial analysis with the results listed in Table 3.

TABLE 3
Duration of Scenes and Associated Story Units in Number of Frames for "Friends," with $T = 2000$, $\delta = 0.3$

Scene	Duration	Locale	Story unit	Duration
1	5562	Central Perk Cafe	U1	3390
			U2	2172
2	3097	Office	U3	143
			U4	2954
3	1176	Monica's apt.	U5	49
			U6	1047
			U7	80
4	3387	Chandler's apt.	U8	3387
5	754	Monica's apt.	U9	754
6	9530	(a) Chandler's apt.	U10	2219
			U11	2227
			U12	4180
		(b) At the door	U13	904
7	1409	Central Perk Cafe	U14	180
			U15	573
			U16	656
8	4865	(a) Street	U17	75
		(b) Monica's apt.	U18	1026
			U19	89
			U20	85
			U21	613
			U22	156
			U23	2821
9	1612	(a) Street	U24	65
		(b) Office	U25	1547
10	4183	(a) Street	U26	3275
		(b) Monica's and Ross's apts.	U27	908

Figure 9 shows the STG constructed after the refinement. The threshold used is $\delta = 0.3$. The number of clusters reduce from 105 before refinement to 92 after refinement. After refinement, the first two story units in Scene 1 are merged into one, reflecting the scene at the Central Perk Cafe. The number of story units in Scene 6 is reduced from 4 to 2, showing the scene inside Chandler's apartment. As expected, the reclustering process leads to better clustering in some story units whose boundaries remain the same and offers a more compact structure. This way T is made to adapt to scene characteristics and we can improve the segmentation results.

TABLE 4
Table Showing the New Segmentation Results with Refined Analysis for "Friends"

Scene	Locale	$\delta = 0.3$	
		$T = 2000$	T Elastic
1		U1–U2	U1
2		U3–U4	U2–U3
3		U5–U7	U4–U6
4		U8	U7
5		U9	U8
6	(a)	U10–U12	U9–U10
	(b)	U13	
7		U14–U16	U11–U13
8	(a)	U17	U14
	(b)	U18–U23	U15–U20
9	(a)	U24	U21
	(b)	U25	U22
10	(a)		
	(b)	U26–U27	U23–U24

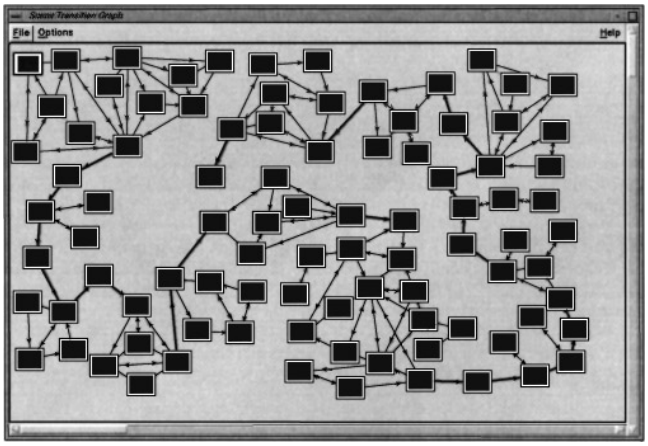

FIG. 9. The STG constructed showing the merging of story units and reclustering results using elastic T for "Friends." The image associated with each node cannot be shown because of copyright considerations.

In general, refinement will merge two neighboring story units with similar visual characteristics. In the "Friends" example, unlike story units in Scenes 1 and 6, story units in other scenes do not exhibit sufficiently similar visual characteristics and thus are not merged. On the other hand, if there exist cases when adjacent scenes have very similar visual characteristics, refinement could lead to merging of story units in two distinct scenes. In the "Friends" example, this does not occur, and in real video, it is typical that adjacent scenes will be set in different locales or about different events, thus inducing contrasting visual characteristics across adjacent scenes.

9. DISCUSSIONS AND CONCLUSIONS

The analysis of video contents based on time-constrained clustering and scene transition graph analysis has contributed to the extraction of story units that cannot be achieved by detection of shot boundaries alone. With a common set of parameters, we are able to segment out satisfactory and meaningful story units which represent distinct events or locales from several types of video programs. A video program is subsequently decomposed into an hierarchy of story units, each of which consists of clusters of similar shots, and within a cluster, there are visually similar shots. This offers better organization of video. In addition, the building of story structure provides a mean for nonlinear access to a featured program and facilitates browsing of video contents. In fact, the compact structure built form the analysis can serve as a form of *visual summaries* [23] of the video contents.

The clustering and segmentation techniques presented in this paper are in part based on low-level visual characteristics of video shots which represent *domain-independent* and *syntactic* features, and in part based on temporal story characteristics. We are able to successfully segment video into semantically meaningful units for a variety of programming types. In other words, we are able to derive temporal semantics from our analysis of syntactic features. A unit of the segmentation, the story unit, conveys by itself a meaning in the story. Other temporal features can include dialogues and actions [24] which can be extracted by analyzing temporal patterns in video. For example, a dialogue can be modeled by a set of dialogue-like patterns involving the interleaving of two or more dominant shots. The temporal structure of video, in other words, the montage presentation, is as important, and many times more important, in cultivating meanings and conveying the storyline, than the salient features in images like a face or a mountain, which are the semantic features known to the image analysis community. We do argue that such a class of temporal features are another form of semantic features that are unique to digital video and are beyond the semantic features that can be conveyed by analyzing image content. We do not, however, preclude the use of image semantics to further the analysis of video. On the contrary, we believe that the use of image semantics will significantly enhance the understanding of content and complement our temporal story analysis. For example, the use of *domain-dependent* features such as those used in parsing news broadcasts [10, 11] and *semantic* features can provide models for the segmentation tasks and to further improve segmentation accuracy. The identification, integration, and application of domain-dependent and semantic features in video analysis and the extraction of high-level structures warrant further research.

ACKNOWLEDGMENTS

The authors are grateful to Dr. Ruud Bolle of IBM T. J. Watson Research Center and the reviewers for their comments and suggestions. In particular, we thank Reviewer 1 for a very thorough and careful reading of the manuscript. This work was funded in part by the Intel Foundation Graduate Fellowship, the Wallace Memorial Fellowship in Engineering, and the Siemens Corporate Research. The support of NIST/ATP under Contract Number 70NANB5H1174 during the revision of the paper is also acknowledged. The computing facilities used in the research were supported by the National Science Foundation under Grant CDA-9216171.

REFERENCES

1. M. M. Yeung, B. L. Yeo, and B. Liu, Extracting story units from long programs for video browsing and navigation, in *International Conference on Multimedia Computing and Systems, IEEE, June 1996*, pp. 296–305.
2. M. M. Yeung and B. L. Yeo, Time-constrained clustering for segmentation of video into story units, in *International Conference on Pattern Recognition (ICPR'96), Aug. 1996*, Vol. C, pp. 375–380.
3. F. Arman, A. Hsu, and M. Y. Chiu, Image processing on compressed data for large video databases, in *Proceedings of First ACM International Conference on Multimedia, Aug. 1993*, pp. 267–212.
4. H. J. Zhang, A. Kankanhalli, and S. W. Smoliar, Automatic partitioning of full-motion video, *Multimedia Systems* **1**, July 1993, 10–28.
5. B. L. Yeo and B. Liu, Rapid scene analysis on compressed videos, *IEEE Trans. Circuits Systems Video Technol.* **5**(6), 1995, 533–544.
6. Andrey Tarkovsky, translated by Kitty Humer-Blaire, *Sculpting in Time—Reflections on the Cinema*, University of Texas Press, Austin, 1986.
7. K. D. Yow, B. L. Yeo, M. M. Yeung, and B. Liu, Analysis and presentation of soccer highlights from digital video, in *Second Asian Conference on Computer Vision, Dec. 1995*, Vol. II, pp. 499–503.
8. B. L. Yeo and B. Liu, Visual content highlighting via automatic extraction of embedded captions on MPEG compressed video, in *Digital Video Compression: Algorithms and Technologies 1996, Feb. 1996*, Vol. SPIE 2668, pp. 38–47.
9. M. A. Smith and T. Kanade, Video skimming for quick browsing based on audio and image characterization, Technical Report, CMU-CS-95-186, Carnegie Mellon University, July 1995.
10. D. Swanberg, C. F. Shu, and R. Jain, Knowledge guided parsing in video databases, in *Storage and Retrieval for Image and Video Databases, 1993*, Vol. SPIE 1908, pp. 13–25.
11. H. J. Zhang, Y. H. Gong, S. W. Smoliar, and S. Y. Yan, Automatic parsing of news video, in *International Conference on Multimedia Computing and Systems, 1994*, pp. 45–54.
12. M. M. Yeung and B. Liu, Efficient matching and clustering of video shots, in *International Conference on Image Processing, 1995*, Vol. I, pp. 338–341.
13. M. M. Yeung, B. L. Yeo, W. Wolf, and B. Liu, Video browsing using clustering and scene transitions on compressed sequences, in *Multimedia Computing and Networking 1995, Feb. 1995*, Vol. SPIE 2417, pp. 399–413.
14. F. Beaver, *Dictionary of Film Terms*, Twayne Publishing, New York, 1994.
15. Britannica Online, http://www.eb.com:180/eb.html, Nov. 1995.
16. S. Eisenstein, *The Film Sense*, Harcourt Brace & Company, New York, 1970.
17. F. Arman, R. Depommier, A. Hsu, and M. Y. Chiu, Content-based browsing of video sequences, in *ACM Multimedia 94, Aug. 1994*, pp. 97–103.
18. W. Niblack, R. Barber, W. Equitz, M. Flickner, E. Glasman, D. Petkovic, P. Yanker, C. Faloustos, and G. Taubin, The QBIC project: Querying images by content using color, texture and shape, in *Storage and Retrieval for Image and Video Databases, 1993*, Vol. SPIE 1908, pp. 13–25.
19. M. Strickter and M. Orengo, Similarity of color images, in *Storage and Retrieval for Image and Video Databases, 1995*, Vol. SPIE 2420, pp. 381–392.
20. B. L. Yeo, *Efficient Processing of Compressed Images and Video*, Ph.D. thesis, Princeton University, Electrical Engineering Department, Jan. 1996. [Available online at http://www.ee.princeton.edu/∼yeo/Thesis.]
21. A. K. Jain and R. C. Dubes, *Algorithms for Clustering Data*, Prentice Hall, New York, 1988.
22. J. A. Bondy and U. S. R. Murty, *Graph Theory with Applications*, Macmilliam, London, 1976.
23. M. M. Yeung, *Analysis, Modeling and Representation of Digital Video*, Ph.D. thesis, Princeton University, Electrical Engineering Department, July 1996.
24. M. M. Yeung and B. L. Yeo, Video content characterization for digital library applications, in *Proceedings, SPIE Storage and Retrieval for Still Image and Video Databases V, Feb. 1997*, Vol. SPIE 3022, pp. 45–58.