International Journal on Artificial Intelligence Tools
Vol. 10, No. 4 (2001) 000—000

IDENTIFYING OVERLAPPED OBJECTS FOR VIDEO INDEXING AND MODELING IN MULTIMEDIA DATABASE SYSTEMS

[1]SHU-CHING CHEN, [2]MEI-LING SHYU, [1]CHENGCUI ZHANG and [3]R. L. KASHYAP

[1] *Distributed Multimedia Information System Laboratory*
School of Computer Science, Florida International University, Miami, FL 33199
chens, czhang02@cs.fiu.edu

[2] *Department of Electrical and Computer Engineering, University of Miami,*
Coral Gables, FL 33124
shyu@miami.edu

[3] *School of Electrical and Computer Engineering, Purdue University,*
West Lafayette, IN 47907
kashyap@ecn.purdue.edu

Received (received date)
Revised (revised date)

The identification of the overlapped objects is a great challenge in object tracking and video data indexing. For this purpose, a backtrack-chain-updation split algorithm is proposed to assist an unsupervised video segmentation method called the "simultaneous partition and class parameter estimation" (SPCPE) algorithm to identify the overlapped objects in the video sequence. The backtrack-chain-updation split algorithm can identify the split segment (object) and use the information in the current frame to update the previous frames in a backtrack-chain manner. The split algorithm provides more accurate temporal and spatial information of the semantic objects so that the semantic objects can be indexed and modeled by multimedia input strings and the multimedia augmented transition network (MATN) model. The MATN model is based on the ATN model that has been used in artificial intelligence (AI) areas for natural language understanding systems, and its inputs are modeled by the multimedia input strings. In this paper, we will show that the SPCPE algorithm together with the backtrack-chain-updation split algorithm can significantly enhance the efficiency of spatio-temporal video indexing by improving the accuracy of multimedia database queries related to semantic objects.

Keywords: Multimedia Augmented Transition Network (MATN), Multimedia Input String, Backtrack-Chain-Updation Split Algorithm, and Multimedia Database Systems.

1. Introduction

As more information sources become available in multimedia systems, the need for efficient modeling, browsing, searching, and retrieving information increases, especially for video data. Digital video has been widely used in many multimedia applications such as education and training, video on demand, and video confer-

encing. Instead of sequential access to the video contents, how to structure and model video data so that users can quickly and easily browse and retrieve interesting materials becomes an important issue in designing multimedia information systems [1]. To better model and structure video data, a multimedia augmented transition network (MATN) model that is based on the augmented transition network (ATN) together with its inputs, the multimedia input strings, were proposed to model and structure video data [2]. The augmented transition network (ATN), developed by Woods [3], has been used in natural language understanding systems and question answering systems for both text and speech. Multimedia input strings are similar to regular expressions [4] and are used to represent the presentation sequences of temporal media streams, spatio-temporal relations of semantic objects, and keyword compositions. In our previous studies, we have already shown that the MATN model and multimedia input strings could model the multimedia presentations, multimedia browsing, multimedia database searching, and the temporal, spatial, or spatio-temporal relations of various media streams and semantic object [2, 5]. Moreover, a multimedia information system should allow users to retrieve their interested video materials via database queries. To answer multimedia database queries related to the temporal or relative spatial positions of semantic objects, it is necessary to have object-based representation of video data. For this purpose, more and more attention has devoted to segmenting video frames into regions such that each region, or a group of regions, corresponds to an object that is meaningful to human viewers [6, 7]. Therefore, key frame selection based on object-based representation is an important issue for video data [1, 8, 9, 10, 11]. A key frame selection approach that is based on the number, temporal, and spatial changes (represented by multimedia input strings) of the semantic objects in the video frames was developed in [2]. The proposed key frame selection approach was based on the temporal and spatial relations of semantic objects in each shot and employed the simultaneous partition and class parameter estimation (SPCPE) algorithm [12] to facilitate the MATN model. Using the SPCPE algorithm, MATN, and multimedia input strings, the temporal and spatial relations of semantic objects are captured and modeled. Compared with previous works such as histogram-based methods [13], clustering- based methods [14] and region-growing methods [15], the SPCPE algorithm is a stochastic model-based method that models the image classes as random fields and poses the segmentation problem as a statistical optimization problem. However, computing the exact MAP estimate of the class label field is considered a hard problem. Central to SPCPE method is the formulation of a cost functional defined on the space of image partitions and the class description parameters that can be minimized in a simple manner. In the SPCPE method, the class description model is chosen in such a way so as to be able to estimate the class parameters directly without resorting to numerical optimization techniques. The proposed class parameterization that enabled us to compute the optimal parameters by using a simple least squares technique is general enough to incorporate the behavior of various classes. Moreover, we can directly estimate the parameters of

the class descriptions without resorting to expensive numerical optimization procedures. The class label field estimation problem is also posed as that of estimating the data partition. By considering both the partition and the class parameters as random variables and estimating them jointly, we can simultaneously compute their MAP estimates. Furthermore, in SPCPE, each frame is partitioned by using the partition of the previous frame as an initial condition. So the correspondence problem need not be addressed explicitly since the information from the previous frame essentially guides the partitioning of the current frame. This approach effectively tracks objects that undergo translations and even rotations that are not within the image plane. Associating the obtained segments in adjacent frames to get a single tube is then a simple matter of linking the nearest segments depending on the nearness of their centroids. However, the SPCPE algorithm is not able to identify the objects that are overlapped together within the video frames. In other words, the SPCPE algorithm cannot distinguish the overlapped objects, which may cause inaccurate answers to multimedia queries. Hence, if we can extend the functionality of the algorithm to allow the identification of the overlapped objects, more accurate information can be obtained to answer more detailed multimedia queries. In this paper, we propose a backtrack-chain-updation split algorithm that can find the split segment (object) and use the information in the current frame to update the previous frames in a backtrack-chain manner. This split algorithm is an enhancement of the SPCPE algorithm to allow the distinguishing of two separate objects that were overlapped previously. That is, the proposed split algorithm provides more accurate temporal and spatial information of the semantic objects for video indexing. Hence, this algorithm can enhance the efficiency of the current MATN model by improving the accuracy of multimedia database queries related to the relative spatial positions of the semantic objects. Moreover, the proposed split algorithm is applied to the air-show video sequence, people-walking video sequence, and soccer game video sequence. The experimental results show that the proposed algorithm can recover the overlapped situation successfully and automatically without any user intervention. This paper is organized as follows. Section 2 gives a brief review of how to use MATNs and multimedia input strings to model video indexing. Section 3 discusses the mechanism of the proposed backtrack-chain-updation split algorithm. Conclusions are given in Section 4.

2. Using MATNs and Multimedia Input Strings for Video Indexing

As mentioned earlier, the MATNs and multimedia input strings are used to model the temporal and relative spatial relations of various media streams and semantic objects, multimedia database searching, multimedia browsing, and multimedia presentations [5]. Multimedia input strings are the inputs for the MATNs and represent the presentation sequence of temporal media streams, spatio-temporal relations of semantic objects, and keyword compositions. Moreover, a key frame selection approach based on the temporal and spatial relations of semantic objects is adopted

as low level indexing for video streams [2]. The unsupervised video segmentation method, i.e., the SPCPE algorithm, can obtain the semantic object information required in the key frame selection approach. In the following subsections, we will first give an overview of the SPCPE algorithm, then briefly describe how to use MATNs and multimedia input strings to model video key frames. A portion of the soccer video clips is used to demonstrate how video indexing is modeled by the MATNs and multimedia input strings.

2.1. *Overview of the SPCPE Algorithm*

The SPCPE (Simultaneous Partition and Class Parameter Estimation) algorithm is an unsupervised video segmentation method to partition video frames. In the SPCPE algorithm, the partition and the class parameters are treated as random variables. The method for partitioning a video frame starts with an arbitrary partition and employs an iterative algorithm to estimate the partition and the class parameters jointly [12]. Since the successive frames in a video do not differ much, the partitions of adjacent frames do not differ significantly. Each frame is partitioned using the partition of the previous frame as an initial condition so the number of iterations in processing can be greatly reduced. A randomly generated initial partition is used for the first frame since there is no previous frame available.

Suppose we have two classes. Let the partition variable be $\boldsymbol{c} = \{\boldsymbol{c}_1, \boldsymbol{c}_2\}$ and the classes be parameterized by $\boldsymbol{\theta} = \{\boldsymbol{\theta}_1, \boldsymbol{\theta}_2\}$. Also, suppose all the pixel values y_{ij} (in the image data Y) belonging to class k ($k = 1, 2$) are put into a vector Y_k. Each row of the matrix Φ is given by $(1, i, j, ij)$ and a_k is the vector of parameters $(a_{k0}, ..., a_{k3})^T$.

$$\begin{aligned} y_{ij} &= a_{k0} + a_{k1}i + a_{k2}j + a_{k3}ij, \forall (i,j)\ y_{ij} \in \boldsymbol{c}_k \\ \mathbf{y_k} &= \Phi a_k \\ \hat{a_k} &= (\Phi^T \Phi)^{-1} \Phi^T \mathbf{y_k} \end{aligned}$$

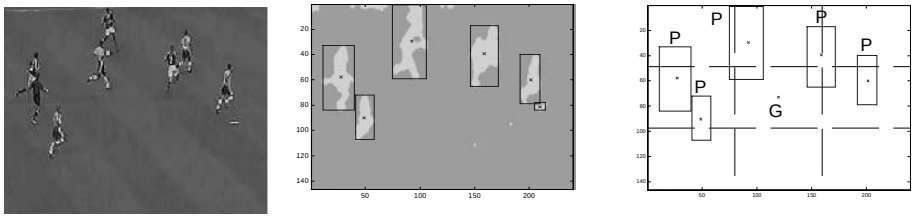
We estimate the best partition as that which maximizes the a posteriori probability (MAP) of the partition variable given the image data Y. Now the MAP estimates of $\boldsymbol{c} = \{\boldsymbol{c}_1, \boldsymbol{c}_2\}$ and $\boldsymbol{\theta} = \{\boldsymbol{\theta}_1, \boldsymbol{\theta}_2\}$ are given by

$$\begin{aligned} (\hat{\boldsymbol{c}}, \hat{\boldsymbol{\theta}}) &= \operatorname*{Arg\,max}_{(\boldsymbol{c}, \boldsymbol{\theta})} P(\boldsymbol{c}, \boldsymbol{\theta} \mid Y) \\ &= \operatorname*{Arg\,max}_{(\boldsymbol{c}, \boldsymbol{\theta})} P(Y \mid \boldsymbol{c}, \boldsymbol{\theta}) P(\boldsymbol{c}, \boldsymbol{\theta}) \end{aligned}$$

Let $J(\boldsymbol{c}, \boldsymbol{\theta})$ be the functional to be minimized. With appropriate assumptions, this joint estimation can be simplified to the following form:

$$(\hat{\boldsymbol{c}}, \hat{\boldsymbol{\theta}}) = \operatorname*{Arg\,min}_{(\boldsymbol{c}, \boldsymbol{\theta})} J(\boldsymbol{c}_1, \boldsymbol{c}_2, \boldsymbol{\theta}_1, \boldsymbol{\theta}_2)$$

$$J(\boldsymbol{c}_1, \boldsymbol{c}_2, \boldsymbol{\theta}_1, \boldsymbol{\theta}_2) = \sum_{y_{ij} \in \boldsymbol{c}_1} -\ln p_1(y_{ij}; \boldsymbol{\theta}_1) + \sum_{y_{ij} \in \boldsymbol{c}_2} -\ln p_2(y_{ij}; \boldsymbol{\theta}_2)$$



(a) Frame 1 (d) Partition of Frame 1 (g) Bounding Boxes for Frame 1

Multimedia input string for frame 1: $G_1\&P_{10}\&P_{10}\&P_4\&P_4\&P_{19}$

(b) Frame 5 (e) Partition of Frame 5 (h) Bounding Boxes for Frame 5

Multimedia input string for frame 5: $G_1\&P_{10}\&P_{10}\&P_4\&B_4\&P_4\&P_{19}$

(c) Frame 8 (f) Partition of Frame 8 (i) Bounding Boxes for Frame 8

Multimedia input string for frame 8: $G_1\&P_{10}\&P_{10}\&P_{13}\&B_4\&P_4\&P_{19}$

Fig. 1. Key frames 1, 5 and 8 for a soccer game video shot and their corresponding multimedia input strings. (a)-(c) are the original frames 1, 5 and 8; (d)-(f) are the segmentation results for frames 1, 5 and 8; (g)-(i) show the segments with bounding boxes and centroids for frames 1, 5 and 8. For each frame, there is a corresponding multimedia input string for it.

The algorithm starts with an arbitrary partition of the data and computes the corresponding class parameters. Using these class parameters and the data, a new partition is estimated. Both the partition and the class parameters are iteratively refined until there is no further change in them. After the segmentation, the minimal bounding rectangle (MBR) concept in R-tree [16] is adopted so that each semantic object is bounded by a rectangle (as shown in Figures 1(d), 1(e), and 1(f)). Moreover, the centroid point of each semantic object is mapped to a point object for spatial reasoning.

2.2. *Using MATNs and Multimedia Input Strings to Model Video Key Frames*

An MATN can be represented diagrammatically by a labeled directed graph, called a transition graph. A multimedia input string is accepted by the grammar if there is a path of transitions which corresponds to the sequence of symbols in the string and which leads from a specified initial state to one of a set of specified final states.

An MATN can build up a video hierarchy [2]. A video clip can be divided into scenes, a scene contains a sequential collection of shots, and each shot contains some contiguous frames that are at the lowest level in the video hierarchy [1]. It is advantageous to use several key frames to represent a shot instead of showing all these frames. Key frames play as the indices for a shot. The key frame selection approach proposed in [2] was based on the number, temporal, and spatial changes of the semantic objects in the video frames. Other features may also be possible for the key frame selections, but we focus on the number, temporal, and spatial relations of semantic objects. Therefore, the spatio-temporal changes in each shot can be represented by these key frames. For example, in each shot of a soccer game, players may change positions in subsequent frames and the number of players appearing may change at the time duration of the shot.

Figure 1 is an example of how to use multimedia input strings to model video key frames. A soccer game video shot from frames 1 to 10 is used as the inputs of the SPCPE video segmentation method. Frames 1, 5 and 8 are chosen as the key frames by using the key frame selection method introduced in [2]. Figures 1(a)-(c) show the original frames, and Figures 1(d)-(f) give the resulting partitions (segments) for the frames. Since only the ball and the players are important from the content-based retrieval perspective, we use Figure 1(g)-(i) to simplify the segments for each frame. As introduced in [5], one semantic object is chosen as the target semantic object in each video frame and the minimal bounding rectangle (MBR) concept is also used. In order to distinguish the relative positions, twenty-seven numbers are used. In the following sections of this paper, we will describe how to use the proposed split algorithm to identify the overlapped players. For this example, each frame is divided into nine subregions.

Each key frame is represented by an input symbol in a multimedia input string and the "&" symbol between two semantic objects is used to denote that the se-

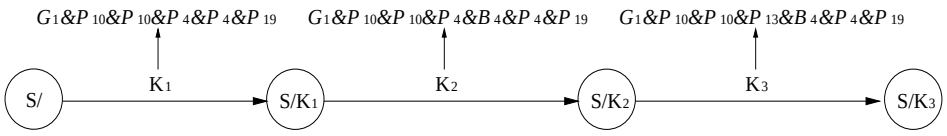


Fig. 2. Multimedia Augmented Transition Network and multimedia input strings for modeling the key frames of soccer game video shot S.

mantic objects appear in the same frame. The subscripted numbers are used to distinguish the relative positions of the semantic objects relative to the target semantic object "ground". So the multimedia input string to represent these three key frames is as follows:

$$\underbrace{(G_1\&P_{10}\&P_{10}\&P_4\&P_4\&P_{19})}_{K_1}\underbrace{(G_1\&P_{10}\&P_{10}\&P_4\&B_4\&P_4\&P_{19})}_{K_2}\underbrace{(G_1\&P_{10}\&P_{10}\&P_{13}\&B_4\&P_4\&P_{19})}_{K_3} \quad (1)$$

As shown above, there are three input symbols that are K_1, K_2 and K_3. The appearance sequence of the semantic objects in an input symbol is based on the spatial locations of the semantic objects in the video frame from left to right and top to bottom. For example, frame 1 is represented by input symbol K_1. G_1 indicates that G is the target semantic object. P_{10} means the first and the second P are on the left of G, P_4 means that the third and the fourth P are above G, and P_{19} means that the fifth P is on the right of G. For frame 5 (K_2), its multimedia input string is almost the same as that of frame 1 except that in which the soccer ball B appears above G. So the number of semantic objects increases from six to seven. This is an example to show how a multimedia input string can represent the changes of the number of semantic objects.

Figure 2 is the MATN for the key frames of the example soccer game video clip. The starting state name for this MATN is **S/**. As shown in Figure 2, there are three arcs with arc labels the same as the three input symbols in (1). For example, the arc symbol K_1 corresponds to the first input symbol of (1). The different state nodes in the MATN model the temporal relations of the selected key frames. The relative spatial relations of the semantic objects are modeled by the multimedia input strings.

3. Improving the Segmentation Method Using Backtrack-Chain-Updation Split Algorithm

As mentioned in Section 1, the current SPCPE algorithm cannot distinguish the overlapped segments within the video frames. Since a multimedia database query may involve information about moving objects and their locations in video sequences, it is very crucial to be able to distinguish the overlapped objects. The more accurate the segmentation algorithm is, the more efficient the key frame selection mechanism is (will be shown in Section 4). The ability to distinguish the

overlapped objects in the video sequences is beneficial for searching/retrieving multimedia information. Here a *backtrack-chain-updation split algorithm* that can find the split segment (object) and use the information in the current frame to update the previous frames in a backtrack-chain manner is proposed. This split algorithm is an enhancement of the SPCPE algorithm to be able to distinguish two separate objects that were overlapped previously. We applied the proposed algorithm to the air-show video sequence, people-walking video sequence, and soccer game video sequence. The experimental results show that the proposed algorithm can recover the separate situation successfully and automatically without any user intervention, which can well comply with the unsupervised feature of the SPCPE algorithm.

3.1. *The Proposed Backtrack-Chain-Updation Split Algorithm*

In this section, the backtrack-chain-updation split algorithm is proposed to enhance the functionality of the current SPCPE algorithm. The proposed backtrack-chain-updation split algorithm considers the situation when overlapping happens between two objects that separate from each other in a later frame. Assume that there are no major differences on the sizes and the shapes of those two objects, and the sizes and shapes of the same object do not change a lot in the consecutive frames.

3.1.1. *Object Tracking*

In order to detect the situation of overlapping, it is necessary to have the ability to track the objects (segments) within the successive video frames. Note that the partition of the previous frame is used as the initial condition in partitioning the current frame. Intuitively, two segments that are spatially the closest in the adjacent frames are connected. Euclidean distance is used here to measure the distance between their centroids. Let ctr_k and ctr_l be the centroids of segments k and l.

$$dist(ctr_k, ctr_l) = ||ctr_k - ctr_l||_2 \leq \delta$$

Besides using Euclidean distance, some size restriction is applied to the process of object tracking. If two segments in successive frames represent the same object, the difference between their sizes should not be large.

Let us think about what happens when overlapped segments separate from each other in the successive frame. When the split happens, some segment with overlapping in the previous frame may not find its corresponding part in the current frame since either the centroid or the size changes a lot. As a result, there may be some segments in the current frame that cannot be tracked back to the segments in the previous frame. These segments are called the *unidentified segments*. Then, we try to build up the relationship between those in previous frame and those in current frame based on the obtained information (i.e., the size and position information of those *unidentified segments*). In our algorithm, the concept of MINDIST in [17] is adopted.

Definition 1: A bounding box B (of dimension 2), will be defined by the two endpoints S and T of its major diagonal [17]:

$$\text{B} = (\text{S, T}), \text{ where}$$
$$\text{S} = [s_1, s_2] \text{ and } \text{T} = [t_1, t_2] \text{ and } s_i \leq t_i \text{ for } i = 1,2.$$

Definition 2: The distance of a point $\text{P} = [p_1, p_2]$ from a rectangle B in the same space, denoted *MINDIST(P, B)*, is defined as follows [17].

$$MINDIST(P,B) = \sum_{i=1}^{2} |p_i - r_i|^2, where$$

$$r_i = \begin{cases} s_i & if\ p_i < s_i \\ t_i & if\ p_i > t_i \\ p_i & otherwise \end{cases}$$

The *MINDIST* is a variation of the classic Euclidean distance that is applied to a point and a rectangle. When the point is inside the rectangle, the distance between them is zero. Whereas when the point is outside the rectangle, the square of the Euclidean distance between the point and the nearest edge of the rectangle is used. The square of the Euclidean distance is used since fewer and less costly computations are involved.

3.1.2. *Backtrack-Chain-Updation Split Algorithm*

Let ctr_k and BB_k be the centroid and bounding box of segment k in frame $i-1$, ctr_l and BB_l be the centroid and bounding box of segment l in current frame i. The backtrack-chain- updation split algorithm is given as follows. Figures 3(a)-(c) are used to illustrate the algorithm.

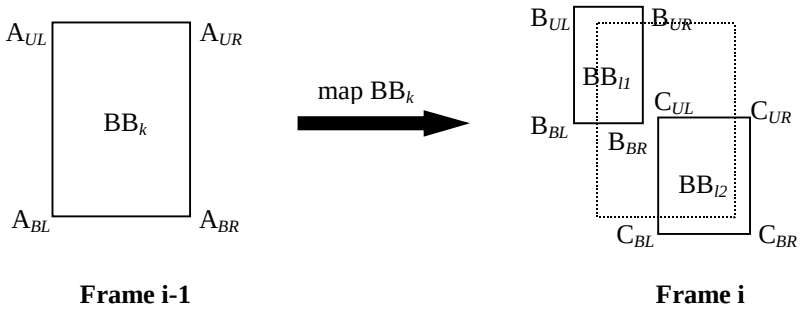
Step 1: *Identify as many as possible of the related segments.*

If $dist(ctr_k, ctr_l) = ||ctr_k - ctr_l||_2 \leq \delta$ **AND** $|size(BB_k) - size(BB_l)| \leq \beta$, where δ and β are threshold values for distances and sizes, then segment l in frame i is related to segment k in frame $i-1$. Mark segments l and k as "identified". Let segment k be the "parent"of segment l, and let segment l be the "child" of segment k.

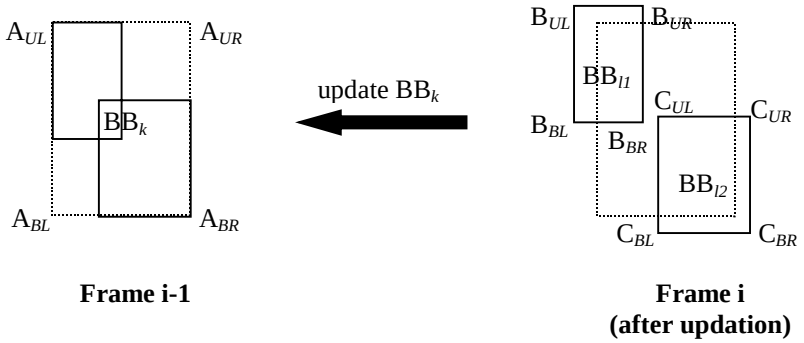
Step 2: *Find out the split segments in the current frame.*

Select one segment that is not identified in frame $i-1$. Let its centroid and bounding box be ctr_k and BB_k. Based on the size of BB_k, find out all the unidentified segments in frame i whose bounding box BB_l overlaps with BB_k and $\beta_{Min} < (size(BB_k)/size(BB_l)) < \beta_{Max}$. Note that we apply a size restriction to avoid the interference of some small segments ("noise"). Select the first two segments (say BB_{l1} and BB_{l2}) whose $MINDIST(ctr_k, BB_l1)$ and $MINDIST(ctr_k, BB_{l2})$ are

(a): Bounding boxes of segment *k* in frame *i* –1 and segments *l1*, *l2* in frame *i*.



(b): Find out the split segments in frame *i*.



(c): Find out the recovery vertices for BB_{l1} and BB_{l2} and update BB_k in frame *i-1*.

Fig. 3. The basic workflow of the backtrack-chain-updation algorithm

the smallest and the second smallest, and mark them as "identified".Then build up a parent-child relationship between BB_k and BB_{l1}, and between BB_k and BB_{l2}.

Figure 3(b) shows how to map BB_k into frame i and to find all the unidentified segments in frame i whose bounding boxes overlap with BB_k. BB_{l1} and BB_{l2} in frame i overlap with the boundary of BB_k; they are selected as the children segments of segment k in frame $i-1$. Now there is a parent segment with bounding box BB_k in frame $i-1$, and there are two children segments with bounding boxes BB_{l1} and BB_{l2} in frame i (as shown in Figure 3(a)). The vertices of each of the bounding boxes are given with the subscripts "UR", "UL", "BR", "BL" which represent the upper-right, upper-left, bottom-right, and bottom-left, respectively.

Step 3: *Do segmentation on the next frame and get the parameter for size adjustment in step 4.*

Once the split segments are identified, we can use the information we have obtained so far to update the parent segment k's bounding box in frame $i-1$. The main idea is to find the recovery vertex in the parent segment's bounding box, then "paste" the children's bounding boxes into the previous frame without changing their sizes and shapes. Remember, we assume that the sizes and shapes of the same object (segment) do not change much in the consecutive frames, but we do allow some small changes in the length or width of its bounding box. In such cases, sometimes the changes may exceed the updated bounding boxes, which results in unsatisfactory recovery results if no adjustment is applied.

Let the current frame be frame i, and the previous one be frame $i-1$. In this step, we do segmentation on frame $i+1$ and build up the parent-child relationship between frame i and frame $i+1$. For the split segments we just identified in frame i, their children segments may exist in frame $i+1$. If that is true, then the ratios of size changes on length and width for each split segment in frame i are calculated. For example, suppose the parent segment in frame i is segment l (BB_l), and the corresponding child segment in frame $i+1$ is segment p (BB_p). Then for segment l, its parameters are

$$Width_{Sensitivity_l} = \frac{|Width_{BB_p} - Width_{BB_l}|}{Width_{BB_l}}$$

$$Length_{Sensitivity_l} = \frac{|Length_{BB_p} - Length_{BB_l}|}{Length_{BB_l}}$$

If $Width_{Sensitivity_l} > Length_{Sensitivity_l}$, we say that segment l is more *width-sensitive*. This means the ratio of width changes is more than that of length changes, or the width changes of that segment/object may be more frequent and significant than that of length changes. Otherwise, it is said to be more *length-sensitive*. Another possibility is that we cannot find the corresponding child segment in frame $i+1$ due to object merging or disappearing. In this case,

$$Width_{Sensitivity_l} = \frac{Width_{BB_l}}{Length_{BB_l}}$$

$$Length_{Sensitivity_l} = \frac{Length_{BB_l}}{Width_{BB_l}}$$

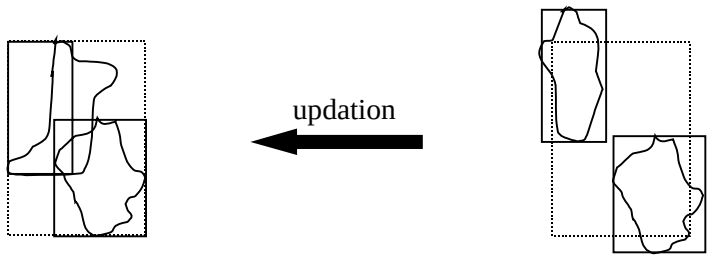
In step 4, we will show how to use this sensitivity parameter for size adjustment during the backtrack-chain-updating process.

Step 4: *Backtrack and update the previous frames plus size adjustment.*

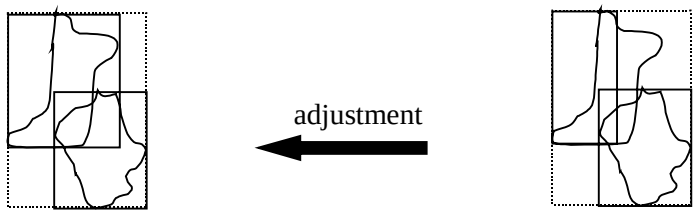
After we find the split segments (i.e., the children segments BB_{l1} and BB_{l2}) in frame i, we can use this information to update the previous frame $i-1$. The goal is to distinguish the separate bounding boxes on the parent segment k (BB_k) with overlapping. The first step is to check the $MINDIST$ from the four vertices of BB_k's to the children's bounding boxes respectively. For BB_{l1}, the vertex P on BB_k with the minimum $MINDIST(P, BB_{l1})$ is selected as the recovery vertex for BB_{l1}. According to this recovery vertex, there is a corresponding vertex in BB_{l1}. So the next step is to move the corresponding vertex to the recovery vertex, and to copy the bounding box BB_{l1} within the boundary of BB_k in frame $i-1$. The same procedures are applied to BB_{l2}.

As shown in Figure 3(b), BB_k is mapped to frame i and used to compute $MINDIST$. For example, for BB_{l1}, compute $MINDIST(A_{vex}, BB_{l1})$, where vex is one of "UR", "UL", "BR", and "BL" Choose the one with minimum $MINDIST$ as the recovery vertex for BB_{l1}. Here, A_{UL} is chosen as the *recovery vertex* for BB_{l1}, and B_{UL} is chosen as the corresponding vertex. Similarly, for BB_{l2}, A_{BR} and C_{BR} are selected as the recovery vertex and the *corresponding vertex*. To update the bounding box BB_k in frame $i-1$, the bounding box BB_{l1} is copied into BB_k with B_{UL} overlapping with A_{UL} and BB_{l2} is copied into BB_k with C_{BR} overlapping with A_{BR}. Notice that all the "copies" should be within the boundary of BB_k. By doing so, the updated version of frame i with separate bounding boxes can be obtained (as shown in Figure 3(c)).

In many cases, it seems satisfactory to just copy the children's bounding boxes to their parent's bounding box without any size adjustment. But there are also many situations that require necessary size adjustments to reduce the recovery error and achieve better results. For example, as shown in Figure 4(a), we can see separate bounding boxes in frame $i-1$, but the upper bounding box seems a little narrow due to the length change of that bounding box. This seems somewhat unsatisfactory for human eyes and for the purpose of video indexing. The parameters obtained in step 3 will be helpful for size adjustment. In this case, we can decide the upper segment in frame $i-1$ is *length-sensitive*, so what we do is to adjust the length of that bounding box to best fit its shape in frame $i-1$. Figure 4(b) shows the result after size adjustment. It looks fairly good.



(a) Update frame *i*-1 without size adjustment



(b) Bounding boxes of frame *i*-1 after size adjustment

Figure 4: Size adjustment after updation for frame $i-1$

This algorithm, which can be applied to update more previous frames by utilizing the information obtained so far, is called *backtrack-chain-updation*.

3.2. *Results of Applying Backtrack-Chain-Updation Split Algorithm on Video Segmentation*

The proposed *backtrack-chain-updation* split algorithm is applied to several example video sequences. In our experiments, three grayscale video sequences are used—the air- show video, people-walking video, and soccer game video. Table 1 lists the features of these three video sequences. Among them the air-show video is obtained from [18]. Parts of the frames in each video sequence are shown to demonstrate the test results of the proposed split algorithm.

Table 1. Three example video sequences.

Category	# of frames	# of rows/frame	# of Columns/frame
Air-show video	50	120	160
People-walking video	50	240	320
Soccer-game video	60	146	240

First, the test results for the air-show video sequence and people-walking video sequence are shown. In these two cases, correct results can be obtained by applying the SPCPE algorithm and the *backtrack-chain-updation* split algorithm even without size adjustment. Since the shapes of objects in these two videos are relatively stable through the sequences, the effect of size adjustments here is not significant. The split results without size adjustments are shown in Figures 5 and 6. The test results for the soccer game video sequence with size adjustments are presented in Figures 7 and 8 to show its effectiveness.

The air-show video frames are partitioned using the SPCPE algorithm with 2 classes. Frames 13 to 16 are shown in Figure 5 to demonstrate the effects of the split algorithm. Figure 5(a) shows the original Frames 13 to 16, while the original segmentation for Frames 13 to 16 are shown in Figure 5(b). Notice that Frame 16 consists of the split segments (objects), and Frames 13 to 15 are the previous frames that have the overlapped segment. First we use the information available in Frame 16 to adjust the corresponding objects in Frame 15. Then Frames 14 to 13 are updated one by one in a chain-updation manner. The centroid of each segment is marked with an "x" and the segment is shown with a bounding box around it. Figure 5(c) shows the segmentation results using our bakctrack-chain-updation split algorithm without size adjustment. We use Figure 5(d) to give a clear view on the semantic objects (Airplanes, in this case) and their relative positions in each frame. In addition, the corresponding multimedia input strings are listed below each frame.

As can be seen from Figure 5(b), the two airplanes are overlapped in Frame 15 but are separated in Frame 16. Under the original *SPCPE* algorithm, there is no way to distinguish these two airplanes in Frame 15 even though they are separated in Frame 16. However, after applying the proposed split algorithm on Frame 15, these two objects can be identified successfully and automatically (as shown in Frame 15 on Figure 5(c)). In addition, the same procedure has been applied to any previous frame that has the overlapped segment to identify the separate objects based on the information in the current frame in a backtrack-chain manner. In other words, the proposed split algorithm can distinguish two separate objects that were overlapped previously. Moreover, it needs to be noted that the multimedia input strings for the four frames remain the same. This allows the use of one key frame instead of two key frames (i.e., without separating the overlapped objects) to represent these four frames so that more accurate video indexing can be achieved.

Figures 6(a)-(d) give another test results for the people-walking example video sequence. As shown in Figure 6(a), two people walked towards each other and then walk away from each other. Frames 10 to 13 are chosen to represent an overlapping-¿split process. As can be seen from Figure 6(c), each overlapped segment in Frames 10 to 12 has been successfully identified as two separate objects. Note that the people-walking video sequence is more complex than the air-show video sequence because the people object consists of different types of heterogeneous regions such as the clothes, arms, hair and face. However, these heterogeneous regions can still be identified as one people object instead of multiple objects due to the effectiveness

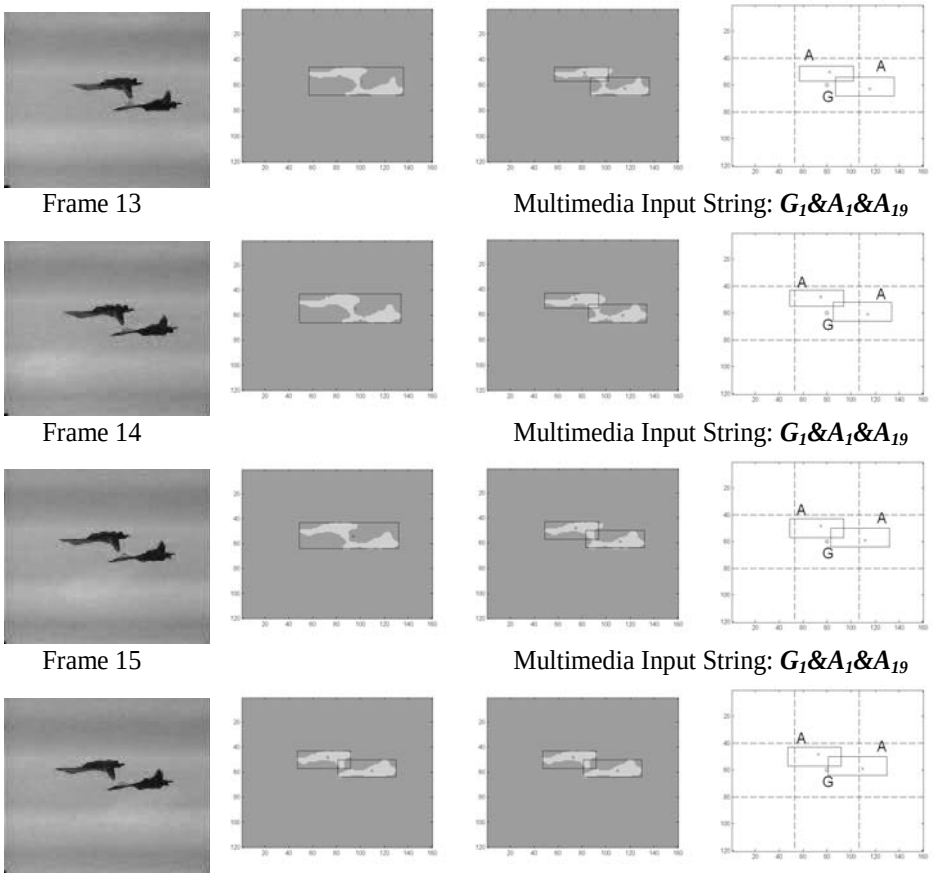


Fig. 5. Test results for the air-show video sequence using the backtrack-chain-updation split algorithm

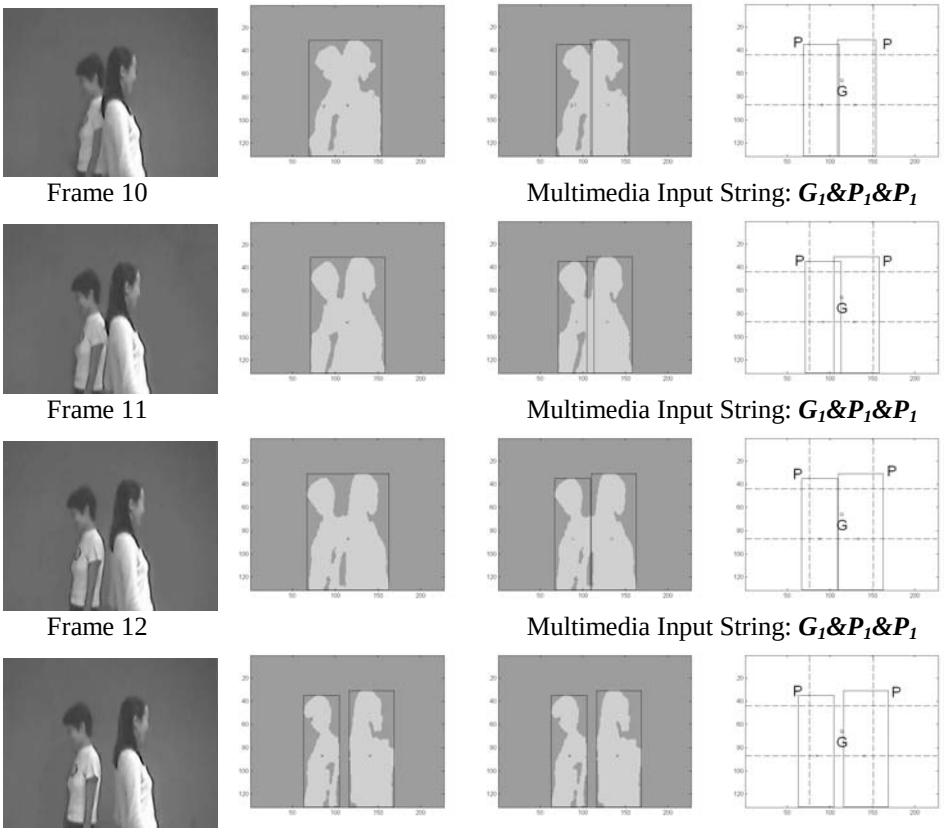


Fig. 6. Test results for the people-walking video sequence using the backtrack-chain- updation split algorithm

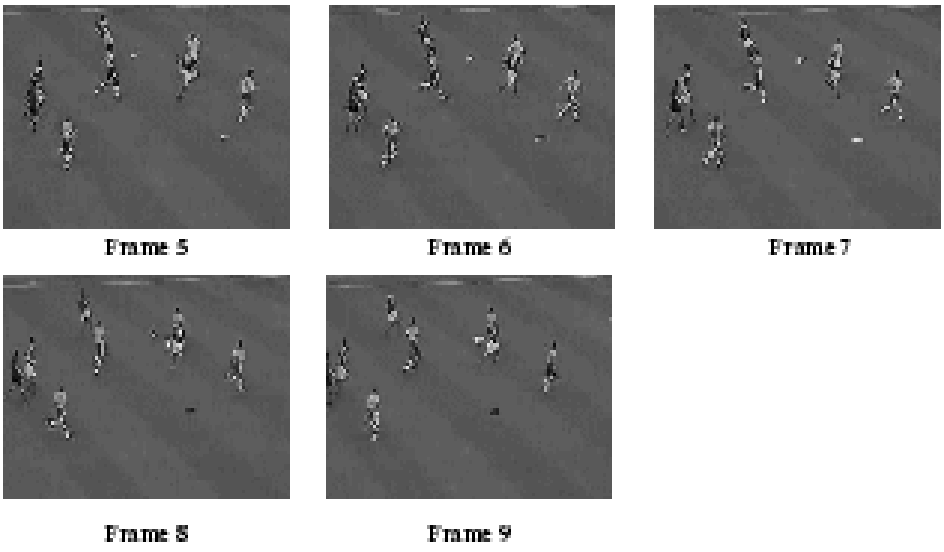
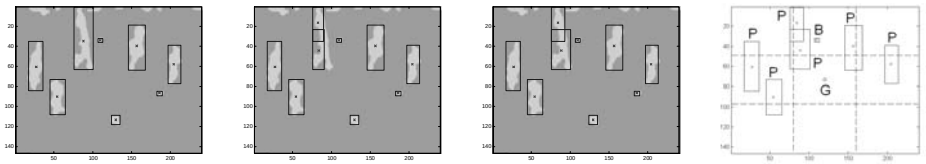


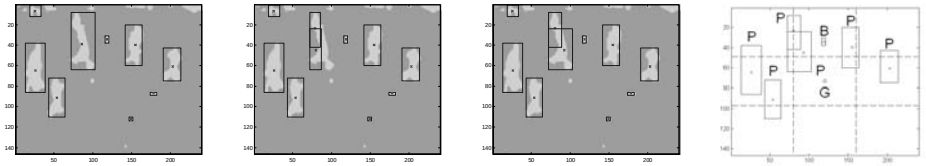
Fig. 7. Original soccer video Frames 5 to 9

of the *SPCPE* algorithm.

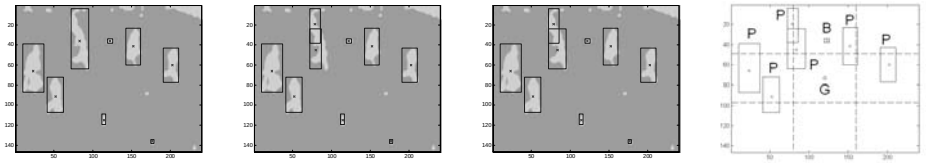
In the above two examples, the two overlapping objects (airplanes or people) are slow-moving objects, and their shapes and sizes do not change a lot between two consecutive frames through the video sequences. In this case, we can still get correct results by just copying the children's bounding boxes to their parent's bounding box without any size adjustment. However, even though our split algorithm can identify the overlapped objects successfully in many cases, the effects of recovery are not always satisfactory since the bounding boxes with fixed sizes may not fit the changing shapes of those segments (objects). Figure 7 shows the original soccer video frames 5 to 9, while the original segmentation for Frames 5 to 9 are shown in Figure 8(a). Figure 8(b) shows the segmentation results using backtrack-chain-updation split algorithm without size adjustments, while the segmentation results with size adjustments are given in Figure 8(c). Figure 8(d) gives the clear bounding box view for the testing result with size adjustment. As shown in Figure 8(b), the fixed size of bounding box makes it unable to fit the fast changing shape of object. Based on the split algorithm, the size adjustments can be applied to refine the segmentation results. As Figure 8(c) shows, the sizes of recovered bounding boxes are not fixed any more. In fact, the size adjustments let the sizes of bounding boxes dynamically fit the changing shapes of segments during the process of backtrack-chain-updation. As mentioned before, according to [2], the key frame selection focuses on the number, temporal, and spatial changes of semantic objects



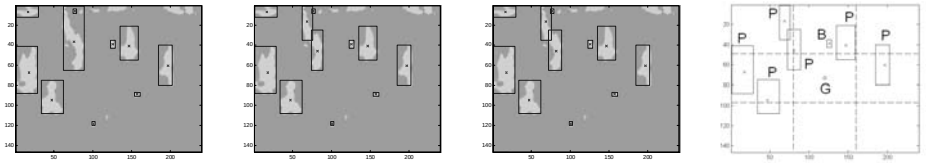
Frame 5 Multimedia Input String: $G_1\&P_{10}\&P_{10}\&P_{13}\&P_4\&B_4\&P_4\&P_{19}$



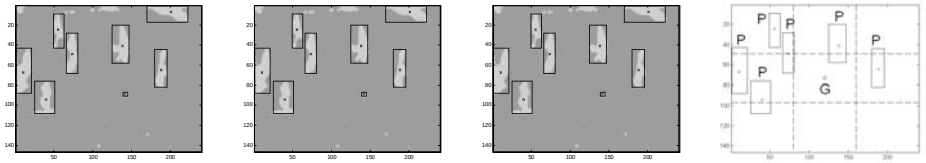
Frame 6 Multimedia Input String: $G_1\&P_{10}\&P_{10}\&P_4\&P_4\&B_4\&P_4\&P_{19}$



Frame 7 Multimedia Input String: $G_1\&P_{10}\&P_{10}\&P_{13}\&P_4\&B_4\&P_4\&P_{19}$



Frame 8 Multimedia Input String: $G_1\&P_{10}\&P_{10}\&P_{13}\&P_4\&B_4\&P_4\&P_{19}$



Frame 9 Multimedia Input String: $G_1\&P_{10}\&P_{10}\&P_{13}\&P_{10}\&P_4\&P_{19}$

(a) Original Segmentation | (b) Split Algorithm Without Adjustment | (c) Split Algorithm With Adjustment | (d) Bounding Boxes With Adjustment

Fig. 8. Applying the backtrack-chain-updation split algorithm and size adjustment on the soccer game video segmentation

in the video frames. The temporal and spatial relations of semantic objects are captured from the unsupervised video segmentation method, and furthermore, the unsupervised video segmentation algorithm has been greatly enhanced by using the proposed split algorithm; thus, we believe that our key frame selection approach will greatly benefit from this enhancement.

4. Conclusions and Future Work

In this paper, a backtrack-chain-updation split algorithm is presented. The proposed split algorithm can distinguish two separate objects that were overlapped previously in the video sequences to provide more accurate temporal and spatial relations of the semantic objects. The temporal and spatial relations of the semantic objects are captured and modeled by the unsupervised SPCPE video segmentation method, the MATN model, and the multimedia input strings. The proposed split algorithm is an enhancement of the SPCPE method. By obtaining more accurate temporal and spatial relations of the semantic objects from the proposed split algorithm, more accurate multimedia database queries can be answered. Three video sequences from different applications are used to demonstrate the effectiveness and accuracy of the proposed split algorithm. The experimental results show that the proposed split algorithm can recover the overlapped situation successfully and automatically without any user intervention. Currently, the proposed split algorithm works on splitting two overlapped semantic objects in the video sequences. In our future work, a more general way to solve the object overlapping problem as well as the merging situations so that the video indexing information obtained from segmentation can be more accurate and provide more semantic meaning.

Acknowledgments

For Shu-Ching Chen, this research was supported in part by NSF CDA-9711582.

References

[1] B-L Yeo and M.M. Yeung, *Retrieving and Visualization Video*, Comm. of the ACM. vol. **40**, No. **12** (1997) 43–52.

[2] S.-C. Chen, S. Sista, M.-L. Shyu, and R. L. Kashyap, *Augmented Transition Networks as Video Browsing Models for Multimedia Databases and Multimedia Information Systems*, the 11th IEEE International Conference on Tools with Artificial Intelligence (ICTAI'99), (Nov. 1999) 175–182.

[3] W. Woods, *Transition Network Grammars for Natural Language Analysis*, Comm. of the ACM. **13** (1970) 591–602.

[4] S.C. Kleene, *Representation of Events in Nerve Nets and Finite Automata, Automata Studies*, Princeton University Press, Princeton, N.J., (1956) 3–41.

[5] S.C. Chen and R. L. Kashyap, *A Spatio Temporal Semantic Model for Multimedia Presentations and Multimedia Database Systems*, IEEE Transactions on Knowledge and Data Engineering. vol. **13**, No. **4** (Jul./Aug. 2001) 607–622.

[6] J.D. Courtney, *Automatic Video Indexing via Object Motion Analysis*, Pattern Recognition. vol. **30**, No. **4** (1997) 607–625.

[7] A.M. Ferman, B. Gunsel, and A.M. Tekalp, *Object Based Indexing of MPEG-4 Compressed Video*, in Proc. SPIE: VCIP, San Jose, USA, vol. **3024** (Feb. 1997) 953–963.

[8] F. Arman, R. Depommer, A. Hsu, and M.Y. Chiu, *Contentbased Browsing of Video Sequences*, ACM Multimedia 94, (Aug. 1994) 97–103.

[9] Y. F. Day, S. Dagtas, M. Iino, A. Khokhar, and A. Ghafoor, *ObjectOriented Concept Modeling of Video Data*, IEEE Int'l Conference on Data Engineering, (Mar. 1995) 401–408.

[10] M. Flickner, H. Sawhney, W. Niblack, J. Ashley, Q. Huang, B. Dom, M. Gorkani, J. Hafner, D. Lee, D. Petkovic, D. Steele, and P. Yanker, *Query by Image and Video Content: The QBIC System*, IEEE Computer. vol. **28**, No. **9** (Sep. 1995) 23–31.

[11] E. Oomoto, and K. Tanaka, *OVID: Design and Implementation of a Video Object Database System*, IEEE Trans. on Knowledge and Data Engineering. vol. **5**, No. **4** (Aug. 1993) 629–643.

[12] S. Sista and R. L. Kashyap, *Unsupervised video segmentation and object tracking*, in IEEE Int'l Conf. on Image Processing, (Oct. 1999).

[13] G. Healey, *Using color for geometry-insensitive segmentation*, Journal of the Optical Society of America A. **6(6)** (1989) 920-937.

[14] J.-M. Jolion, P. Meer, and S. Bataouche, *Robust Clustering with Applications in Computer Vision*, IEEE Transactions on Pattern Analysis and Machine Intelligence. vol. **13** (Aug. 1991) 791–802.

[15] P. J. Besl and R. C. Jain, *Segmentation through variable-order surface fitting*, IEEE Transactions on Pattern Analysis and Machine Intelligence. vol. **10(2)** (Mar. 1988) 167–192.

[16] A. Guttman, *Rtree: A Dynamic Index Structure for Spatial Search*, in Proc. ACM SIGMOD, (Jun. 1984) 47–57.

[17] N. Roussopoulos, C. Faloutsos, and T. Sellis, *Rtree: Nearest Neighbor Queries*, Proc. ACM SIGMOD Intl. Conf. on Management of Data, (1995) 71–79.

[18] http://www.royfc.com/links/acft_video_flank.html, The Roy Cochrun Collection.