AMOS: An Active System For MPEG-4 Video Object Segmentation

Di Zhong and Shih-Fu Chang
Department of Electrical Engineering
Columbia University, New York, NY 10027, USA
{dzhong, sfchang}@ee.columbia.edu

Abstract

*Object segmentation and tracking is a fundamental step for many digital video applications. In this paper, we present an active system (**AMOS**) which combines low level automatic region segmentation with an active method for defining and tracking high-level semantic video objects. The system contains two stages: an initial object segmentation stage where user input in the starting frame is used to create a semantic object; and an object tracking stage where underlying regions of the semantic object are tracked and grouped through successive frames. Experiments with different types of videos show very good performance.*

1. Introduction

Recent progress in content-based video indexing and audiovisual object-based coding (MPEG-4) has shown great potential for many new applications, such as content-based image/video retrieval [1], content-based scalable video transmission, and object-based video editing. However, there is still a lack of robust techniques for segmenting video objects. State of the art work has shown success in decomposing video into regions with uniform features [4,7]. Segmenting video into meaningful objects (such as people, car) still remains a challenging issue. In this paper, we refer to this type of meaningful objects as "MPEG-4 video objects" or "semantic video objects" interchangeably.

Due to the semantic ambiguity and content complexity, fully automatic segmentation of semantic objects so far can only be developed for constrained domains. For general video sources, object definition usually comes from user manual input. Users identify semantic objects in the first frame by using tracing interfaces (*e.g.*, mouse or optical pen). Given the initial objects, the segmentation system tracks the movement of objects in subsequent frames. We call this type of system an active segmentation (in the sense that "active" user input is required).

Similar approaches using active segmentation methods have been reported in [2,3]. In [2], based on an active contour model (snake), an energy-minimizing elastic contour model was used to track the moving contour of the object. An elastic contour in the previous frame is moved and deformed iteratively to the best position in the current frame according to the energy function. In [3], Gu and Lee proposed a semantic object tracking system using mathematical morphology and perspective motion. Their system uses a modified morphological watershed procedure to segment uncertain areas between the interior and exterior outlines. Flooding seeds are sampled on both interior and exterior outlines. Regions growing from interior seeds define the segmented object boundary. In the first frame, the interior outline is defined by users, and the exterior outline is obtained by dilation of the interior one. For the subsequent frames, the interior and exterior outlines are created by erosion and dilation of motion projected boundaries from the previous frame.

Satisfactory results from the aforementioned work were reported for certain types of video content, *e.g.*, those with rigid objects and simple motion. However, these techniques track a single contour or video object, ignoring the complex components and associated motions within the semantic object. In general video sources, a semantic object usually contains several parts with different motions (sometimes with rapid changes and non-rigid). In such cases, one single motion model is not adequate to track a semantic object over time. In addition, these techniques tend to ignore the background content in tracking the foreground object. This may cause problems in tracking regions near the boundary of the object (as will be illustrated later).

We have developed an active system for MPEG-4 video object segmentation called **AMOS**. AMOS combines low level automatic region segmentation and tracking [6,7] with the active method for defining and tracking video objects at a higher level. At the low region level, image frames are decomposed into a set of non-overlapping regions with homogeneous visual features such as color, texture and motion. At the object level, semantic objects are obtained by aggregation of homogenous regions. The aggregation process uses the optimal combination of user input in the first frame, joint foreground and background simultaneous tracking, and new region classification based on visual feature similarity. Region-level segmentation is context independent and can be fully automated. Application of our automated region segmentation has been demonstrated successfully in our video search system, VideoQ [1]. This paper presents new methods for combining the automatic region segmentation methods with the active methods for defining and tracking video objects in a higher level.

This paper is organized as follows. We first give an overview of the system in Section 2. Section 3 presents static region segmentation and object aggregation in the starting frame. The automatic region based object tracking schema is discussed in Section 4. Experimental results are analyzed in section 5. Section 6 includes conclusion and discussion of future work.

2. AMOS Overview

In the AMOS system, a semantic object is represented as a set of underlying homogeneous regions. The goal of the system is to construct a semantic object according to user input in the starting frame and then track the object in the successive frames based on its feature regions. AMOS consists of two stages in semantic object segmentation (**figure 1**).

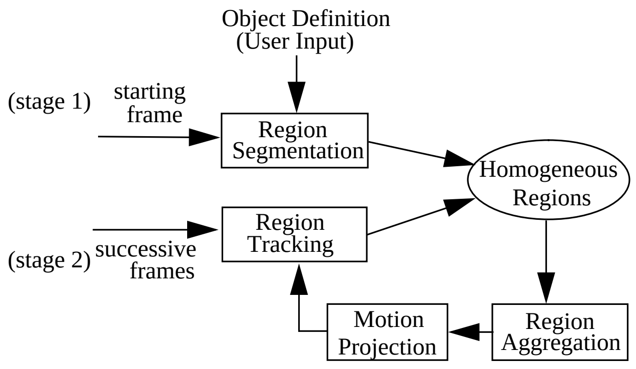

Figure 1. General Structure Of AMOS

In the first stage, definition of a semantic object (*e.g.* rough outlines) is provided by users in the starting frame. A snake algorithm[5] is provided as an option to refine the user input. Then the semantic object is generated through a region segmentation and aggregation process. Effective region segmentation is applied to the area inside the slightly expanded bounding box of the user-specified object. It effectively fuses color and edge features in a region-growing process that produces homogeneous color regions with accurate boundaries. To extract homogeneous regions in both color and motion, motion segmentation based on a dense motion field is used to further split the color regions. As color segmentation is more robust and gives more accurate region boundaries, it is applied before motion segmentation. Motion is used as a secondary feature due to the noisy nature of the motion field. This is important for the following aggregation procedure, where homogeneous regions are classified as either foreground or background to form the semantic object with an accurate boundary. Region aggregation is based on the coverage of each region by the initial object mask: regions that are covered more than a certain percentage are grouped into the foreground object. The final contour of the semantic object is computed from foreground regions. Foreground regions belonging to the object and background regions are all stored and will be tracked over time in the successive frames.

Tracking at both the region and object levels is the main task in the second stage. As shown in **figure 1**, segmented regions from the previous frame are first projected to the current frame using their individual affine motion models[6]. An expanded bounding box including all projected foreground regions is computed. Then the area inside the bounding box is split to homogeneous color and motion regions following a region tracking process. Unlike in other existing approaches, projected regions are not used directly as the new segmentation, but as seeds in another color based region growing process, which is similar to the fusing algorithm in the first stage. Pixels that can not be tracked from any old regions are labeled as new regions. Thus the resulting homogeneous regions are tagged either *foreground* (meaning tracked from a foreground region), or *background* (meaning tracked from a background region), or *new* (meaning not tracked). They are then passed to an aggregation process and classified as either belonging to the foreground object or the background. Here instead of the user input, the approximated object boundary is obtained from projected foreground regions. Furthermore, to handle possible motion estimation errors, the aggregation process is carried out iteratively. Finally, the object contour is computed from foreground regions and all regions are advanced to the tracking process for the next frame.

A detailed explanation of the above two stages is given in the following two sections.

3. Initial Semantic Object Segmentation

Semantic object segmentation at the starting frame consists of several major processes as shown in **figure 2**.

First, users identify 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. To tolerate user-input error, a snake algorithm [5] is used to align the user-specified polygon to the actual object boundary. The snake algorithm is based on minimizing a specific energy function associated with

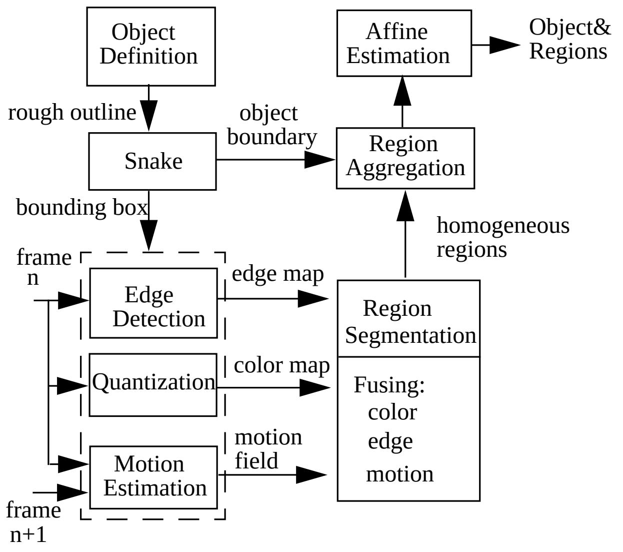

Figure 2. Object segmentation at the starting frame

edge pixels [5]. Users may also choose to skip the snake module if a relatively accurate outline is already provided.

After the object definition, users can start the tracking process by specifying a set of thresholds. These thresholds include a color merging threshold, weights on three color channels (i.e. L*u*v*), a motion merging threshold and a tracking buffer size (see following sections for their usage). These thresholds can be chosen based on the characteristic of a given video shot and experimental results. For example, for a video shot where foreground objects have similar luminance with background regions, users may put a lower weight on the luminance channel. Users can start the tracking process for a few frames with the default thresholds which are automatically generated by the system, and then adjust the thresholds based on the segmentation and tracking results. Our system also allows a user to stop the tracking process at any frame, modify the object boundary that is being tracked and then restart the tracking process from the modified frame.

Given the initial object boundary from users (or the snake module), a slightly extended (~15 pixels) bounding box surrounding the arbitrarily shaped object is computed. Within the bounding box, three feature maps, edge map, color map and motion field, are created from the original images. Color map is the major feature map in the following segmentation module. It is generated by first converting the original image into the CIE L*u*v* color space and then quantizing pixels to a limited number of colors (*e.g.* 32 or 16 bins) using a clustering based (*e.g.* K-Means) method. The edge map is a binary mask where edge pixels are set to 1 and non-edge-pixels are set to 0. It is generated by applying the Canny edge detection algorithm. Motion field is generated by a hierarchical block matching algorithm [8]. We adopted a 3-level hierarchy as suggested in [8].

The intra-frame segmentation module is developed based on our previous work on automatic region segmentation algorithm using color and edge [6,7]. As stated in [6,7], color-based region segmentation can be greatly improved by fusion with edge information. Color based region merging process works well on quantized and smoothed images. On the contrary, edge detection captures high-frequency details in an image. In AMOS, to further improve the accuracy, a motion-based segmentation process using the optical flow is applied to segmented color regions to check the uniformity of the motion distribution. Although the complete process utilizing color, edge, and motion is not trivial, the computational complexity is greatly reduced by applying the above region segmentation process only inside the bounding box of the snake object instead of the whole frame.

The region aggregation module takes homogeneous regions from the segmentation and the initial object boundary from the snake (or user input directly). Aggregation at the starting frame is relatively simple compared with that for the subsequent frames, 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 in the initial object. 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.

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. In our system, a 2-D affine model with 6 parameters is used. These affine models will be used to help track the regions and object in the future frames, as we will discuss in the next section.

4. Semantic Object Tracking

Given the object with homogeneous regions constructed at the starting frame, tracking in the successive frames is achieved by motion projection and an inter-frame segmentation process. The main objectives of the tracking process are to avoid losing foreground regions and to avoid including false background regions. It contains following steps (**figure 3**).

First, segmented regions from the previous frame,

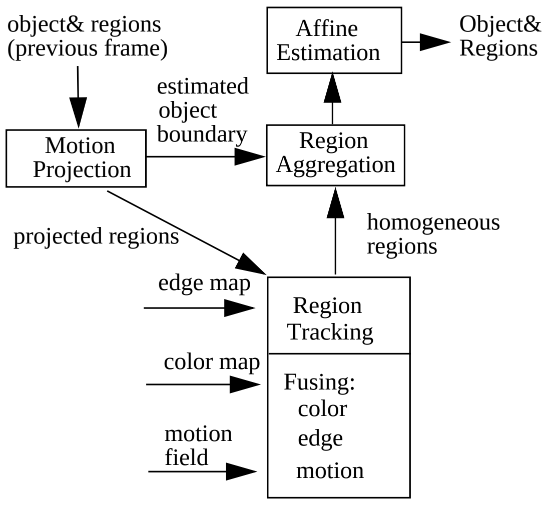

Figure 3. Automatic semantic object tracking

including both foreground and background, are projected onto the current frame (virtually) using their individual affine motion models. Projected regions keep their labels and original classifications. For video shots with static or homogeneous background (i.e. only one moving object), users can choose not to project background regions to save time.

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 based 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 generated automatically when a large quantization error is encountered.

In the tracking module (i.e. inter-frame segmentation), regions are classified into *foreground*, *background* and *new* regions. *Foreground* or *background* regions tracked from the previous frame are allowed to be merged with regions of the same class, but merging across different classes 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 merging result inherits its label and classification from the tracked region. In motion segmentation, split regions remain in their original classes. After this inter-frame tracking process, we obtain a list of regions temporarily tagged as either *foreground*, *background*, or *new*. They are then passed to an iterative region aggregation process.

The region aggregation module takes 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 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 error which may cause the loss of foreground regions around object boundary, the mask is further dilated with the tracking buffer size, which is specified by users at the beginning of the tracking.

The region aggregation module implements a region grouping and boundary alignment algorithm based on the estimated object boundary as well as the edge and motion features of the region. *Background* regions are first excluded from the semantic object. For every *foreground* or *new* region, compute intersection ratio of the region with the object mask. Then if:

<u>1) the region is *foreground*</u>

If it is covered by the object mask by more than 80%, it belongs to the semantic object. Otherwise, the region is intersected with the object mask and split :

a) split regions inside the object mask are kept as *foreground*

b) split regions outside the object mask are tagged as *new*

<u>2) the region is *new*</u>

If it is covered by the object mask by less than 30%, keep it as new; Else if the region is covered by the object mask by more than 80%, classify it as *foreground*. Otherwise:

a) Compute numbers of edge pixels (using the edge map) between this region and the current *background* and *foreground* regions. Compute differences between the mean motion vector of this region with those of its neighbouring regions and find the neighbour with the most similar motion.

b) If the region is separated from *background* regions by more edge pixels than *foreground* regions (or if this region is not connected to any *background* regions) and its closest motion neighbour is a *foreground* region, intersect it with the object mask and split :

- split regions inside the object mask are classified as *foreground*
- split regions outside the object mask are tagged as *new*

c) Otherwise, keep the region as *new*.

Compared with the aggregation process in section 3, a relatively lower ratio (80%) is used to include a foreground or new region. This is to handle motion projection errors. As it is possible to have multiple layers

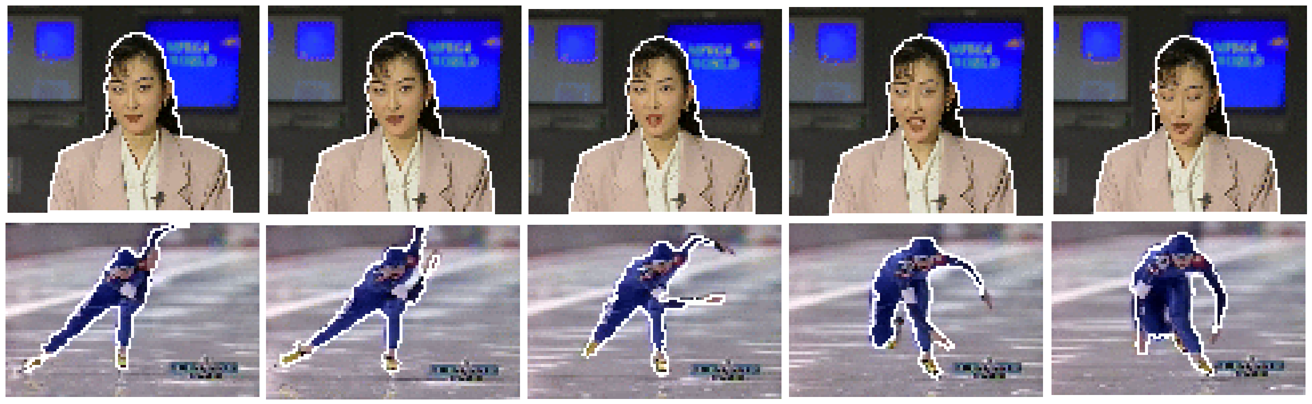

Figure 3. Semantic object tracking results of two sequences (from left to right, frame# 1, 10, 20, 30, 40)

of new regions emerge between the foreground and the background, the above aggregation and boundary alignment process is iterated multiple times. We have found this step useful in correcting errors caused by rapid motions. At the end of the last iteration, all remaining new regions are classified into background regions.

Finally, affine models of all regions, including both foreground and background, are estimated. As described before, these affine models are used to project regions onto the future frame in the motion projection module.

5. Experimental Results

Segmentation and tracking results of semantic objects (human) for two test sequences (CIF size) are shown at five frames (frame 1, 10, 20, 30 and 40) in **figure 3**. Tracked object boundaries are highlighted with white pixels. The first sequence consists of simple motion while the second sequence includes a rapid object with multiple distinctive moving regions.

In the objective evaluation, we manually extracted objects for the two sequences (with the help of the snake algorithm), and then computed the average numbers of missing and false pixels over the 40 frames. The average numbers of missing and false pixels and the average size of objects are show in **table 1**. Considering inherent errors caused by boundary smoothness (especially for the second MPEG-1 sequences with fast motion) as well as by manual segmentation, the tracking results are very good. This can be also confirmed by the satisfactory subjective evaluation as shown in **figure 3**.

	Missed	False	Object Size
seq1	241	126	37224
seq2	351	447	13023

Table 1. Average missing and false pixels

6. Conclusions

Semantic object segmentation is important for content based video representation. In this paper we presented an object segmentation system, AMOS, based on an innovative method integrating low-level region tracking and high level object segmentation. Fusion of color, edge, and motion features is proven very effective for region-level segmentation, while the iterative boundary alignment process is effective in maintaining object boundary accuracy over time. Our experiments have shown very good results. On-going study includes complexity analysis and tracking of multiple objects.

References

[1] 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, Seattle, WA, Nov. 1997.

[2] N.Ueda and K. Mase, "Tracking moving contours using energy minimizing elastic contour models", Computer Vision-ECCV'92, Vol 588, pp453-457, Springer-Verlag, 1992.

[3] C. Gu and M.-G. Lee "Semantic Video Object Segmentation and Tracking Using Mathematical Morphology and Perspective Motion Model", ICIP'97, October 26-29, 1997 Santa Barbara, CA.

[4] E. Saber, A.M. Takalp, & G. Bozdagi, "Fusion of Color and Edge Information for Improved Segmentation and Edge Linking," in IEEE ICASSP'96, Atlanta, GA, May 1996.

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

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

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

[8] M. Bierling, "Displacement Estimation by Hierarchical Block Matching", SPIE Vol 1001, Visual Communication & Image Processing, 1988.