SEMI-AUTOMATIC SEMANTIC VIDEO OBJECT EXTRACTION BY ACTIVE CONTOUR MODEL

Zhitao Lu and W. A. Pearlman

Electrical Computer and System Engineering Department
Rensselare Polytechnic Institute Troy NY 12180
luz2, pearlw@rpi.edu

ABSTRACT

An algorithm to extract the semantic video object in the first frame of a video sequence is proposed. The extracted closed object contour provides an accurate video object model for automatic object extraction in the following frames. Two polygons are input by the user to specify the area in which the object boundary is located. Based on this, edges which belong to the video object of interest are selected by a local object contour model. Then an active contour model (snake) is applied to refine the closed object contour.

1. INTRODUCTION

With the development of computer technology and the advent of network based visual communications, a huge amount of multimedia information needs to be stored, transmitted and accessed. Present and future multimedia applications require object based functionalities, for example, searching, manipulating and interacting with meaningful video objects. Source coding efficiency may also be improved from object based representation [1]. Since most digital images and video signals are in pixel format without semantic information, the extraction of semantic video objects from each frame of video sequences is a very important issue.

The techniques for video object extraction have been investigated in recent years. Most of them belong to two main categories: automatic and semi-automatic techniques.

Fully automatic extraction of semantic video object from a generic video sequence is a very difficult task. The main problem is that semantic objects are not necessarily homogeneous with respect to low level features, such as, color, intensity or motion. Since humans can easily identify an object in a video frame, semi-automatic techniques are more promising for generic video object extraction.

In semi-automatic techniques, information used to define the video object, for example, shape or some tuning parameters, are input by the user, then a final accurate video object extraction is done automatically [6].

In this work, we present a semi-automatic semantic video object extraction technique. We define a closed contour as the model of the semantic video object. The contour consists of the edges which distinguish the object and background in the frame. At the first frame, the user inputs two polygons which enclose the boundary of the object of interest. Then an active contour model – snake is used to determine the accurate location of the contour. The advantages of this scheme are:

- Since it is easy for humans to identify the scope of the video object of interest, no complex definition of "semantic" is needed.
- It is more generic than defining "semantic" by specifying the type of low level features, for examples, color, intensity and motion.
- A fairly accurate boundary of the object can be extracted in the first frame.

This scheme can also be applied to fully automatic inter-frame video object extraction. Instead of manually inputing an initial object model, the motion compensated video object model from the previous frame is used as the initial object model. The motion information can be defined as an energy term during the object contour finding process.

2. OVERVIEW OF THE PROPOSED ALGORITHM

In this work, we focus on automatic extraction of a video object in the first frame based on the initial model given by users. Two polygons are input by the user to specify the shape of the video object. The object contour is located between these two polygons. Each line of the outer polygon also approximately describes the direction of the object contour locally. The edge information of the whole image is detected by the Canny edge detector [5]. Then an edge selection module is used to select edges which belong to the object of interest. The snake technique is used to find the accurate locations of boundary points of the object. Finally a linking process forms the closed object contour using an edge based distance transform technique. The flow chart of the algorithm is shown in Fig. 1.

3. SEMANTIC VIDEO OBJECT EXTRACTION

3.1. Initial Video Object Model Input by User

In any semi-automatic technique, initial information needed from the user should be easy to input. Here we require two polygons to specify the shape of the object of interest, shown as in Fig. 3. The outer polygon must contain all pixels which belong to the video object, and the inner polygon must be totally within the object. Another important requirement for the outer polygon is that, each line of

it approximates the local direction of the object boundary. Objects with more complicated shapes will require initial outer polygon with more vertices. Fortunately, it is not difficult for the user to input these vertices. A better result is expected when users input better initial polygons. We have found that 10 to 30 vertices are needed for all sequences used in our experiments.

3.2. Selection of Edges Belonging to Object

Edges are detected by the Canny edge detector. The main problem is to determine which edges belong to the video object of interest. Since it is difficult to describe the global shape of a generic video object, a local contour model is used. We distinguish the edges of the video object by their location and local directions.

First, we eliminate the edges which are not in the area between the two polygons, because we know that they are definitely not part of the object boundary. For frames with homogeneous background near the video object, the object contour can be extracted easily by the snake technique presented in next section. For frames with texture background near the video object, the edges which belong to the background are removed so that the control points of the snake will not go to the unwanted edge points.

Here we use a simple local contour model to distinguish edges that belong to the video objects and the background. We assume that the difference between the directions of these two kinds of edges is large enough in most cases. We use a local contour model, which is a line segment with length d and direction θ, to check if the edge has similar direction with the model or not. The direction of the local contour model, θ which is the angle between the line segment and horizontal line, comes from the user input polygon. The process is described below:

1. For each edge point between the two polygons, find the nearest model point.
2. Generate the local contour model according to the direction at this object model point, θ.
3. Turn the local contour model by angle $n * \delta\theta$, where $-N < n < N$, $\delta\theta$ is the least angle to turn and N is the largest range in both direction can be turned. Check if the current edge segment match the direction of the local contour model or not.
4. If not, go back to last step. If yes, denote current edge point as object edge point.
5. If the current edge segment does not match the local contour model at any angle within $\pm N * \delta\theta$, denote it as background edge point and remove it.

For more complex object contour, a more complicated local contour model, such as a spline, may work well. This increases the complexity of the user input initial object model. Some segments of the initial model would be splines instead of straight lines.

3.3. Active Contour Modeling

After the edge information has been selected, an active contour model (snake) is used to find the accurate location of the object contour.

The Snake technique was originally proposed for edge detection problems [2]. It is widely used in object contour detection and computer vision applications. A snake is a set of ordered points, called control points. By moving these control points, snake can approach any shape at any location. The behavior of the snake is determined by an energy function. The energy function is defined so that it reaches minimum at desired location and shape. Usually the energy function E_{snake} consists of internal energy E_{int} and external energy E_{ext}.

$$E_{snake} = \alpha * E_{int} + \beta * E_{ext}$$

The internal energy is usually defined to determine the shape of the snake, and external energy is used to determine the location of the snake. α and β are the weight coefficients to balance these two terms. The key components of the snake model are:

- Define appropriate energy terms so that the energy at the desired location (in our case, the object edge) is at a minimum.
- Use an appropriate optimization algorithm to move the control points of the snake to the positions where the minimum energy is achieved.

In most object contour detection applications, the external energy is often defined as the magnitude of the edge or magnitude and direction of the edge [3]. The internal energy is often defined as smoothness or length of snake. One disadvantage of these definitions of external energy is that the external energy gives little information for the minimization process, since in the non-edge area, there are no differences in external energy terms. The movement of the snake only depends on the internal energy of the snake. Large search area, appropriate weights of internal and external energy and complicated searching algorithm are required.

In order to solve this problem. Two techniques are applied. First, the object edge selection module presented in the last section is used, so that most of the unwanted edges are removed. Second, we define the external energy as the distance to the nearest edge. The distance from the edge is calculated by the distance transform [4]. With this definition for external energy, the control points will go to the nearest edge which is our objective. During the optimization process, only the neighbors of the control points need to be checked, and the control points move to the neighbor with lowest value which means nearest to the object edge. This reduces the search area and makes the searching algorithm very simple and fast.

3.4. Create Closed Contour of Object

After we get the final snake, the closed contour of the video object is formed by linking the discrete snake control points. When we insert the points between two control points, we choose the points which belong to the edges first, otherwise the points with least Euclidean distance are chosen. We accomplish this by an edge based distance transform. The difference between it and traditional distance transform in [4] is that we assign a small value d_{edge} to the distance between two edge points, and a relative large value $d_{non-edge}$ to the distance between non-edge point pair and edge point to

Table 1: The number of vertices of the user input object model

Sequence	outside	inside
Claire	16	11
foreman	15	16

non-edge point pair. As shown in Fig. 2, distance between p0 and p1, p1 and p2 is d_{edge}, distance between p0 and p3 is $d_{non-edge}$, distance between p0 and p4 is $d_{non-edge} * \sqrt{2}$, distance between p0 and p5 is $d_{non_edge} * \sqrt{5}$. The linking process is as follows: suppose we link two control points p1 to p2.

- Set p1 as zero distance.
- Get the distance of every pixel to p1 by edge based distance transform.
- From p2, choose the neighbor point which has least distance to p1 as a point of the closed contour.
- Repeat last step until p1 is reached.

Figures 9 and 10 show the closed contour that is generated from the snake shown in figures 7 and 8 respectively.

4. EXPERIMENT RESULT

We have used several sequences to test the performance of the proposed algorithm. The objective is to extract the contour of the video object of interest based upon the user input initial object model. Figures 3 and 4 show the original image frames in the original Claire and Foreman sequences. The white dots are the user input initial models. The number of vertices we used for each sequence are shown in table 1. The polygons are interpolated linearly so that the distance between two consecutive control points is less than 8. The video object of interest in each sequence is the person.

The edges are detected by Canny edge detector. After that we remove the edges outside and inside the object according to the user input initial model, we also remove the edges which do not belong to the object of interest according to the local contour model. The results of the edges of video objects are shown in Figures 5 and 6. In the Foreman sequences, there are textures in the background near the video object, such as the edges on the wall. Most of these are removed by the local contour model. After the edge selection, the snake model is applied to find the location of object contour, the final snakes are shown as the white dots in Figures 7 and 8. The final closed object contour of each object is shown as in Figures 9 and 10.

5. CONCLUSION AND FUTURE WORK

A new algorithm to extract the semantic video object is proposed in this work. It generates an accurate video object model for automatic video object extraction. The local contour model is used to select the edges which belong to the video object of interest. The accurate closed object contour is extracted through the snake model. The use of a local contour model and newly defined energy function reduce the complexity of the snake optimization. The performance of proposed algorithm is demonstrated by the experiments on several widely used test sequences.

The application of this scheme on automatic inter frame object extraction is investigated. The motion information is integrated into the object edge selection process and the energy function of the snake model. Also the more complex local contour model, such as spline, is under consideration.

6. REFERENCES

[1] M. Kunt, A. Ikonomopoulos, M. Kocher, "Second Generation Image Coding Techniques", Proceedings of the IEEE, vol 73, No. 4 pp.549-675, April 1985

[2] M. Kass, A. Witkin, D. Terzopoulos, "Snakes:Active Contour Models", International Journal of Computer Vision, PP321-331, 1988

[3] K.F. Lai, "Deformable Contours: Modeling, Extraction, Detection and Classification", PhD thesis, Dept. of EE, Univ. of Wisc.1995

[4] G. Borgefors,"Distance Transformations in digital images" , Computer Vision, Graphics and Image Processing, vol. 34, pp.344-371 , 1986

[5] J. Canny, " A computational Approach to Edge Detection", IEEE Trans. PAMI, pp679-698, Nov. 1986

[6] C. Gu and M. C. Lee, "Semi-automatic Segmentation and Tracking of Semantic Objects", IEEE Trans. on Circuits and System for Video Technology, vol. 8, No. 5 pp572-584, sep. 1998

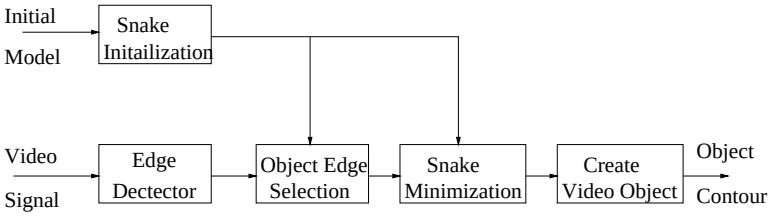

Figure 1: The Diagram of Proposed Semantic Video Object Algorithm

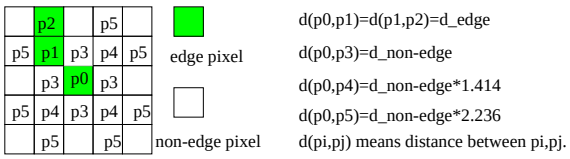

Figure 2: The Unit Distance of Proposed Edge Based Distance Transform

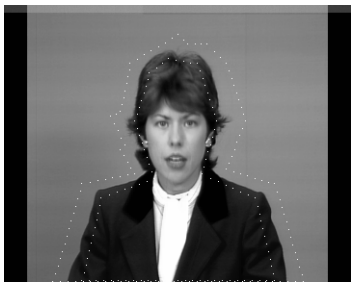

Figure 3: User Input Initial Object Model

Figure 4: User Input Initial Object Model

Figure 5: Edges of Video Object

Figure 6: Edges of Video Object

Figure 7: Final Snake Model

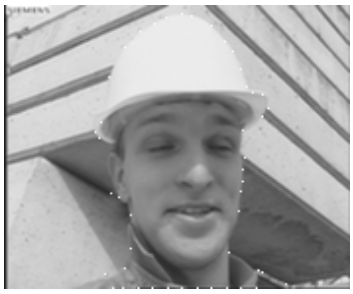

Figure 8: Final Snake Model

Figure 9: Final Object Model

Figure 10: Final Object Model