VII. CONCLUSION

In this paper we presented an integrated schema for semantic object segmentation and content-based object search based on the region-based video object model. We first discussed AMOS, a generic video object segmentation system which combines low level automatic region segmentation with user input for defining and tracking semantic video objects. Our experiments and performance evaluation have shown very good segmentation results. Using the region-based video object model, an object query model which effectively combines local region-level features and spatial-temporal structures is then presented. Experiments have shown promising results and great potential for developing advanced video search tools for semantic video representations such as MPEG-4.

In the future work, we will include multiple objects tracking, which can potentially be supported by our current algorithms. It would also be desirable to develop a hybrid strategy where video objects with simple motion and background can be tracked faster with less computation. Building upon the visual searching tools of video objects, we will explore object level video search and study application of the object search tools to MPEG-4 scene descriptions (i.e., BIFS).

REFERENCES

[1] M.P. Dubuisson and Anil K. Jain, "Contour Extraction of Moving Objects in Complex Outdoor Scenes", International Journal of Computer Vision, Vol. 14, pp.83-105, 1995.

[2] E. Saber, A.M. Takalp, & G. Bozdagi, "Fusion of Color and Edge Information for Improved Segmentation and Edge Linking", ICASSP'96, pp.2176-2179, Atlanta, GA, May 1996.

[3] M. Tabb and N. Ahuja, "Multiscale Image Segmentation by Integrated Edge and Region Detection", IEEE Transactions on Image Processing, Vol 6, No. 5, pp.642-655, May 1997.

[4] M.Flickner,H.Sawhney,W.Niblack,J.Ashley,Q.Huang,B.Dom,M. Gorkani,J.Hafner,D.Lee,D.Petkovic,D.Steele, P.Yanker, "Query by Image and Video Content: The QBIC System", IEEE Computer Magazine, Vol.28, No.9, pp.23-32, Sep 1995.

[5] A.Petland,R.W.Picard, and S.Sclaroff, "Photobook: Content-based Manipulation of image database", international Journal of Computer Vision, Vol.18, No. 3, pp.233-254, 1996.

[6] J.R.Smith and S.F.Chang, "VisualSEEK: a fully automated content-based image query system", ACM Multimedia 96, pp.87-98, Boston, MA, Nov 20, 1996.

[7] S.-F. Chang, W. Chen, H. Meng, H. Sundaram, and D. Zhong, "*VideoQ*: An Automated Content-Based Video Search System Using Visual Cues", ACM 5th Multimedia Conference, pp.313-324, Seattle, WA, Nov. 1997.

[8] Y. Deng and B. S. Manjunath, "Content-based search of video using color, texture and motion," ICIP'97, pp.534-537, Santa Barbara, California, October1997.

[9] N. Dimitrova and F. Golshani, "RX for Semantic Video Database Retrieval," ACM Multimedia Conference, pp.219-226, San FRancisco, Oct. 1994.

[10] J.Y.A. Wang and E. H. Adelson, "Representing Moving Images with Layers", IEEE Transactions on Image Processing, Vol 3, No. 5, pp.625-638, Sept 1994.

[11] A. Ayer, and H.S. Sawhney, "Layered Representation of Motion Video Using Robust Maximum-Likelihood Estimation of Mixture Models and MDL Encoding", Proc.Fifth Int'l Conf. Computer Vision, 1995, pp777-784.

[12] F. G. Meyer and P. Bouthemy, "Region based tracking using Affine Motion Models in Logn Image Sequences", CVGIP: Image Understanding, Vol. 60, No. 2, Spet, pp. 119-140, 1994.

[13] M. Kass, A. Witkin and D. Terzopoulos, "Snakes: Active contour models", International Journal of Computer Vision, pp. 321-331, 1988.

[14] B. Bascle, P.Bouthemy, R.Deriche and F. Meyer, "Tracking complex primitives in image sequence", 12th International. Conference on Pattern Recognition, 1994, pp.426-431.

[15] A. Blake, R. Curwen and A. Zisserman, "A framework for spatiotemporal control in the tracking of visual contours", International Journal of Computer Vision, 11(2):127-145, 1993.

[16] C. Gu and M.-C. Lee, "Semantic Video Object Segmentation and Tracking Using Mathematical Morphology and Perspective Motion Model", ICIP'97, pp.514-517, October 1997 Santa Barbara, CA.

[17] K. Zhang, M. Bober and J. Kittler, "Motion Based Image Segmentation for Video Coding", Proceedings of the International Conference on Image Processing, Washington (ICIP'95) Los Alamitos, CA, USA, pp.476-479, 1995.

[18] D.Zhong and S.-F.Chang, "Video Object Model and Segmentation for Content-Based Video Indexing", ISCAS'97, pp.1492-1495, HongKong, June 9-12, 1997.

[19] D.Zhong and S.-F.Chang, "Spatio-Temporal Video Search Using the Object Based Video Representation", ICIP'97, pp.21-24, October 26-29, 1997 Santa Barbara, CA.

[20] J.Meng,Y.Juan and S.F.Chang, "Scene Change Detection in a MPEG Compressed Video Sequence", Proc. SPIE Vol. 2419, pp.14-25, San Jose, Feb 1995.

[21] M. Ortega, Y. Rui, K. Chakrabarti, S. Mehrotra and T. Huang, "Supporting Similarity Queries in MARS," ACM Multimedia '97, pp.403-413, Seattle, WA, Nov. 1997.

[22] S.-F. Chang, W. Chen, "Semantic Visual Templates: Linking Visual Features to Semantics", ICIP'98, pp.531-535, Chicago, IL, October, 1998.

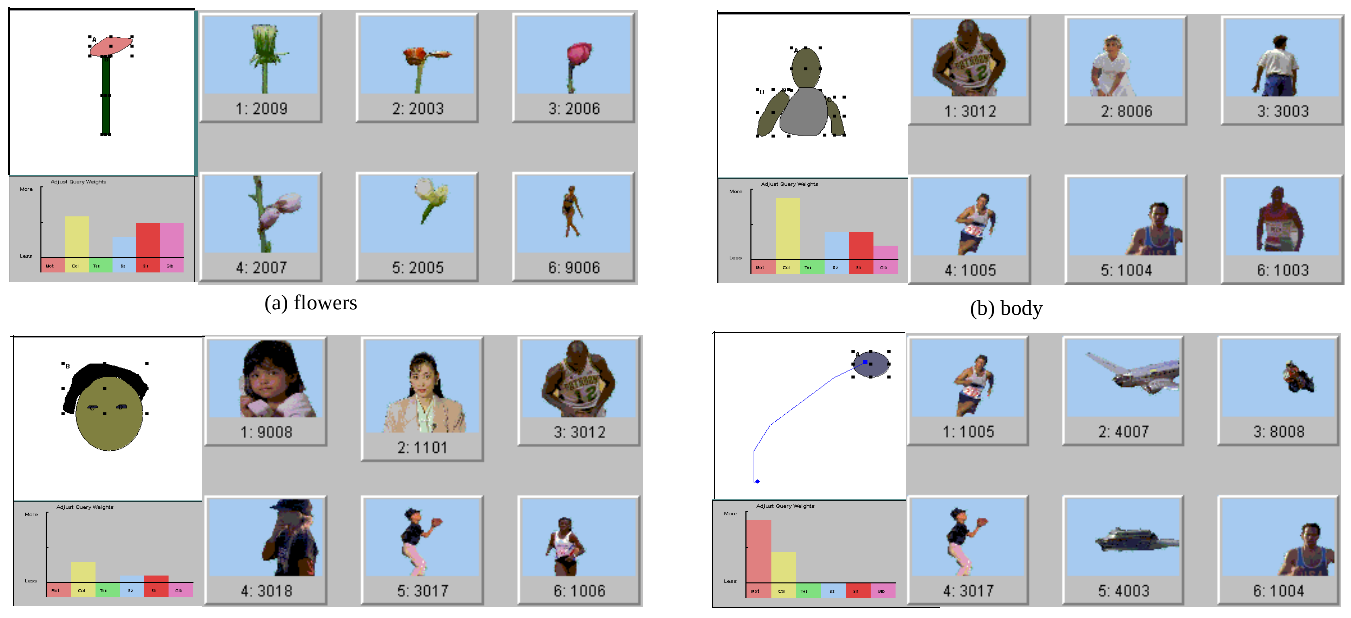
[23] M. Bierling, "Displacement Estimation by Hierarchical Block Matching", SPIE Vol 1001, Visual Communication & Image Processing, 1988, pp.942-951.

[24] S.-K. Chang, Q. Y. Shi, and C. Y. Yan, "Iconic indexing by 2-D strings", IEEE Trans. Pattern Anal. Machine Intell., 9(3):413-428, May 1987.

[25] A. D. Bimbo, E. Vicario and D. Zingoni, "Symbolic description and visual querying of image sequences using spatio-temporal logic", IEEE Transactions on Knowledge and Data Engineering, Vol 7, No. 4, pp.609-622, August, 1995.

[26] V.N. Gudivada and V.V. Raghavan, "Design and Evaluation of Algorithms for Image Retrieval by Spatial Similarity", ACM Transaction on Information Systems, Vol.13, No,2, pp.115-144, April 1995.

[27] C.-S. Li, J.R. Smith, L. Bergman and V. Castelli, "Sequential Processing for Content-based Retrieval of Composite Objects", SPIE Storage&Retrieval of Image/Video DB, 1998, pp.2-13.



(a) flowers (b) body

(c) face (d) trajectory

Figure 9. Video object query examples

ate this problem and improve searching accuracy, we develop a novel query time region merging process (**Figure 8**). For each candidate region list, the query system will merge regions from a same video object into a large "virtual region" if 1) they spatially connected with each other; 2) the merged region is closer to the query region (feature distance). When a "virtual region" is generated, it is added to the current candidate region list.

3) Perform join (outer join) of the region lists on *ObjectID* to create a candidate object list. Each candidate object in turn contains a list of regions. A "NULL" region is used when :
   - a region list doesn't contains regions with the *ObjectID* of a being-joined object
   - a region appears (i.e. matched) more than once in a being-joined object
4) Compute the distance between the query object and each object in the candidate object list as follows:

$$D = w_0 \sum_i FD(q_i, r_i) + w_1 SD(sog_q, sog_o) + w_2 SD(topo_q, topo_o) + w_3 SD(temp_q, temp_o) \qquad (3)$$

where q_i is the ith query region. r_i is the ith region in a candidate object. FD(.) is the feature distance between a region and its corresponding query region. If r_i is NULL, maximum distance (i.e., 1) is assigned. sog_q (spatial), $topo_q$ (topological) and $temp_q$ (temporal) are structure features of the query object. sog_o, $topo_o$ and $temp_o$ are retrieved from database using indices on *ObjectID*, *RegionID*. When there is a NULL region (due to the above join process), the corresponding dimension of the retrieved feature vector will have a NULL value. SD(.) is the L1-distance and a penalty of maximum difference is assigned to any dimension with a NULL value.

5) Sort the candidate object list according to the above distance measure D (Eq. 3) and return the result.

C. Query Results

A prototype system has been developed to demonstrate and evaluate our proposed visual similarity searching method which integrates matching of localized feature matching and spatial-temporal structures. We created a semantic video object database with 104 video objects from different types, including people, sports, animal, flowers and transportation. About 500 salient regions and their visual features are extracted and stored in the database.

Our query experiments have shown encouraging results. The matching of structure features has been proved to be critical in finding objects with multiple parts and special relationships (e.g. human body). Four query examples are shown in **Figure 9.** In the first example (**Figure 9a**), we are trying to find flowers by specifying two regions with color, shape and spatial relationship. Higher weight is put on the shape feature. Our experiments show that the shape matching of the stem part plays a key role in successfully finding flowers. In the second example (**Figure 9b**), we define a more complex query object, which has four parts with specific spatial relationship. While color and shape features are important in finding similar feature regions for each part, the matching of spatial structure eliminates false candidates and brings correct objects to the top of the return list. **Figure 9c** shows an example of partial matching. Users draw a face with two eyes. In the feature library, small regions like eyes are usually not extracted. Using the outer join and partial match method, we are able to find similar faces including those do not have extracted eye regions (results except the first one shown in **Figure 9c**). **Figure 9d** shows an example of trajectory matching. The retrieved objects are moving to the left-bottom direction on the screen.

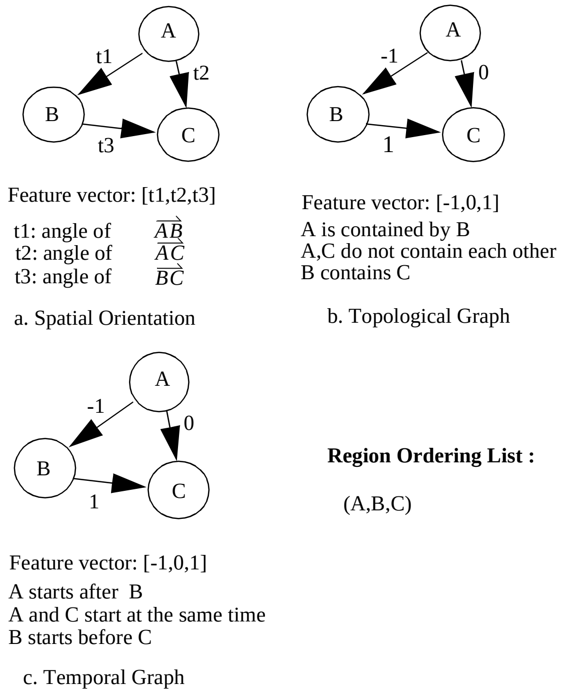


Figure 7. Examples of the three relationship graphs

stored so that they can be quickly accessed and examined in the similarity matching process.

There are several different ways to compare spatial structures, such as 2D-Strings [24] and spatial-temporal logics [25]. We use a relatively fast method, the spatial-orientation graph, to represents spatial relationships as edges in a weighted graph [26]. This graph can be easily constructed by computing the orientation of each edge between the centroids of a pair of query regions, and stored as a n*(n-1)/2-dimension feature vector, where *n* is the number of query regions (**Figure 7a**). As the spatial-orientation graph cannot handle the "contain" relationship, we extend it with a topological graph (**Figure 7b**), which defines the contain relation between each pair of regions with three possible values: contains(1), is-contained(-1) or not containing each other(0). Similarly, the temporal graph (**Figure 7c**) defines whether one region starts before(1), after(-1) or at the same time(0) with another region. Note here for simplicity we only define the temporal order according to the first appearing time (or starting time) of each region. By taking the ending time, a more complicated temporal relation graph can also be generated. These structure features are computed from ordered region lists, and each feature vector is stored together with its region ordering list (e.g. ABC in **Figure 7**). This provides an index for the sub-graph matching issue that will be met in the object query process.

B. Region-Based Object Query Model

Given a query object (composition of a set of query regions), there are generally two searching approaches. One is to directly match query object against objects in the database based on indexing techniques such as R-Tree. However, they are usually not suitable for a large set of high-dimension features.

The other searching approach is to first find a matching region list for each query region based on visual features, and then "join" these region lists to find the best matched video objects by combining visual and structure similarity measures. We used this approach in our earlier work, VisualSEEK [6]. In [27], Li and Smith further proposed a querying scheme which divides a composite query into a sequential of representation of sub-queries. A fast dynamic programming method is then used to retrieve the best matches for a composite object. However, sequentialization of a large set of visual and structure features is difficult. In this paper, we use a parallel query and join scheme which supports partial matches.

Given a query object with N regions, the object searching process consists of three stages. The first stage, *region search*, is to find a candidate region list from the database for each query region. The second stage, *region merge*, is to merge regions from a same video object into a large "virtual region". The final stage, *join & validation,* is to join the candidate region lists to produce the object candidate list and to compute the final global distance measure. The detailed procedure is given as follows:

1) For every query region, find a candidate region list based on the weighted sum of distance measures of different visual features. Only regions with distances smaller than a *threshold* are added to a candidate list. Here the *threshold* is a pre-set value used to empirically control the number or percentage of objects the query system will return. For example, a threshold 0.3 indicates that users want to retrieve around 30 percent of video objects in the database[1]. The threshold can be set to a large value to ensure completeness, or a small value to improve speed.
2) Sort regions in each candidate region list by their *ObjectID's,* and perform query time region merge. As users don't know exactly which segmented regions are included in the database, query regions may not match regions in the database on some visual features, e.g., size and shape. To allevi-

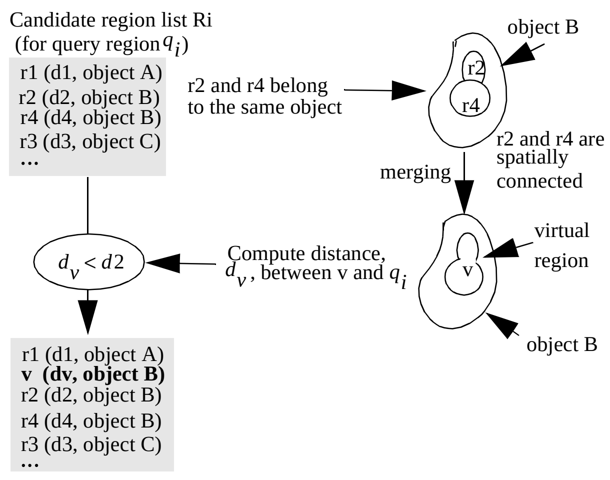
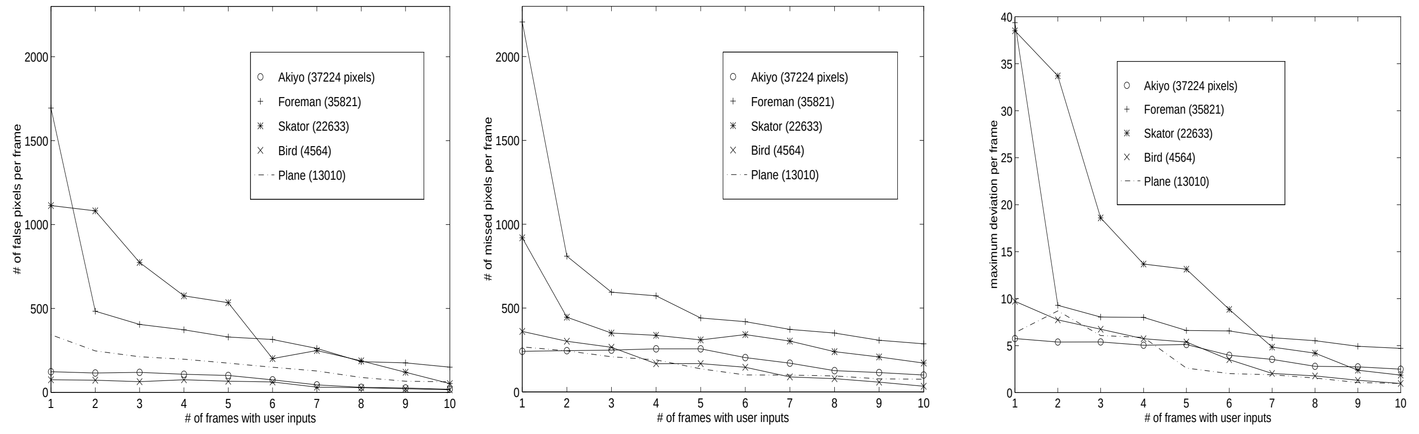


Figure 8. Query time region merging process

1. Assume the feature vectors of the video objects in the database have normal distribution in feature spaces.



a. Average # of false pixels b. Average # of missed pixels c. Maximum boundary deviations

Figure 6. Objective evaluation over 100 frames (numbers in the legends are the average region sizes)

tional user inputs bring very small improvement. The bird sequence contains a small, fast-moving object. The abrupt shape change between successive frames causes the missing of part of the wings in some frames. These errors are corrected after 2-3 more user inputs. For the plane sequence, after the first input, most false pixels come from a black strip at the left frame border. Part of the strip is merged into the plane in some successive frames. This error is easily corrected after 1-2 more user inputs.

The foreman and skater sequences have relatively large errors at the first user input, due to complex object motion. Missing pixels are mainly caused by the emerging of new foreground regions around the frame border (e.g., the left shoulder in the foreman sequence). These new regions connect with old object regions only by a few pixels when they first appear and they also abruptly change the shape of object. Thus they are classified as background regions during the tracking process. False pixels are included mainly because of uncovered background regions enclosed by the object. In the skater sequence, frame 50 has a background region included in the object. The background region is uncovered when the two legs move apart. As it is enclosed by the object and the frame border and thus not connected to any exist background regions, the new background region is falsely classified as foreground. The above two types of errors can be corrected by one user input at the frame where the error starts to happen. For the foreman and skater sequence, the numbers of missing and false pixels drop rapidly after the first 2 user inputs. And after this, maximum boundary deviations are generally within 10 pixels.

Generally the segmentation speed depends on the object size and the complexity of the scene. For a typical object like *akiyo* in CIF size images, it takes around 20 seconds per frame on a SUN UltraSparc-2 workstation. Note this includes all the computation processes described in the system architecture. Also it is based on a pure JAVA implementation of all segmentation and tracking processes in the system without speed optimization. Optimization may be performed to determine critical processes and ignore non-critical processes to reduce computations. Parallel processing can greatly improve the speed of our system. As one can see, the computations of three feature maps, which are the most computation intensive parts of the system, can easily be done in parallel. The main region segmentation and tracking algorithm can also have parallel implementation, as region merging is accomplished by examining local minimums.

We have been using this system to extract more than 100 video objects, and building a video object database for many object-based video applications such as MPEG-4 compression, content-based retrieval (which we will discussed in Section VI), and network transmission.

VI. REGION FEATURE BASED OBJECT QUERY

Although in the AMOS system, the underlying regions of video objects are already created and tracked during the object segmentation process, these regions usually have small sizes and short durations in order to achieve accurate object boundaries. Thus extra efforts (e.g., grouping or user interaction) are required to create salient feature regions that can facilitate efficient object retrieval.

Here we developed an extra module in the AMOS to preform a second pass region tracking inside video objects. As the segmentation is within the masks of a tracked object, this process is relatively simple, and can be done automatically. After the salient region segmentation, all segmented region and their corresponding objects are stored as a sequence of masks for the following feature extraction and matching process.

A. Visual and Structure Features

Similar to those in the VideoQ system, a large set of visual features of video objects and their underlying regions are computed and stored in a visual feature library. Detailed descriptions of the features can be found in [7]. A feature vector is stored together with its corresponding *ObjectID* and *RegionID*. These *ID's* are used as index in the following object matching and retrieval process.

Given a set of regions of an object, the structure features can be derived from their spatial and temporal positions and boundaries. These features need to be pre-computed and properly

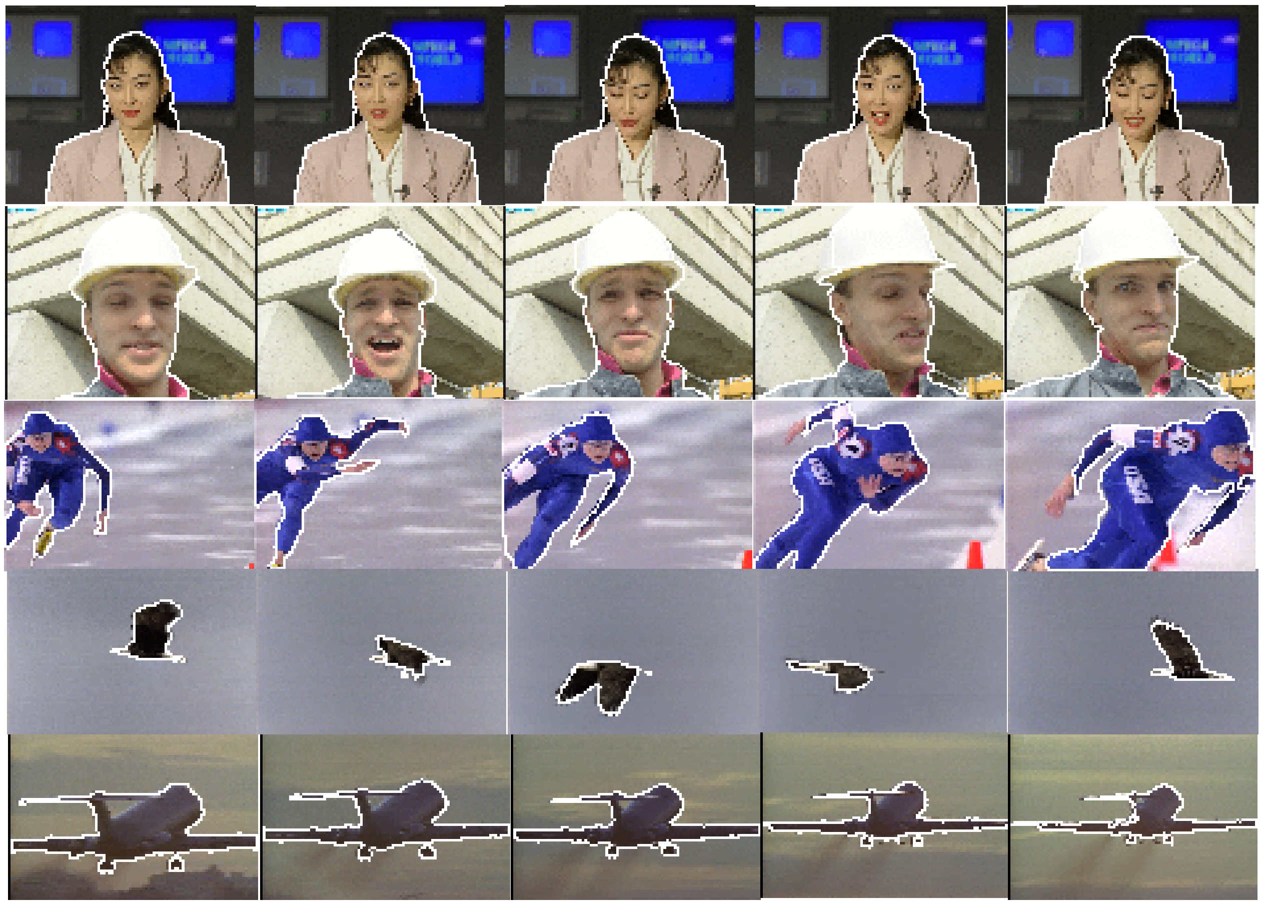


Figure 4. Object tracking results of five sequences after 3 user inputs (from left to right, frame # 1,25,50,75,100)

they are composed with the background with random noise. For the akiyo sequence, there are no noticeable errors after only one input at the starting frame. For the foreman, bird and plane sequences, three user inputs give us outputs without noticeable errors. The skater sequence, which has fast and complex motion, requires 4 user inputs to remove noticeable errors. These subjective evaluations are confirmed and further explained in detail in the following objective evaluation experiments.

In the objective evaluation, we manually extracted semantic objects in each frame over 100 successive frames, and considered these as the ground truth. We then computed the average numbers of missing pixels, false pixels and maximum boundary deviations between the ground truth and the segmentation results.

Numbers of missing and false pixels are simply computed by comparing a segmented object mask with its related ground truth. While there are different ways to define the boundary deviation, we define the deviation for a missing pixel, as the distance between this pixel and its nearest foreground pixel in the segmented object mask; and for a false pixel, as the distance between this pixel and its nearest foreground pixel in the ground truth mask (**Figure 5**). The maximum boundary deviation is the maximum value of the deviations of all missing or false pixels.

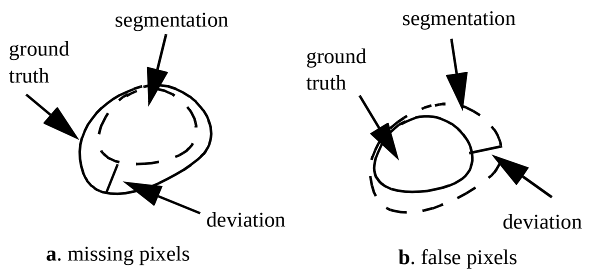


Figure 5. The boundary deviations

The performance results are shown with different numbers of user inputs in **Figure 6**. Main tracking errors are usually caused by new or uncovered background or foreground regions. This is clearly reflected in **Figure 6c**, where the maximum deviations are around 40 pixels when a large region is missed or falsely included. As shown in the plots, such errors are corrected after 2 or 3 user inputs. The remaining errors may be considered an accuracy limitation of our segmentation and tracking algorithm. The number of false and missing pixels can be reduced from 30 to 200 pixels depending on the object size. The maximum deviation curves show that these missing and false pixels are less than 5 pixels away from the ground truth. Considering inherent errors caused by boundary blur (especially for the MPEG sequences with fast motion) and manual segmentation, our system generates very good tracking results.

As shown in **Figure 6**, the Akiyo sequence only requires one user input to obtain good tracking result over 100 frames. Missing and false pixels are mainly around the hair, which is not clearly separated from background by color or by edge. Addi-

B. Region Tracking

Unlike existing approaches, projected regions are not used directly as the new segmentation, but as seeds in another color based region growing process to track existing regions. First, non-edge pixels are walked through and labeled one by one from left to right and top to down. As in the common labeling process, if a pixel has the same color as that of its labeled neighboring pixels, it is assigned the label as its neighbors. When the color of a pixel is different from the colors of its labeled neighboring pixels, its color is compared with all projected regions that cover its coordinate at the current frame. If its color distance to the closest region is below the given color threshold, this pixel is "tracked", and assigned the label as well as the classification (i.e., *foreground or background*) of the closest projected region. Otherwise, a new label is generated and assigned to the pixel. Regions identified by this new label are classified as "*new*" regions.

The subsequent color merge, edge labeling, motion split and small region elimination processes are similar to those in Section III with some additional constraints. *Foreground* or *background* regions tracked from the previous frame are allowed to be merged only with regions of the same class or *new* regions, but merging between a *foreground* region and a *background* region is forbidden. *New* regions can be merged with each other or merged with *foreground/background* regions. When a new region is merged with a tracked region, the merged result inherits its label and classification from the tracked region.

C. Region Aggregation and Object Composition

As shown in **Figure 3**, region aggregation includes two inputs: the homogeneous region and the estimated object boundary. The object boundary is estimated from projected foreground regions. Foreground regions from the previous frame are projected independently of one another and the combination of projected regions forms the mask of the estimated object. The mask is refined with a morphological closing operation (i.e. dilation followed by erosion) with a size of several pixels in order to close tiny holes and smooth boundaries. To tolerate motion estimation errors, which are common for general video sources, the resulting mask is further dilated for the tracking buffer size, which is specified by users at the beginning of tracking. Generally a larger buffer size is required for objects with fast motion or abrupt shape change.

The region aggregation module implements an iterative region grouping and boundary alignment algorithm based on the estimated object boundary as well as the edge and motion features of the region. The algorithm is described as follows.

For every segmented region, if the region is tagged as *background*, keep it as *background*. If it is a *foreground* or *new region*, we compute the intersection ratio of the region with the object. If a *foreground* region is covered by the object mask by more than certain percentage (e.g., 80%), it is kept as *foreground*; otherwise, it is intersected with the object mask

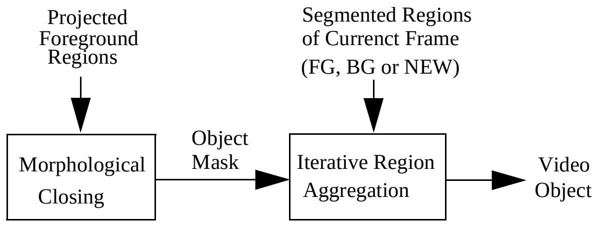


Figure 3. Region Aggregation Using Projected Object

and split into one foreground region and one new region. We use a relatively lower ratio (around 80%) here to include a foreground or new region, as the project object boundary is not as accurate as the initial boundary given by users.

For a *new* region, if it is covered by the object mask by more than certain percentage (e.g.,80%), it is grouped into the object as *foreground;* if the intersection ratio is very small (e.g., less than 30%), it is kept as *new*. Otherwise, visual similarity (including edge and motion) between this region and its neighbors is examined. The region is grouped into the object as *foreground* if the region is separated from background regions by more edge pixels than foreground regions (or this region is not connected to any background regions), and its closest neighbor according to the motion feature (e.g., mean motion vector) is a *foreground* region.

The above aggregation and boundary alignment process is iterated multiple times (e.g., 2 or 3) to handle possible motion projection errors, especially for fast motion. At the end of the last iteration, all remaining *new* regions are classified into background regions.

V. SEGMENTATION EVALUATION

The AMOS system has shown very good segmentation and tracking results on general video sources. As shown in **Figure 4**, five video sequences with different types of motion and background are used to do subjective and objective evaluation of our system.

After the first user input at the starting frame, more user inputs are applied at subsequent frames where the largest tracking errors occur or start to occur to refine the tracking results. Notice that the effort users need to correct boundary errors are much less than the initial object definition. As we will discuss below, major tracking errors come from uncovered background or foreground regions, as such errors usually propagate to the subsequent frames. Thus, it is obvious for users to identify such frames for correction. Users can also at the beginning provide multiple inputs at the frames where errors are expected to occur. This will reduce the user intervention and attendance in the tracking process.

Figure 4 shows object tracking results of the five testing sequence after 3 user inputs, which gave us acceptable results for all five sequences. For subjective evaluation, segmented objects are superimposed onto gray background with random noise and played in real time for users to see whether there are observable errors. To remove boundary jitter (i.e. high frequency noise), a temporal median filtering process (with a radius of 2 frames) is applied to the binary object masks before and non-edge-pixels are set to 0. It is generated by applying the Canny edge detection algorithm. Finally, the motion field is generated by a hierarchical block matching algorithm [23].

C. Region Segmentation

The segmentation algorithm is developed based on the three feature maps: color map, edge map and motion field. Departing from old merge-and-split methods where the edge is applied after color-based region merge, we propose a new method to fuse edge information directly in the color merging process.

First, a color-based pixel labeling process is applied to the color map. Labeling is a process where one label is assigned to a group of neighboring pixels with the same (or very similar) color. To prevent assigning one label to two regions with the same color but separated by edge pixels, only non-edge pixels (i.e. pixels that are set to 0 in the edge mask) are labeled in the process. Edge pixels remains un-labeled. This process generates an initial group of regions (i.e. pixels with the same label) as well as their connection graph. Two regions are linked as neighbors if pixels in one region has neighboring pixels in another region.

The color merging is an interactive spatial-constrained color clustering process. Color distances (Eq. 1) between every two connected regions are computed. Two connected regions (e.g., i and j) are merged if the color distance between them is (1) smaller than the given color threshold; and (2) the local minimal, (i.e. it is smaller than all the other distances between these two regions and their neighbors). Once a new region is generated from two adjoining regions, its mean color m is computed by taking weighted average of the mean colors of the two old regions (m_1, m_2),

$$m^c = \frac{(m_1^c s_2 + m_2^c s_2)}{(s_1 + s_2)} \qquad (2)$$

where c is *L*, u* or v**; sizes of the two old regions s_1, s_2 are used as weights. The region connections are also updated for all neighbors of the two old regions. The new region takes the label of the larger one of the two merged regions. Then the two old regions are dropped.

The merging is iterated until color distances between every two connected regions are above the color threshold. As edge pixels are not labeled, two regions separated by edge pixels are not connected as neighbors. Thus, the growth of each region is naturally stopped at its edge pixels. Note that (1) some missing edge pixels will not cause a false merge of two neighboring regions provided that their colors are different; (2) short edges inside a large homogeneous color region do not stop the merging process, as the region connection graph can "walk around" such internal edges. These properties ensure that the merging process is not sensitive to noises which usually exist in edge maps. After the color merging process, edge pixels are simply assigned to their neighboring regions with the smallest color distances.

To ensure homogeneous motion, an optional motion-based segmentation using dense optical flow is applied to the segmented color regions to check the uniformity of the motion distribution. A similar clustering process is used to group pixels inside a color region according to their motion vectors and the given motion threshold.

D. Region Aggregation

The region aggregation module takes homogeneous regions from the segmentation and the initial object boundary from the user input. Aggregation in the starting frame is relatively simple, as all regions are newly generated (not tracked) and the initial outline is usually not far from the real object boundary. A region is classified as foreground if more than a certain percentage (e.g., 90%) of the region is included by the initial object boundary. On the other hand, if less than a certain percentage (e.g., 30%) of a region is covered, it is considered as background. Regions between the low and high thresholds are split into foreground and background regions according to the intersection with the initial object mask. These thresholds depend on the boundary accuracy of user inputs.

Finally, affine motion parameters of all regions, including both foreground and background, are estimated by a multivariate linear regression process over the dense optical flow inside each region. They are stored and will be tracked over time in the successive frames. The reason for tracking background regions is to improve the object boundary accuracy. This has proven very useful, especially when the background is complex or contains other moving objects.

IV. SEMANTIC OBJECT TRACKING

Segmented regions from the previous frame, including both foreground and background, are first projected onto the current frame using their individual affine motion models. Projected regions keep their labels and original classifications. Then an inter-frame segmentation process as follows is applied to track the semantic object.

A. Generation of Feature Maps

Generation of the three feature maps (color, edge and motion) utilizes the same methods as we described in the previous section. The only difference is that in the quantization step, the existing color palette computed at the starting frame is directly used to quantize the current frame. Using a consistent quantization palette enhances the color consistency of segmented regions between successive frames, and thus improves the performance of region tracking. As object tracking is limited to single video shots, in which there is no abrupt scene change, using one color palette is generally valid. Certainly, a new quantization palette can be automatically generated when a large quantization error is encountered.

library, the system supports sketch based visual queries in which users can define one or more query objects with various visual features.

Although VideoQ supports localized feature matching, it mainly uses low-level uniform feature regions. These regions do not necessarily correspond to meaningful real-world objects. While it has the advantage of being fully automatic, further efforts have to be made to support high level semantic queries [22]. Under the MPEG-4 framework, we have semantic objects available in video streams, and this gives us a new opportunity to extend the visual searching methods to high-level queries.

Semantic video objects introduce one more description level in the visual feature library of VideoQ. While existing techniques can be applied to similarity search of semantic video objects, there are several new critical issues. First, underlying regions of a video object are tightly connected with each other, and thus similarity matching of spatial-temporal structures become more important and should be performed more precisely. Secondly, the problem of partial matching between query regions and object regions stored in the feature library need to be solved. In the previous work, query regions can be a subset of regions of a matched object (or video clip). However, a matched object has to contain matches for all query regions. This implicitly limits the number of regions that a user can use to define an object. Finally, there are efficiency concerns, especially when using a rich set of temporal and spatial structure features. It is time-consuming to compute and match them in query time. They need to be pre-computed and properly indexed for efficient query processing.

Here we would like to address the problem of similarity matching of video objects using localized visual features. Based on the same object model and the segmentation framework in the AMOS system, we extend our prior work, VideoQ, to develop a new content-based search system for semantic objects. The system contains three basic processes. The first step is to create salient regions suitable for object searching. The second step extracts visual features at both object and region levels and builds a visual feature library for segmented video objects. The last one is an object searching process which effectively searches the visual feature library, combines global and local features and examines the spatial-temporal structures of video objects.

III. SEGMENTATION IN THE INITIAL FRAME

The system diagram of semantic object segmentation at the starting frame is shown in **Figure 2**. It consists of four major processes as follows.

A. Object Definition and Threshold Specification

First, the user identifies a semantic object by using tracing interfaces (e.g., mouse). The input is a polygon whose vertices and edges are roughly along the desired object boundary.

After the object definition, users can specify a set of parameters to start the tracking process. These parameters include a color merging threshold, weights on the three color channels, a motion merging threshold, and a tracking buffer size. Usually users may just rely on the default values for these parameters. Determination of these parameters will be explained in more detail in the following sections. These parameters can be optimized based on the characteristic of a given video shot and experimental results. The segmentation results are not sensitive to slight changes of parameter values. However, optimization may be used to reduce the computation complexity. The system also allows a user to stop the tracking process at any frame, modify the object boundary that is being tracked, then restart the tracking process from the modified frame.

B. Generation of Feature Maps

The three feature maps, edge map, color map and motion field are created from the original images. The color map is the major feature map in the following segmentation module. We have developed a novel process to generate it. This process contains the following steps: First, the original image is converted into CIE L*u*v* color space. The color difference is thus given by the weighted Euclidean distance in the three dimensional space, i.e.

$$\Delta s = \sqrt{w_L(\Delta L^*)^2 + w_u(\Delta u^*)^2 + w_v(\Delta v^*)^2} \qquad (1)$$

where w_L, w_u, w_v are weights given by users. Our experiments show that it generally generates better results to put higher weights on chrominance channels (e.g., two times higher than that of the luminance channel). The aforementioned color threshold is chosen according to this distance measure.

Then the L*u*v* image is smoothed to remove noise, as well as tiny detail, for the purpose of region merging. This is done by an adaptive quantization and median filtering process. We use a clustering-based (e.g., K-Means) method to analyze the input image and determine an adaptive quantizor in the L*u*v* space. After quantization, a median filtering process is applied on each of the L*u*v* channels to produces the final color map.

The edge map is a binary mask where edge pixels are set to 1

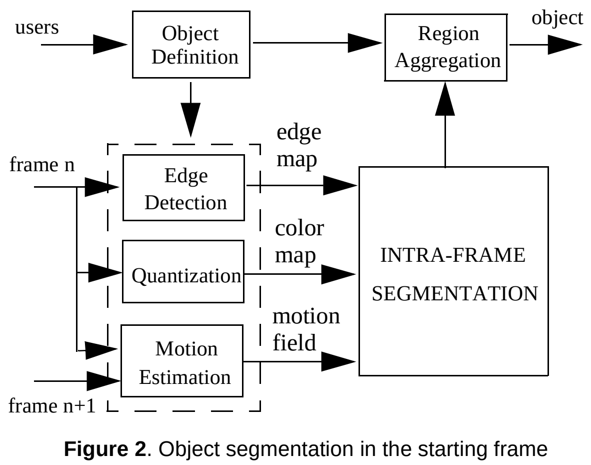


Figure 2. Object segmentation in the starting frame

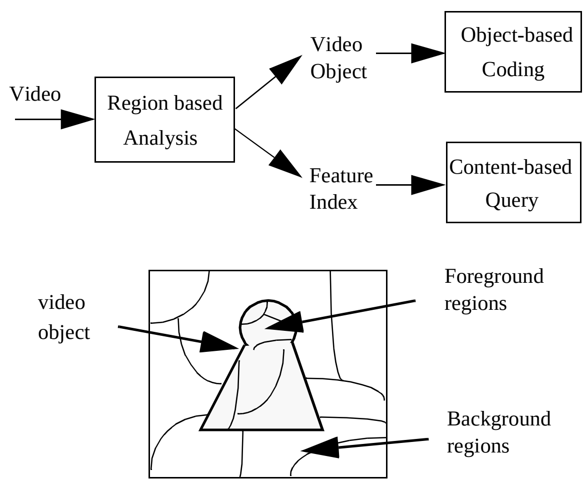


Figure 1. An integrated approach for video object segmentation and search based on region analysis

to semantic objects, (2) extracting accurate object boundaries and salient indexed regions simultaneously, and (3) effective similarity matching at multiple levels. We first review these issues in the following subsections.

A. Active Video Object Segmentation

Many approaches have been developed to segment moving objects using the motion field or optical flow. Wang and Adelson in [10] presented an affine-clustering based algorithm. In [11], instead of using optical flow, Ayer and Sawhney proposed a method to estimate motion models and their layer support simultaneously. In [12], Meyer and Bouthemy developed a pursuit algorithm to track an object based on the multi-resolution estimation of the affine model from the motion field within the object. In general, the above methods concentrate on segmenting moving objects and cannot track static objects or objects with intermittent motions (e.g., people stop and move while crossing the street). Furthermore, due to the accuracy limitation of motion estimation, motion segmentation may not give clear object boundaries.

To track non-rigid objects, deformable models have been widely studied. Active contour (i.e., snakes) [13] is one of the basic energy-minimizing elastic contour models. As snakes require very accurate initialization (thus can handle only slow motion) and are sensitive to textured image regions, many improvements [14,15] have been developed.

Recently, with the demand for object tracking in general videos and the requirement of more accurate segmentation boundaries, region-based methods, which combine common image segmentation techniques with motion estimation methods, have been reported in [1,16,17]. In [1], Dubuisson and Jain presented an approach to combine motion segmentation using image subtraction with static color segmentation using the split-and-merge paradigm. In [17], an algorithm is developed to match edge detection and line approximation results with motion segmentation, and to determine the final object boundary. In [16], Gu and Lee proposed a semantic object tracking system using mathematical morphology and perspective motion.

Satisfactory results from the aforementioned region-based work were reported for certain type of video content, e.g., those with rigid objects and simple motions. However, these techniques usually track a single contour or video object, ignoring complex components and their associated motions within the object. In real-world video sources, an object usually contains several parts with different motions (sometimes non-rigid and with rapid changes). One single motion model is not adequate to track a semantic object. Meanwhile, these techniques still use the motion field as the main feature for tracking purposes. Static color or gray-level segmentation is fulfilled separately, and the fusion of the two segmentation results is done only at the final stage using certain heuristic rules. Due to the noisy nature of the motion field in real-world scenes, tracking results may also be error prone. As there are no constraints being applied between motion and static segmentation [1], when the two results are different from each other, it is hard to align them to generate the final object mask. Furthermore, these techniques tend to ignore the background content during the tracking process. This may cause problems in tracking regions near the boundary of the object.

To solve the above problems for general video sources, we developed an active system (**AMOS**) which uses an innovative method for combining low level automatic region segmentation and tracking methods [18,19] with an active method for defining and tracking video objects at a higher level. The system contains two stages: an initial object segmentation stage where user input at the starting frame is used to create a semantic object with underlying homogeneous regions; and an object tracking stage where homogeneous regions and the object are tracked through the successive frames. Note that the segmentation process is applied within individual video shots where there are no scene cuts. Automatic scene cut detection algorithms [20] can be used to cut a long video stream into short clips.

B. Region Feature Based Object Query

Content-based image and video retrieval has been studied by many researches in the recent years. Examples of systems include QBIC, PhotoBook, VisualSEEK, MARS and VideoQ [4,5,6,21,7]. These systems provide methods for content-based retrieval of images and videos by using query examples and sketches. Color, texture, shape, motion and spatial-temporal composition are popular visual features being used for visual similarity match. While QBIC, PhotoBook, VisualSEEK, Virage and many other systems support only still image retrieval, content-based video search has been explored in VideoQ [7,8].

Our previous work, VideoQ is a content-based video searching system which allows users to search video clips based on a rich set of visual features and spatio-temporal relationships. In this system, a fully automatic video region segmentation and tracking process is utilized to build a rich visual feature library including color, texture, shape, and motion. Based on this

An Integrated Approach for Content-Based Video Object Segmentation and Retrieval

Di Zhong and Shih-Fu Chang, *Member, IEEE*

Abstract - Object-based video data representations enable unprecedented functionalities of content access and manipulation. In this paper, we present an integrated approach using region-based analysis for semantic video object segmentation and retrieval. We first present an active system which combines low level region segmentation with user inputs for defining and tracking semantic video objects. The proposed technique is novel in using an integrated feature fusion framework for tracking and segmentation at both region and object levels. Experimental results and extensive performance evaluation show excellent results compared to existing systems. Building upon the segmentation framework, we then present a unique region-based query system for semantic video object. The model facilitates powerful object search, such as spatio-temporal similarity searching at multiple levels.

Index Terms **- object segmentation, region analysis, content-based retrieval, video object model, spatio-temporal structure**

I. INTRODUCTION

To meet the challenges of future multimedia applications, the newly established MPEG-4 standard has proposed a new object-based framework for efficient multimedia representation. This representation enables unprecedented, flexible access and manipulation functionalities of multimedia content. However semantic object segmentation and content-based object searching for general sources remain as two challenging tasks.

Image and video object segmentation has been a challenging task over decades. Although much work has been done in decomposing images into regions with uniform features [1,2,3], we are still lacking robust techniques for segmenting semantic video objects in general video sources; especially when accurate object boundaries are needed. In this paper, we first propose techniques for effective semantic object segmentation and tracking in general video sequences. We use "*semantic object*" to refer to video objects corresponding to meaningful real-world objects, in contrast with lower-level regions, which correspond to image areas with homogeneous features. In the rest of the paper, we use "semantic object" and "object" interchangeably.

Object-based video representation also provides a natural framework for content-based video retrieval. Although much research has been done in the area of content-based image retrieval in the past few years [4,5,6], content-based search of video objects has not been fully explored. In our prior work VideoQ [7], automatically extracted video regions and their visual features are used to build a visual feature library. The video regions and spatial-temporal features are then used in the similarity-based retrieval of video clips. Low-level feature similarity searching of video has been explored in other works [8,9] as well. While in VideoQ the feature extraction is a fully automatic process, only low-level salient feature regions are used. Semantic objects in MPEG-4 provide new potential for more powerful, high-level video search. Specifically, similarity searching of video objects at multiple levels (region vs. object vs. scene) introduces new important issues.

In this paper an integrated approach is proposed to address both video object segmentation and content-based retrieval. In this approach, a semantic object is modeled as a set of regions with corresponding spatial and visual features. This model directly links the semantic object to its underlying feature regions. For segmentation, the region-based method generates more accurate object boundaries and is also more robust in handling various real-world situations, including complex objects, fast and/or intermittent motion, multiple moving objects and partial occlusion. For content-based similarity search, underlying regions provide localized features as well as the spatial-temporal structure of a video object.

We first give an overview of our integrated approach compared with existing techniques in Section II. The object segmentation and tracking algorithms are discussed in Section III and IV. Extensive segmentation results and performance evaluation are given in Section V. In Section VI, we present a content-based searching system which addresses search and retrieval of semantic objects. The conclusion and future work are discussed is in Section VII.

II. APPROACH OVERVIEW

Automatic segmentation of semantic objects is difficult except for specific application domains. In general video, semantic objects may correspond to multiple regions which may have very great spatio-temporal variations. On the other hand, segmentation and tracking of uniform feature regions are considered more practical. The rich set of features and spatio-temporal structural information at the region level have also been proved to be effective in video retrieval [7]. Therefore, we propose an integrated region-based approach for video object segmentation and retrieval. As shown in **Figure 1**, the integrated region based analysis extracts the semantic objects that are suitable for object-based coding such as MPEG-4, and at the same time the salient features at both the region and object levels that are very useful for content-based query.

Key research issues lies in (1) linking the low-level regions