NeTra-V: Toward an Object-Based Video Representation

Yining Deng, *Student Member, IEEE,* and B. S. Manjunath, *Member, IEEE*

***Abstract*— We present here a prototype video analysis and retrieval system, called NeTra-V, that is being developed to build an object-based video representation for functionalities such as search and retrieval of video objects. A region-based content description scheme using low-level visual descriptors is proposed. In order to obtain regions for local feature extraction, a new spatio-temporal segmentation and region-tracking scheme is employed. The segmentation algorithm uses all three visual features: color, texture, and motion in the video data. A group processing scheme similar to the one in the MPEG-2 standard is used to ensure the robustness of the segmentation. The proposed approach can handle complex scenes with large motion. After segmentation, regions are tracked through the video sequence using extracted local features. The results of tracking are sequences of coherent regions, called "subobjects." Subobjects are the fundamental elements in our low-level content description scheme, which can be used to obtain meaningful physical objects in a high-level content description scheme. Experimental results illustrating segmentation and retrieval are provided.**

***Index Terms*— Content-based retrieval, object-based video retrieval, region tracking, spatio-temporal segmentation, video indexing.**

I. Introduction

With the rapid developments in multimedia and Internet applications, there is a growing need for new representations of video that allow not only compact storage of data, but also content-based functionalities such as search and manipulation of objects, semantic description of scenes, detection of unusual events, and possible recognition of objects. Current video standards, such as MPEG-2 and H.263 [10], are designed to achieve good data compression, but do not provide any content-related functionalities.

There has been much work done on the emerging MPEG-4 standard [21], which is targeted toward access and manipulation of objects as well as more efficient data compression. However, to date, applications are limited to cutting and pasting a few objects in simple scenes. There is no visual information extracted from the object itself that can be used for a similarity search or for recognition.

On the other hand, growing research in content-based video retrieval [1], [5], [9], [12], [14], [25], [26] has provided ways of searching video clips based on global similarities such as color, texture, and motion. However, much of the prior work in this area is restricted to global visual features. Very few [4], [24] have addressed practical issues of spatio-temporal segmentation for object-based video retrieval.

This paper describes some key aspects of a video analysis and retrieval system, called NeTra-V,[1] which is being developed with the objective of providing content-related functionalities. One of the main research focuses of NeTra-V is to build an object-based representation for video data, which will enable the search and retrieval of meaningful physical objects in a video database as well as other video content analysis. Toward this objective, a region-based content description scheme using low-level visual descriptors is proposed here. In order to obtain regions for local feature extraction, a new spatio-temporal segmentation and region tracking scheme is employed. Section II gives an overview of the system. Section III details the spatio-temporal segmentation scheme. Section IV illustrates the use of local features for region tracking. Section V describes the low-level video content description scheme and the corresponding feature descriptors. Section VI concludes with discussions.

II. System Overview

Fig. 1 shows the schematic diagram of the NeTra-V system. The research focus of this paper is on the shaded blocks, with the emphasis on the segmentation and tracking scheme essential to the low-level content description. Our work on temporal shot parsing and global feature extraction has been reported earlier in [5].

A video clip, whether in raw format or in one of the current compression standards, is a stream of binaries not well organized in terms of its content. As a first step toward better organization of data, the long video clip is parsed in the temporal domain into short video shots, each of which contains consistent visual content. Often, partitioning points of the video shots are natural camera breaks or editing cuts. A video shot can be considered as a basic unit of video data. Since visual information is similar in each shot, global image features such as color, texture, and motion can be extracted and used for the search and retrieval of similar video shots. This is what most current video retrieval systems do.

However, this approach provides very limited video information, and cannot answer simple questions such as, "What kind of *things* are in the video?" In order to further exploit the video content, a video shot needs to be decomposed into meaningful physical objects, so that search, retrieval, and content manipulation based on object characteristics, activi-

Manuscript received October 31, 1997; revised May 31, 1998. This work was supported by a grant from Samsung. This paper was recommended by Associate Editor M.-T. Sun.

The authors are with the Department of Electrical and Computer Engineering, University of California, Santa Barbara, CA 93106 USA.

Publisher Item Identifier S 1051-8215(98)06327-7.

[1] Netra means eye in Sanskrit, an ancient Indian language. NeTra is also the name of an image retrieval system described in [17].

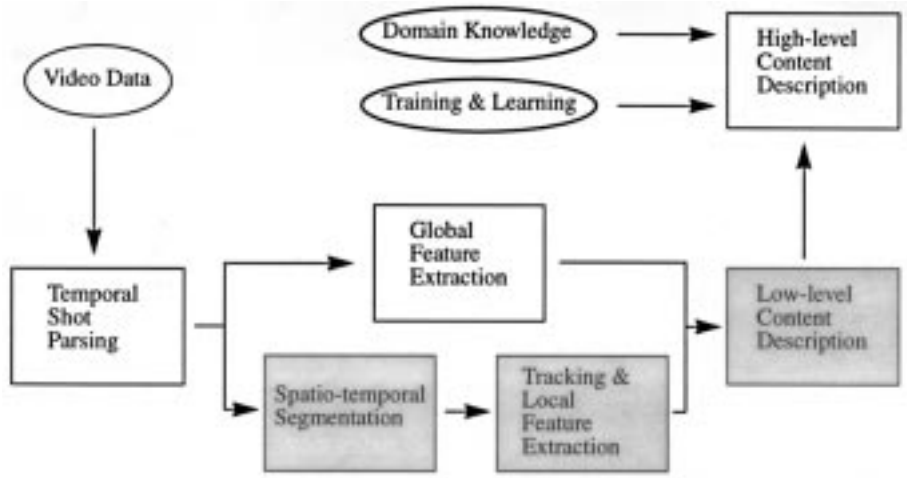

Fig. 1. Schematic diagram of the NeTra-V system.

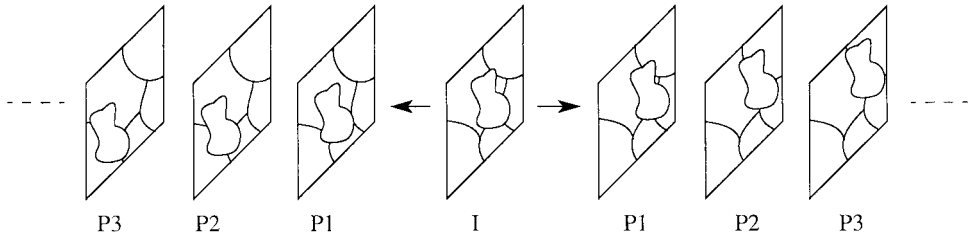

Fig. 2. General segmentation scheme. One group of frames is shown. I is the spatially segmented frame. $P1$, $P2$, and $P3$ are the first, second, and third predicted frames, respectively.

ties, and relationships are possible. The strategy adopted in NeTra-V is to first build a low-level video content description that can be completely automated. This low-level content description serves as the base for a high-level video content description which will incorporate the domain knowledge to "intelligently" organize the low-level description through training and learning, to achieve the goal of a true object-based representation. However, high-level content analysis is part of our ongoing research and is beyond the scope of this paper.

The following sections will discuss the spatio-temporal segmentation and tracking schemes that are essential to this low-level object-based content description. A description scheme and the feature descriptors for defining such a description are also given.

III. Spatio-Temporal Segmentation

A. General Scheme

Spatio-temporal segmentation has been a very challenging research problem, and many algorithms are proposed in the literature [6], [7], [13], [22], [23]. Many approaches use optical flow methods [11] to estimate motion vectors at the pixel level, and then cluster pixels into regions of coherent motion to obtain segmentation results. Using motion information for segmentation is a good idea that exploits the underlying nature of the video data, but there are two major drawbacks to this approach.

- The optical flow method does not cope well with large motion.
- Regions of coherent motion may contain multiple objects and need further segmentation for object extraction. For example, a region of blue sky and white clouds that undergoes the same camera panning will be segmented into one region.

To overcome these drawbacks, it is important to incorporate spatial information into motion segmentation. One feasible approach is to spatially segment the first frame to obtain initial segmentation results, and then motion segment subsequent frames using affine region matching. There are several advantages of doing this.

- Multiple objects with the same motion are separated by spatial segmentation. These objects can still be merged together for analysis if necessary after motion estimation.
- Affine region matching is a more reliable way of estimating motion than optical flow methods, and there are some fast numerical methods proposed to estimate the affine motion parameters [2], [3], [20].

Problems remain for this approach to work in practice. The new objects entering the scene and the propagation error due to affine region matching must be handled. We propose using a group processing scheme similar to the one employed in MPEG-2 to refresh the spatial segmentation and ensure the robustness of the algorithm. This scheme is illustrated in Fig. 2.

Video data are processed in consecutive groups of frames. These groups are nonoverlapping and independent of each other. The best value for the number of frames in a group depends on object motion activities in the video data. In our experiments with football video sequences, this number was set to 7. There is one I-frame in each group, which is spatially segmented first. Starting from the I-frame, the rest of the frames in the group are segmented consecutively by affine matching the segmented regions to their next frame. These motion segmented frames are called P-frames. Instead of the beginning frame, the middle frame of each group is set as the I-frame in our implementation. Motion prediction is toward both the forward and backward directions. This way, the length of motion prediction propagation is reduced by half and segmentation accuracy is improved.

This scheme does have a disadvantage in that objects disappearing just before the I-frame or appearing just after the I-frame are not handled in that group. The initial regions are determined at the time of I-frame segmentation. Regions can disappear because of being covered by other regions or moving out of the image boundary, but no new regions are labeled during the P-frame segmentation. The situations of region disappearing and appearing are handled by the adjacent groups provided that the objects stay in the scene for at least a group of frames. The time span for error is short—at most three frames or 1/10 s, and the false segmentation should not be severe under normal circumstances. An alternative way of implementing the scheme could be simultaneous forward and backward prediction like the one used for B-frames in MPEG-2, i.e., the I-frames are set at the two ends of each group, and the prediction with less error is chosen from the two directions. This approach should achieve better segmentation results, and can handle situations of region disappearing and appearing. However, the initial spatial segmentations at the two ends could differ significantly because of large object motion, thus increasing the complexity of region tracking. For this reason, this approach is not adopted in our current implementation.

B. Spatial Segmentation

The success of the spatio-temporal segmentation algorithm depends largely on a good initial spatial segmentation. We use an algorithm proposed in our previous work [16], which provides a general framework for color and texture image segmentation, and gives good segmentation results on a diverse collection of images. A brief description of the algorithm is given here. Unlike previous work which considers color and texture segmentation as two separate issues, this method integrates color and texture features together to compute the segmentation. First, the direction of change in color and texture is identified and integrated at each pixel location. Then a vector is constructed at each pixel pointing in the direction that a region boundary is most likely to occur. The vector field propagates to neighboring points with similar directions, and stops if two neighboring points have opposite flow directions, which indicates the presence of a boundary between the two pixels. After boundary detection, disjoint boundaries are connected to form closed contours. This is followed by region merging based on color and texture similarities as well as the boundary length between the two regions. The algorithm is designed for general image data, and requires very little parameter tuning from the user. The only parameter to be specified is the scale factor for localizing the boundaries. Two examples of image segmentation using this algorithm are given in Figs. 3 and 4 where the spatially segmented images are marked "I."

C. Motion Segmentation

The results of spatial segmentation can be used for affine region matching. A six-parameter two-dimensional (2-D) affine transformation is assumed for each region in the frame, and is estimated by finding the best match in the next frame. Consequently, segmentation results for the next frame is obtained. A Gaussian smoothing is performed on each frame before affine estimation.

1) Gradient-Based Affine Motion Estimation: Gradient-based affine motion estimation is performed on the luminance component of the video data only. Mathematically, the following functional f is to be minimized for each region R:

$$f(a) = \sum_{(x,y)\in R} g[I_1(x', y') - I_2(x, y)] \tag{1}$$

where a is a six-parameter affine motion vector which can be separated in the x and y directions, $a = [a_x \;\; a_y]^T$ and $a_x = [a_{x1} \;\; a_{x2} \;\; a_{x3}]^T$, $a_y = [a_{y1} \;\; a_{y2} \;\; a_{y3}]^T$, g is a robust error norm to reject outliers, defined as [13]

$$g(e) = e^2/(\sigma + e^2) \tag{2}$$

where σ is a scale parameter, I_1 and I_2 are the current frame and the next frame, respectively, x and y are pixel locations, $x' = x + dx$ and $y' = y + dy$, dx, and dy are displacement vectors, $dx = b^T a_x$ and $dy = b^T a_y$, where $b = [1 \; x \; y]^T$.

Ignoring high-order terms, a Taylor expansion of (1) gives

$$\begin{aligned} f(a) = {} & f(a_0) + \nabla f(a_0)\,(a - a_0) \\ & + \tfrac{1}{2}(a - a_0)^T \nabla^2 f(a_0)\,(a - a_0). \end{aligned} \tag{3}$$

Using a modified Newton's method [15] that ensures both descent and convergence, a can be iteratively solved by updating the following equation at the kth iteration:

$$a[k+1] = a[k] - c[k]\{\nabla^2 f(a[k])\}^{-1} \nabla f(a[k]) \tag{4}$$

where $c[k]$ is a search parameter selected to minimize f. For (1), ∇f and $\nabla^2 f$ are calculated as shown in (5) and (6) at the bottom of the next page. Note that the gradient components $\partial I_1/\partial x'$ and $\partial I_1/\partial y'$ can be precomputed before the start of iterations.

2) Affine Parameter Initialization: The method derived in Section III-C1 requires the cost function to be convex in affine space to guarantee convergence to the global minimum. We assume this to be true in the vicinity of the actual affine parameter values. Thus, a good initialization is needed before starting the iterations. In the case that affine parameters of the previous frame are known, we can make a first-order

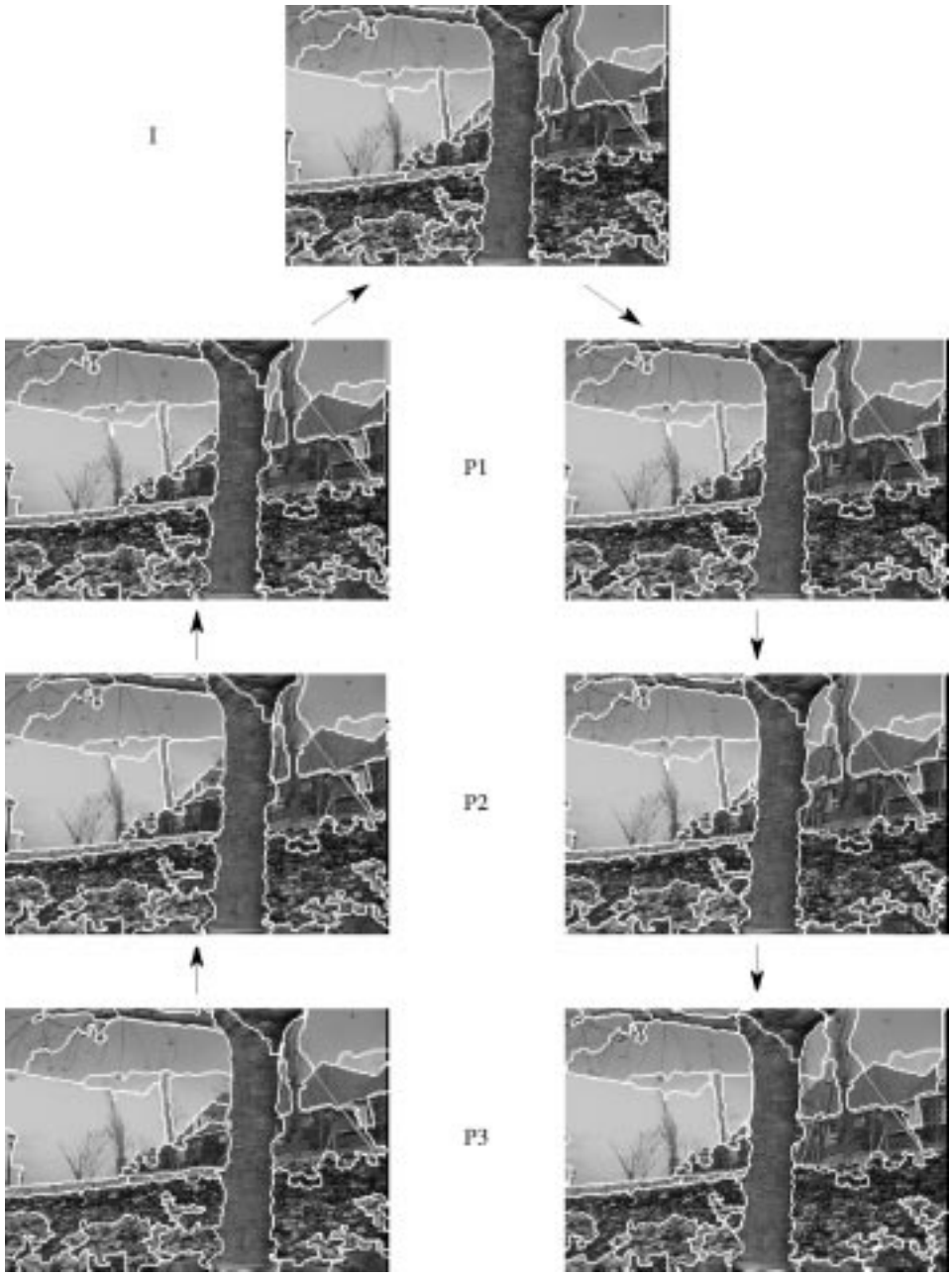

Fig. 3. Example of spatio-temporal segmentation of one group of frames in the "flower garden" sequence. Arrows indicate the actual flow of video frames in time. Image marked "I" is segmented spatially, while images marked "$P1$," "$P2$," and "$P3$" are segmented by motion prediction.

assumption that the region is going to keep the same motion, and use the affine parameters of the previous frame as the initial values for the current frame.

For the first frame to be segmented by motion prediction in each group, whose previous frame is an I-frame, a hierarchical search is performed to obtain the best initial affine parameters. This is only needed once for every group of frames, and can be performed at the beginning of either forward or backward direction. The initialization of the other direction is directly estimated by reversing the affine parameters. The search is performed using all three color components to ensure best results. To reduce the complexity, a four-parameter affine

$$\nabla f = \sum_{R} \left(\frac{\partial g}{\partial e} \left[\frac{\partial I_1}{\partial x'} b^T \; \frac{\partial I_1}{\partial y'} b^T \right]^T \right) \tag{5}$$

$$\nabla^2 f = \sum_{R} \begin{bmatrix} \dfrac{\partial^2 g}{\partial e^2} \left(\dfrac{\partial I_1}{\partial x'} \right)^2 + \dfrac{\partial g}{\partial e} \dfrac{\partial^2 I_1}{\partial x'^2} & \dfrac{\partial^2 g}{\partial e^2} \dfrac{\partial I_1}{\partial x'} \dfrac{\partial I_1}{\partial y'} \\ \dfrac{\partial^2 g}{\partial e^2} \dfrac{\partial I_1}{\partial x'} \dfrac{\partial I_1}{\partial y'} & \dfrac{\partial^2 g}{\partial e^2} \left(\dfrac{\partial I_1}{\partial y'} \right)^2 + \dfrac{\partial g}{\partial e} \dfrac{\partial^2 I_1}{\partial y'^2} \end{bmatrix} \otimes \begin{bmatrix} 1 & x & y \\ x & x^2 & yx \\ y & xy & y^2 \end{bmatrix}. \tag{6}$$

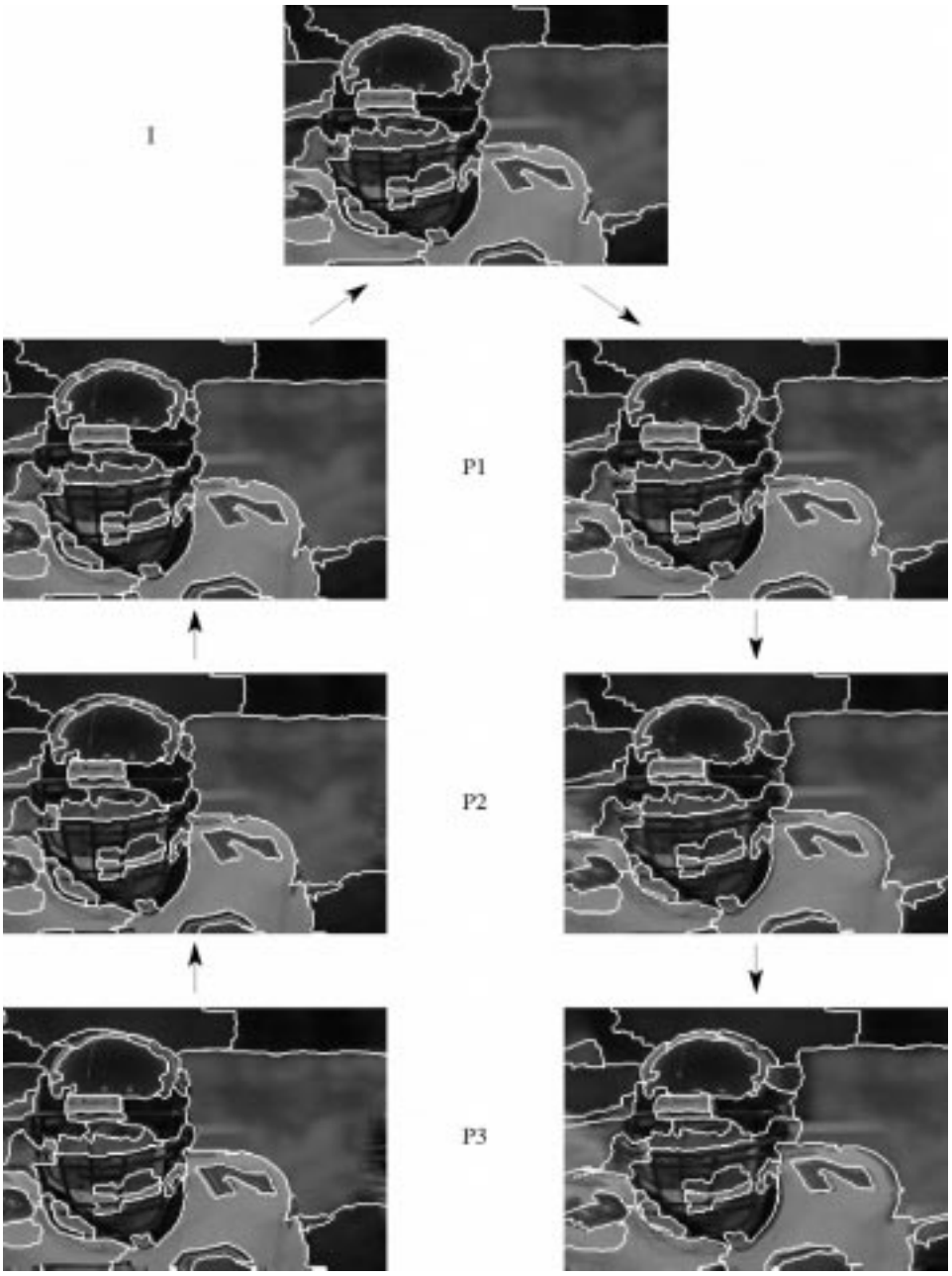

Fig. 4. Example of spatio-temporal segmentation of one group of frames in a football game video database. Arrows indicate the actual flow of video frames in time. Image marked "I" is segmented spatially, while images marked "$P1$," "$P2$," and "$P3$" are segmented by motion prediction.

TABLE I
SEARCH RANGE AND STEP OF AFFINE PARAMETERS

	x-translation (pixel)	y-translation (pixel)	scale	rotation (degree)
range	-22 to 22	-22 to 22	0.89 to 1.11	-9.5 to 9.5
step	1	1	0.01	0.5

model which accounts for x and y translations, scale, and rotation is used. The image is downsampled first, and results of the search at a lower resolution are projected back to the original image size for fine tuning. Table I gives the values of search range and search step for each parameter at the original image resolution. The values of search range are the effective values taken into account of the projection from the downsampled image.

3) Occluded and Uncovered Areas: Occluded and uncovered areas are difficult to handle in motion prediction. The problems are ill posed, and cannot be solved without assuming some prior knowledge. The following heuristics are used in

our method to avoid false segmentation due to these two situations.

In order to reduce the effects of occlusion, pixels along boundaries are not used for affine estimation. An occlusion occurs when there are pixels from different regions predicting to the same location in the next frame. There is an ambiguity in region labeling of that point in the next frame. The problem is solved if the layer information of the regions, i.e., which region is on top and which is below, is known. A layered representation of the two overlapping regions is computed as follows. First, errors between a predicted point and its predicting pixels from the two regions are calculated. The point is "possibly" coming from the region with a smaller error. Next, for all of the overlapping points, the number of the points "possibly" coming from each region is counted. The region with more "possible" points is considered on top of the other one, and all overlapping points are assigned to this region.

An uncovered area appears when two or more regions move apart. The problem is more difficult because there is little information known about the new area. It may contain multiple regions, and may cause the algorithm to fail if it is treated as one integrated new region. Intuitively, a point is uncovered when the original pixel moves away while there is no refilling from the motion of neighboring pixels. This indicates that the uncovered point might belong to the nearest region along the direction opposite of the original pixel motion. This is true when the boundary line between the two neighboring regions is perpendicular to the motion direction of the original region at the uncovered area. Since the uncovered area is usually a narrow long strip, this method works well most of the time. There are some extreme cases when this approach fails, but nevertheless, it gives a simple and logical estimation of the uncovered area.

D. Results of Spatio-Temporal Segmentation

Figs. 3 and 4 give two spatio-temporal segmentation examples using our proposed approach, one from an MPEG-2 standard test sequence "flower garden," the other from a football game sequence. The resolution of all our testing video sequences is 352×240. One group of frames is shown for each example. Arrows indicate the actual flow of video frames in time, while the order of segmentation process starts with the I frames and proceeds to $P1$, $P2$, and $P3$ frames. It can be seen from the examples that the segmentation of the tree and the sky in the "flower garden" sequence, and the helmet and the jersey in the football sequence is quite clear.[2]

Although these initial results shown here seem to be promising, they are far from being perfect. Further improvements are needed, and segmentation continues to be an important research issue. To summarize, we have presented a practical solution to the spatial–temporal segmentation problem, which uses all three visual features: color, texture, and motion in the video data, to segment the video data. To ensure the robustness of the algorithm, a group processing scheme similar to the one employed in the MPEG-2 standard is used. Each group of video frames has one I frame that is segmented by color and texture information and several P-frames that are segmented by motion information. Note that the segmentation technique is designed quite generally, and can be adapted for object-based coding with modifications [19].

[2]The original color images can be found at http://copland.ece.ucsb.edu/Demo/video/.

IV. Region Tracking and Local Features

A. General Scheme

The processes of tracking and local feature extraction are closely related in the NeTra-V system. One advantage of using the proposed segmentation scheme is that the problem of region tracking is simplified. There are two types of region tracking: intragroup region tracking and intergroup region tracking. Within each group, region tracking is already done during the segmentation. This accounts for most of the frames in the video and reduces the complexity of tracking. The remaining problem is to track regions across two adjacent groups. Because the regions could differ significantly in two groups due to large object motion, it is impossible to match all of the regions between the two groups. The constraint for tracking has to be relaxed. We do not attempt to match every region in the current group to the regions in the next group.

Naturally, intergroup region tracking would be done on the boundary frames between the two groups. In our scheme, it is performed on the I-frames of the two groups because of the following.

- I-frame segmentation is more accurate than P-frame segmentation.
- There are no new regions introduced in P-frames. Since the tracking requirement is loose and the length of a group is not too long, it is adequate to use the I-frames.
- In our video content description discussed later, each group of frames is represented by its I-frame, and local features are only extracted from the segmented regions in the I-frame. Therefore, it is convenient to use I-frames for intergroup region tracking.

The tracking between two consecutive I-frames is done by comparing the similarity between regions in the two frames using some of the local features extracted from each region. Fig. 5 illustrates the intergroup region-tracking process. In this drawing, regions labeled A in six consecutive I-frames are tracked. Notice the occlusion effect that the tracking algorithm must handle.

B. Region Tracking Using Local Features

The local features used for measuring region similarity include color, texture, size, and location. Motion features are used to predict region locations. Integrating information from different features is an important issue. A weighted sum of individual feature distances is often used in the literature, but it does not have much physical meaning, and the weights are usually assigned arbitrarily. In our approach to the region-matching problem, we observe that color is the most dominant visual feature, and we use it to rank the distance measure. Other features are only used as constraints to eliminate false matches, and this is achieved by the use of thresholds. In other

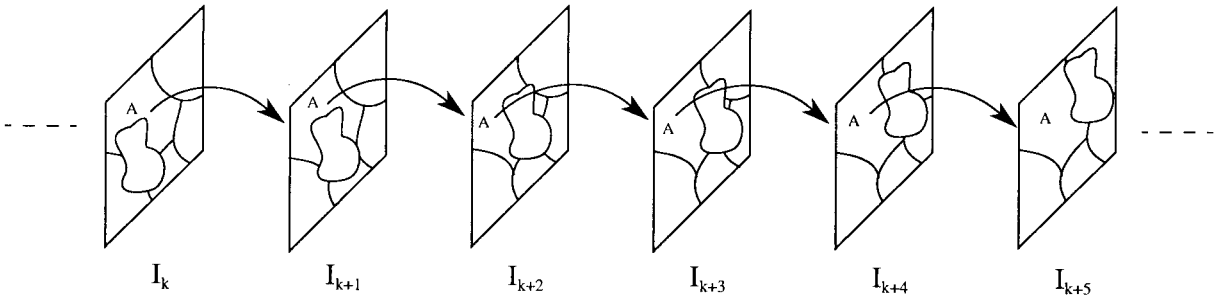

Fig. 5. Illustration of intergroup region tracking. Regions labeled A in six consecutive I-frames are tracked in the drawing. Notice the occlusion effect that the tracking algorithm must take care of.

words, the goal is to find the most similar region in color that is also close in texture, size, and location. The optimal values of the thresholds are determined experimentally and remain unchanged for the same type of data, for example, all of the football game databases.

Details on each feature and its distance measure are described in Section V. For a region R in the current I-frame, the steps for finding the matching region in the next I-frame are as follows.

1) For every region in the next I frame, the size and texture differences with R are compared with θ_S and θ_T—the preset thresholds for differences in size and texture, respectively. If both values are less than the thresholds, the region is considered as a candidate for color rank ordering.
2) Affine motion compensation is estimated to predict the approximate location of R in the next I-frame. The distance between the location of a candidate region and the predicted location of R is calculated and is denoted by d_L.d_L has to be less than θ_L, an image-size-dependent threshold, for the region to remain in candidacy.
3) The color feature distance between a candidate region and R is weighted by $1 + W_L d_L$, where W_L is a constant. That is, the further away a candidate region, the less likely that it will be the true match. The particular weighting by location difference is needed because there might be some objects nearby with similar color, texture, and size. The weighted color distance d_C is checked against a threshold θ_C to validate candidacy.
4) The candidate regions are ranked in terms of d_C. The region having the minimum value is determined to be the match for R.

If a match satisfying all of the conditions cannot be found, R is determined to have no match in the next I-frame. If there is a sequence of regions tracked up to R, the tracking ends at the current frame. Possible scenarios for this kind of situation could be that the object is getting occluded, moving out of the image boundary, or turning away from the camera so that the surface is no longer seen. On the other hand, R can be the start of a new sequence of matched regions if there is no previous tracking that goes up to R.

If two regions in the current I-frame have the same most similar region in the next I-frame, the more similar one is chosen for matching. For the other region, its second most similar region in the next I-frame is chosen for consideration, and the same process continues until either the match is found or no matching region is determined.

C. Results of Region Tracking

Fig. 6 shows two examples of intergroup tracking. A set of six consecutive I-frames is shown in each example, and a bounding box covering the tracked region is drawn on each image. The thresholds are set to be the same for all of the testing sequences: $\theta_S = 0.5$, $\theta_T = 1.0$, $\theta_L = 60$, $W_L = 0.005$, and $\theta_C = 0.25$. Fig. 6(a) is a half zoom-out view of a football field. The tracked region is a moving upper body of a football player. Notice the movements of other similar objects nearby that makes tracking more difficult. Fig. 6(b) shows the tracking of a person's face. Notice that in the first frame, the face is partially occluded, while in the last frame, there is a segmentation failure which merges the face with the helmet. In both cases, the tracking algorithm is robust enough to find the region.

The tracking technique presented here still needs to be improved. One consideration is the multiple-to-one and one-to-multiple region matching that can match one region to several connected regions instead of the one-to-one method used here. This will be especially useful when oversegmentation or undersegmentation occurs. Other issues include global motion compensation to estimate true object motion and the use of shape features for the matching process.

V. Low-Level Video Content Description

A. Description Scheme and Feature Descriptors[3]

With regions tracked and local features extracted, it is now possible to define a low-level object-based description scheme for video data. Before we proceed, two new items need to be defined.

- ***I*-region:** a region in an I-frame is called an I region.
- **Subobject:** a subobject is a sequence of tracked I-regions. The reason for using the name "subobject" is that object definitions are often quite subjective, and a spatially segmented I-region is usually a part of a

[3]The terminology of descriptors and the description scheme used here is similar to the ones used in the MPEG-7 requirements document V.5.

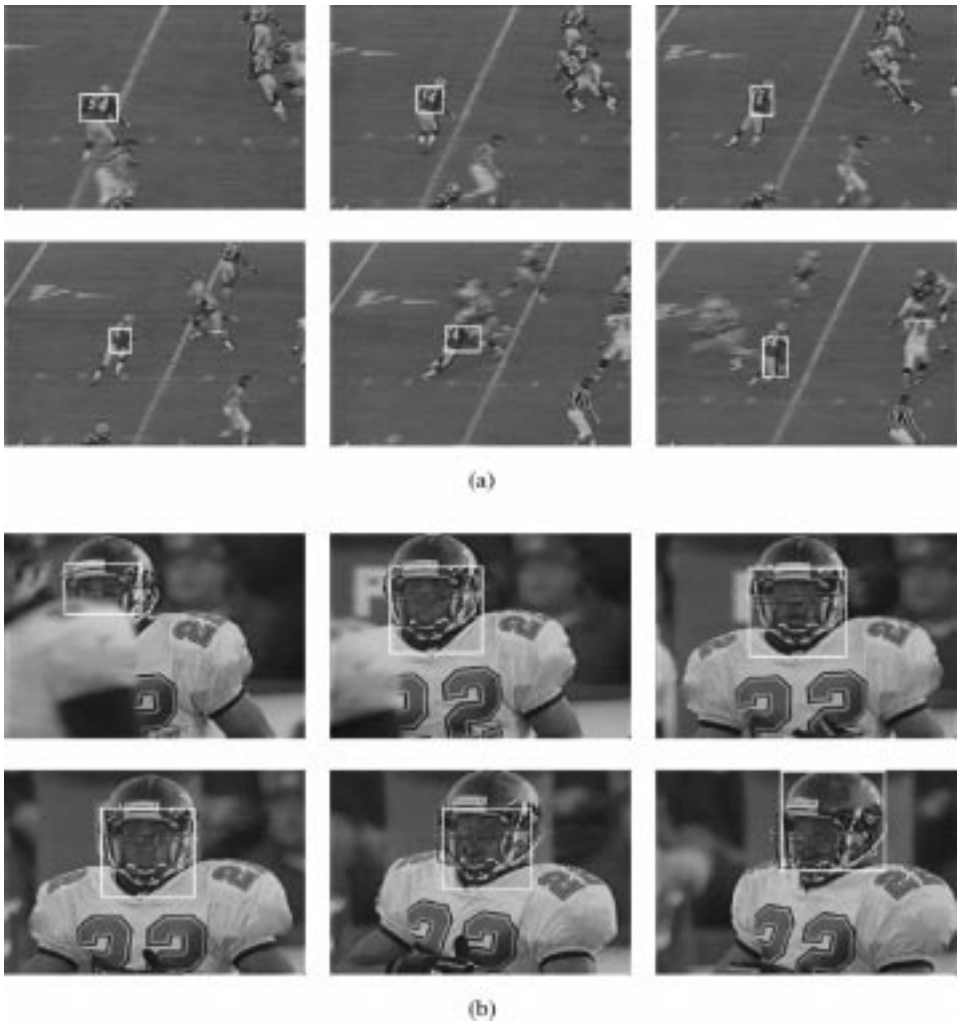

Fig. 6. Two examples of intergroup tracking in a football game video database, each showing a set of six consecutive I-frames, which covers 42 frames or 1.4 s of video.

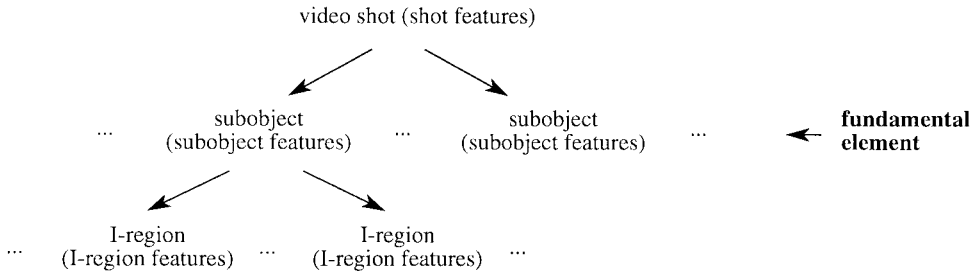

Fig. 7. Structure of low-level content description scheme.

meaningful large physical object. In the actual video, a subobject describes the visual characteristics of that part of an object. In our experiments, the duration of a subobject is required to be at least 3 I-frames or 0.5 s long to ensure that the subobject is indeed a part of an object that stays significantly long in the scene.

The structure of the description scheme is given in Fig. 7.

- Each video shot is decomposed into a set of subobjects. The subobjects are obtained by tracking.
- Each subobject consists of a sequence of tracked I-regions. The I-regions are obtained by segmentation.

It can be seen from the structure that each group of video frames is represented by its I-frame. Since the length of the group is not long and feature information does not change much within each group, this approach is valid, and reduces the complexity. Within each group, the segmentation provides subobject access on a frame-by-frame basis, which is needed for object manipulation.

A subobject is a fundamental element of this description scheme. Similarity search and retrieval are mainly performed using the subobjects. I-region information can also be used if necessary. For example, in order to answer a query such

TABLE II
FEATURE DESCRIPTORS AT EACH LEVEL

Features	I-region	Subobject	Video Shot
label	region label in the subobject region label in the frame	subobject label in the shot frame numbers labels of all I-regions	labels of all subobjects
color	region color histogram	mean of I-region features	global color histogram
texture	region Gabor texture feature	mean of I-region features	global Gabor texture feature
motion	affine motion parameters	mean and variance of I-region features	global motion histogram
shape	Fourier-based descriptor using curvature, centroid distance, and complex coordinate functions	mean of I-region features	
size	number of pixels	mean of I-region features	
location	centroid and bounding box	mean	

as "find the subobjects that move from left to right," motion and location information of each I-region of the subobject is needed. Extracted subobjects are intermediate results that will be used to obtain meaningful physical objects in a high-level content description scheme.

The three elements in the description scheme, shot, subobject, and I-region, are described by their corresponding feature descriptors, summarized in Table II. Shot-level features are global visual features. Subobject-level features are derived from the statistics of the I-region features. Brief explanations of the feature descriptors are given below.

Label: This is for access purpose. For example, if a specific I-region is needed, the label of the subobject that it belongs to is first obtained from the shot label. From the subobject label, the I-region label and the frame number are found. Finally, from the I-region label, the actual region in the frame is obtained.

Color: A 256-dimensional color histogram is used. The 256 color clusters are trained from the test database using a generalized Lloyd algorithm. The color distance is defined quadratically using the method described in [8].

Texture: Each pixel in the image is filtered by a Gabor filterbank of five scales and six orientations. A 60-dimensional Gabor texture feature vector is obtained by the mean and the variance of these filtered coefficients. A Euclidean distance measure is used. Details of Gabor texture features can be found in [18].

Motion: Six-dimensional affine motion parameters are used for I-region motion features. Because a Euclidean distance for the affine parameters does not have much physical meaning, an appropriate similarity measure for this type of motion feature is still under investigation. However, the I-region affine motion feature can be used indirectly to predict region locations as mentioned in the tracking process. At shot level, a 256-dimensional global motion histogram is computed. The motion histogram describes statistical information of the motion vectors, and details can be found in [5].

Shape: The contour of a region is represented by curvature, centroid distance, and complex coordinate functions. The contour length is normalized to 64. For each representation, the amplitude part of its FFT is used as shape feature to reduce noise effects and preserve rotational invariance. The feature vectors of all three representations are lexically arranged to obtain the final feature vector. Euclidean distance is used for similarity measure. Details can be found in [17].

Size: The size of each region is its number of pixels normalized by the video frame size. For two regions with size S_A and S_B, the difference in size is expressed in a ratio form, $|S_A - S_B|/\max(S_A, S_B)$.

Location: Location of a region is described by its centroid. The difference in locations is roughly estimated by the Euclidean distance between the two centroids.

In our current subobject search and retrieval experiments, the method of integrating different subobject features is similar to the one used in region matching. There is one dominant feature for rank ordering, and other features are only used as constraints. However, many problems remain for further investigation: Which features will best characterize the subobjects for some common application needs? Which feature among them is to be the dominant one? At this stage, the user has to select the appropriate features for the query.

B. Search Results and Discussions

Fig. 8 shows an example of retrieving subobjects with similar color (dominant) and texture. The query is a football player's face with a mask. The search performed on a group of 46 similar video shots including the query shot. These shots all contain zoom-in views of football players from two teams of different uniforms. The shots range from 60 to 300 frames long and contain a total of 2500–3000 subobjects. The top six matches are shown, and they all contain faces with a mask. The first retrieval is, in fact, the reappearance of the player's face in the query video shot after he turns his face sideways. The third and fourth retrievals also belong to one video shot,

Fig. 8. Example of retrieving subobjects with similar color and texture.

Fig. 9. Example of retrieving scenes where there is a subject moving across the screen in 2 s. A box is drawn on the images in the third column to highlight the object.

but are separated into two subobjects due to a segmentation failure.

By using the subobject information in the video data, the system is also able to answer more complicated questions, such as "show all the scenes where there is an object moving across the screen in 2 s." The corresponding query can no longer be described by a simple example, but could be expressed using a query language, for example, as shown in Fig. 9. Some retrieval results are also given in the figure. In each of the retrievals, there is a human object moving across the screen. Some of the movements are actual object motion, while some are due to camera panning.

It can be seen from the examples that the proposed low-level content description provides a way of finding information in the video without any high-level understanding of the actual content. How to organize this information into a meaningful representation is an interesting and challenging problem. Most of the time, users are not satisfied with meaningless retrieval results that are merely similar in visual appearance. For example, in Fig. 10, the system successfully identifies three subobjects: a flying football, a diving player's head, and his upper body in the same video sequence. While each individual subobject contains one separate piece of information in the video, the integration of all of the information to classify such a

Fig. 10. Three identified subobjects in the same video sequence. The same consecutive I-frames are shown in each row. The three subobjects are, from top to bottom, a football, a player's head, and his upper body, respectively.

scene as an action of catching a football would provide a more meaningful description of the video content. However, this is hard to achieve automatically using low-level features. Incorporating domain knowledge into these features through training and learning is necessary to go beyond the current stage and achieve the goal of a true object-based representation.

VI. Conclusions

In this paper, we have described some key aspects of the NeTra-V system, whose main objective is to build an object-based video representation. A low-level content description scheme is proposed, and the components necessary to achieve such a description are implemented, including a new automatic spatio-temporal segmentation and region-tracking scheme. Some experimental results are provided to demonstrate the feasibility of our approach. Quantitative performance evaluation measures are being investigated. For future research, our main focus is on incorporating domain-specific knowledge to achieve a high-level content description.

Acknowledgment

The authors would like to thank the reviewers for their valuable comments.

References

[1] E. Ardizzone and M. L. Cascia, "Multifeature image and video content-based storage and retrieval," *Proc. SPIE*, vol. 2916, pp. 265–276, 1996.
[2] J. Bergen, P. Burt, R. Hingorani, and S. Peleg, "A three-frame algorithm for estimating two-component image motion," *IEEE Trans. Pattern Anal. Machine Intell.* vol. 14, pp. 886–896, Sept. 1992.
[3] M. Bober and J. Kittler, "Robust motion analysis," in *Proc. IEEE Conf. Comput. Vision Pattern Recognition*, 1994, pp. 947–952.
[4] J. D. Courtney, "Automatic video indexing via object motion analysis," *Pattern Recognition*, vol. 30, no. 4, pp. 607–625, 1997.
[5] Y. Deng and B. S. Manjunath, "Content-based search of video using color, texture and motion," in *Proc. IEEE Int. Conf. Image Processing*, 1997, vol. 2, pp. 534–537.
[6] B. Duc, P. Schroeter, and J. Bigun, "Spatio-temporal robust motion estimation and segmentation," in *Proc. 6th Int. Conf. Comput. Anal. Images and Patterns*, 1995, pp. 238–245.
[7] F. Dufaux, F. Moscheni, and A. Lippman, "Spatio-temporal segmentation based on motion and static segmentation," in *Proc. IEEE Int. Conf. Image Processing*, vol. 1, 1995, pp. 306–309.
[8] J. Hafner *et al.*, "Efficient color histogram indexing for quadratic form distance functions," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 17, no. 7, 1995.
[9] A. Hampapur *et al.*, "Virage video engine," *Proc. SPIE*, vol. 3022, pp. 188–200, 1997.
[10] B. G. Haskell, A. Puri, and A. N. Netravali, *Digital Video: An Introduction to MPEG-2.* London, U.K.: Chapman & Hall, 1997.
[11] B. Horn and B. Schunck, "Determining optical flow," *Artif. Intell.*, vol. 17, pp. 185–203, 1981.
[12] G. Iyengar and A. B. Lippman, "Videobook: An experiment in characterization of video," in *Proc. IEEE Int. Conf. Image Processing*, vol. 3, 1996, pp. 855–858.
[13] S. Ju, M. Black, and A. Jepson, "Skin and bones: Multi-layer, locally affine, optical flow and regularization with transparency," in *Proc. IEEE Conf. Comput. Vision Pattern Recognition*, 1996, pp. 307–314.
[14] V. Kobla, D. Doermann, and K. Lin, "Archiving, indexing, and retrieval of video in the compressed domain," *Proc. SPIE*, vol. 2916, pp. 78–89, 1996.
[15] D. Luenberger, *Linear and Nonlinear Programming*, 2nd ed. Reading, MA: Addison-Wesley, 1984.
[16] W. Y. Ma and B. S. Manjunath, "Edge flow: A framework of boundary detection and image segmentation," in *Proc. IEEE Conf. Comput. Vision Pattern Recognition*, 1997, pp. 744–749. Also, *IEEE Trans. Image Processing*, submitted for publication, 1998.
[17] ——, "NeTra: A toolbox for navigating large image databases," in *Proc. IEEE Int. Conf. Image Processing*, vol. 1, 1997, pp. 568–571. Also, *ACM Multimedia Syst. J.*, to be published.
[18] B. S. Manjunath and W. Y. Ma, "Texture features for browsing and retrieval of image data," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 18, pp. 837–842, Aug. 1996.
[19] D. Mukherjee, Y. Deng, and S. K. Mitra, "A region-based video coder using edge flow segmentation and hierarchical affine region matching," in *Proc. SPIE, Visual Commun. Image Processing*, vol. 3309, 1998.
[20] H. Sanson, "Toward a robust parametric identification of motion on regions of arbitrary shape by nonlinear optimization," in *Proc. IEEE Int. Conf. Image Processing*, vol. 1, 1995, pp. 203–206.

[21] *IEEE Trans. Circuits Syst. Video Technol.* (*Special Issue on MPEG-4*), vol. 7, no. 1, 1997.
[22] J. Wang and E. Adelson, "Spatio-temporal segmentation of video data," *Proc. SPIE*, vol. 2182, pp. 120–131, 1994.
[23] L. Wu, J. Benois-Pineau, and D. Barba, "Spatio-temporal segmentation of image sequences for object-oriented low bit-rate image coding," in *Proc. IEEE Int. Conf. Image Processing*, vol. 2, 1995, pp. 406–409.
[24] D. Zhong and S. F. Chang, "Video object model and segmentation for content-based video indexing," in *Proc. IEEE Int. Symp. Circuits Syst.*, 1997.
[25] D. Zhong, H. J. Zhang, and S.-F. Chang, "Clustering methods for video browsing and annotation," *Proc. SPIE*, vol. 2670, pp. 239–240, 1996.
[26] H. J. Zhang, J. Wu, D. Zhong, and S. W. Smolliar, "An integrated system for content-based video retrieval and browsing," *Pattern Recognition*, vol. 30, no. 4, pp. 643–658, 1997.

Yining Deng (S'95) received the B.E. degree in electrical engineering from the Cooper Union for the Advancement of Science and Art, New York, NY, in 1995, and the M.S. degree in electrical engineering from the University of California at Santa Barbara in 1996. During the summer of 1994, he interned at the Polytechnic University, New York, NY. He is currently a Ph.D. student in the Department of Electrical and Computer Engineering, University of California at Santa Barbara. His research interests include image and video analysis, multimedia databases.

Mr. Deng is a member of Eta Kappa Nu and Tau Beta Pi.

B. S. Manjunath (S'88–M'91) received the B.E. degree in electronics (with distinction) from the Bangalore University in 1985, the M.E. degree (with distinction) in systems science and automation from the Indian Institute of Science in 1987, and the Ph.D. degree in electrical engineering from the University of Southern California (UCSB) in 1991.

He joined the Electrical and Computer Engineering Department at UCSB in 1991 where he is now an Associate Professor. During the summer of 1990, he worked at the IBM T.J. Watson Research Center, Yorktown Heights, NY. His current research interests include computer vision, learning algorithms, image/video databases, and digital libraries.

Dr. Manjunath was a recipient of the national merit scholarship (1978–1985) and was awarded the university gold medal for the best graduating student in electronics engineering in 1985 from the Bangalore University. He has served on the program committees of many international conferences and workshops and was on the organizing committee of the 1997 International Conference on Image Processing (ICIP'97). He is currently an Associate Editor of the IEEE TRANSACTIONS ON IMAGE PROCESSING and is a Guest Editor of a special issue on image and video processing in digital libraries to be published in the same TRANSACTIONS in 1999.