Hierarchical Video Object Segmentation

Junliang Xing, Haizhou Ai
Dept. of Computer Science and Technology, Tsinghua Univ.
Beijing 100084, China
xjl07@mails.tsinghua.edu.cn, ahz@mail.tsinghua.edu.cn

Shihong Lao
Development Center, OMRON Social Solutions Co., LTD.
Kyoto 619-0283, Japan
lao@ari.ncl.omron.co.jp

Abstract—In this paper, we propose a general video object segmentation framework which views object segmentation from a unified Bayesian perspective and optimizes the MAP formulated problem in a progressive manner. Based on object detection and tracking results, a three-level hierarchical video object segmentation approach is presented. At the first level, an offline learned segmentor is applied to each object tracking result of previous frame to current frame based on a discriminative feature points voting process. At the second level, the coarse segmentation is updated into an intermediate segmentation by a temporal model which propagates the fine segmentation of previous frame to current frame based on a discriminative feature points voting process. At the third level, the intermediate segmentation is refined by an iterative procedure which uses online collected color-and-shape information to get the final result. We apply the approach to pedestrian segmentation on many challenging datasets that demonstrates its effectiveness.

Index Terms—object detection, tracking, segmentation.

I. INTRODUCTION

Supposing objects (e.g. humans) have been detected and tracked through a video sequence which results in sequences of rectangular image windows containing the objects, video object segmentation aims at delineating the object out from its background in its corresponding window among different frames as illustrated in Fig. 1.

In the past decade, object detection and tracking have achieved significant advances [1][2] that attract attentions on further researches such as pose estimation and human re-identification, in which object segmentation is an important intermediate step. Previous methods deal with this problem including tracking coherent image regions [3], using a graph-cut like algorithm [4] and spatial-temporal grouping [5]. Without employing a concrete object model, these methods tend to be inefficient for specific object segmentation. In [6], top-down object model are used to guide segmentation in the key frames, followed by bottom-up propagation in other frames. Due to the difficulty of the problem itself, only rough object regions can be extracted. In this work, we formulate the video object segmentation as a general MAP problem and develop a progressive way to optimize it. Based on the object detection-and-tracking results, we present a fully automatic video object segmentation approach in which multiple cues, both offline and online, intra-frame and inter-frame information, are efficiently fused together to achieve accurate segmentation results.

The main contribution of this paper includes: 1) a general video object segmentation framework which hierarchically segments video objects from a coarse level to a fine level; 2) a unified Bayesian formulation of the object segmentation models and a progressive optimization strategy to solve the MAP formulated problem of the whole segmentation framework; 3) a set of well chosen-and-designed object segmentation models which exploit different kinds of useful information to help improve the segmentation result.

II. PROBLEM FORMULATION

A large number of previous methods try to solve the segmentation problem from a certain perspective, e.g., the Watershed [7] algorithm views the image as topographic reliefs and tries to segment the image by finding its watersheds, the GraphCut [8] and GrabCut [9] algorithms view the image as a graph and try to segment it by finding a minimal cut. However, no matter what forms those methods present, they in general can be viewed as a pixel labeling process. Given an image of \( N \) pixels indexed by \( X \), \( S = \{ S_x | x \in X \} \) is defined as a set of corresponding pixel label states, and \( B = \{1, \ldots, M \} \) as a set of all possible label states. To segment the image into \( M \) visually meaningful groups, each pixel is assigned one of the prescribed labels so that \( \forall x \in X \), \( S_x \in B \). To accomplish this pixel labeling process, observations of the image are explored to guide the process. Denote \( O = \{O_x | x \in X \} \) as the set of observations collected from the image. Now, based on the observations, Bayes’s rule can be used to estimate the likelihood of the image labels by

\[
P(S|O) \propto P(O|S)P(S). \tag{1}
\]

Then object segmentation can be formulated as a maximum a posteriori (MAP) estimation of \( P(S|O) \), i.e.,

\[
S^* = \arg \max_S P(S|O). \tag{2}
\]

Most previous work [7][8][9][10] try to solve the MAP problem by directly optimizing one energy function. To improve the segmentation accuracy, they may use multiple observations and try to fuse them parallelly into the energy function. We found that, through extensive experiments, it is very hard for
A. Coarse Segmentation with Offline Learned Segmentor

The input of the first level segmentor is the image patch within the bounding box generated by the object detector and tracker. The label state of each corresponding pixel within the bounding boxes has similar probability distribution, for example, the pixels which lie in the center of the bounding box are mostly likely to be the Foreground while those lie around the box boundary are mostly likely to be the Background. In order to capture these statistic characteristics of the pixel state, we turn to using an offline segmentation model.

We base on the boosting method in [12] to train the offline segmentation model and make two main improvements. On the one hand, we adopt the EHoG [2] feature instead of the Edgelet feature in [12] to train the segmentation. This is because the HoG feature has demonstrated its great successes in capturing gradient information of objects that is useful to object segmentation. What’s more, since our object detector is trained based on EHoG feature, the calculation of the EHoG feature can be shared by the segmentation model. On the other hand, one basic assumption in [12] is that good features for detection are also good for segmentation which is not always true. Hence in our boosting procedure, we select the best features that minimize only the segmentation training error which results a more effective segmentation model. For other details on training the segmentation model, please refer to [12].

Suppose the final strong classifier \(H(x; u)\) consists \(T\) weak classifiers and take zero as the default threshold, i.e. the pixel \(u\) within image patch \(x\) is classified as foreground, iff

\[
H(x; u) = \sum_{i=0}^{T} h_i(x; u) > 0.
\]

The value of \(H(x; u)\) also reflects the segmentation confidence of pixel \(u\).

The strong classifiers of all the pixels within training sample form the offline segmentor. Now given a new image patch generated by the detector, we firstly resize it to the training sample size and then apply offline segmentor on the resized image to get the segmentation result along with the confidence. The segmentation result and the confidence in training sample size are then scaled back to original image patch size. The segmentation result of the first level \(P(S_1|O_1)\) is generated from the segmentation confidence by mapping it to probability...
using a sigmoid function
\[ P(S_1|O_1) = \frac{\exp(2H(x; u))}{1 + \exp(2H(x; u))}. \] (7)

**B. Temporal Propagation with Discriminative Points Voting**

Objects in video sequences undergo strong inter-frame correlation which provides very useful information for segmenting objects. The second level of our system is a temporal model that uses this inter-frame correlation to update the segmentation result of the first level. By using the object tracker, we have got the correspondence of the object region bounding boxes in each frame. Based on the corresponding bounding boxes, the temporal model captures the inter-frame correlation to update the segmentation result of the first level.

In order to propagate the segmentation result of previous frame to current frame, we need to find the correspondence between the bounding boxes in the neighboring frames in pixel level. We make this correspondence through a discriminative feature points voting process based on the KLT feature point detection and tracking method [13]. First, we detect and track KLT feature points within the bounding box between two consecutive frames. And then we find the correspondence by voting using these corresponded points. The voting rule is illustrated in Fig. 3. For a feature point, the segmentation result is directly shifted from previous frame to current frame since it has the *casting vote*. For a point within the neighborhood (9 × 9 in our experiment) of some feature points, its result is voted by these feature points. And for a point out of the neighborhoods of all feature points, its result is determined by shift and scale transformation of previous result (*abstention vote*).

After propagating the fine result of previous frame to current frame, the result is then used to update the segmentation result of level one. The updating strategy can either be addition rule (add two probabilities to get a mean probability) or multiplication rule (multiply two probabilities to get a mean probability). In our experiments, the addition rule is adopted.

**C. Online Refinement with Color and Shape Model**

Since different objects in video sequence have their specific properties, such as appearances and gestures, the whole segmentation framework needs to suit for these specialties of the objects. The third level of our system is to online collect the object-specific information to help improve the segmentation result. Two kinds of object-specific information we used are the object color and shape information. For color information, a Gaussian Mixture Model (GMM) is used to model the color distribution of the foreground and background respectively; for shape information, the Active Contour Model (Snake model) [10] is used to optimize the object shape.

1) **Color Model:** We use the GMM to model the color distribution of the foreground and background pixels as in GrabCut [9]. A GMM uses a linear superposition of Gaussian distribution to represent a complex distribution which can be written in the form

\[ p(z) = \sum_{k=1}^{K} \pi_k G(z|\mu_k, \Sigma_k). \] (8)

where \( K \) is the number of components. From the segmentation result of the first two levels, each pixel within the bounding box gets a probability reflecting its state: the Foreground (\( P(S_2|O_2) \rightarrow 1 \)) or the Background (\( P(S_2|O_2) \rightarrow 0 \)). Based on the probability, we build two GMMs, one for the foreground pixel set and one for the background pixel set. The parameters of the GMM are learned from the pixel color value in RGB color space using the EM algorithm.

After the GMMs for foreground and background are learned, all the pixels within the bounding box are reclassified using the learned color models as in [9].

2) **Shape Model:** We use a Shape Model to further accurate the object contour. The shape model we adopt is the Snake Model [10]. A Snake is an energy minimizing spline guided by a set of forces which could include internal forces, external forces and other constraint forces applied by user to pull it towards features such as lines and edges. In our implementation, we concentrate on the object shape information, so the energy function is designed to reflect the shape forces. The energy function in our system includes three parts: the continuity energy (calculated from the first derivative of the snake), the curvature energy (calculated from the second derivative of the snake) and the image energy (calculated from the image data). Denote \( \nu(s) = [x(s), y(s)] \) as the parametric position of a snake, the energy function of the snake in our system can be represented as

\[ E_s = \int_0^1 [\omega_1 E_{con}(\nu(s)) + \omega_2 E_{cur}(\nu(s)) + \omega_3 E_{img}(\nu(s))] ds, \] (9)

where \( E_{con} \), \( E_{cur} \) and \( E_{img} \) are the continuity, curvature and image energy respectively, \( \omega_i \) (\( i = 1, 2, 3 \)) is the weight reflecting the importance of corresponded type of energy.

The initial “snake” of the shape model was generated from the result of the color model by finding the maximal contour of the binary image of the segmentation mask. By applying the snake algorithm, we get the converged snake. Based on this snake, all the pixels are reclassified.

The third level of our system is an iterative procedure: the color model and shape model are iteratively used to refine the segmentation result. When a certain iterative stop criteria is met (the segmentation result does not change distinctly or exceeds the maximum number of iteration), the iterative procedure terminates to get the final result.

**IV. Experiments**

We apply the proposed framework to pedestrian class and test it on two public dataset [14][15] and some video sequences...
we collected. We first conduct experiments to analyze the segmentation accuracy of different level segmentors in the system. And then we compare our method with some representative methods that are closely related to ours.

A. Analysis of Different Level Segmentors

To train the offline segmentor, we collect 650 front/rear view pedestrian samples with a normalized size $24 \times 58$ and manually label the foreground/background ground truth of these samples. Some samples are showed in Fig. 4. The training process stops at the segmentation rate 90.8% in the training set. For the color model, we use 5 Gaussian components both for the background and foreground region ($K = 5$). And for the shape model, we set the importance of continuity, curvature and image energy equally ($w_1 = w_2 = w_3$). The maximal iteration number of Level Three is set as 5 and the average iteration number is only about 2 in the experiments.

In Fig. 5, we give the segmentation results of each level on some pedestrians in the video sequences from which we can see that our video segmentation method hierarchically segments the pedestrian out of the background from a coarse level to a fine level. The offline model only gives a rough estimation of the human body region and the segmentation result it gives is far from accurate especially in the boundary region. The temporal model further validates the segmentation results by applying the temporal constraint to point out the unsure region. The color model then classifies the unsure region using the color information and the shape model refines the human body shape to remove some noises, such as holes in the human body and isolated patches. From Fig. 5 we can also see that although our offline segmentor is trained using the front/rear view pedestrians our whole framework can well segment the pedestrians in profile views.

![Fig. 4. Samples for front/rear view pedestrians. Top row: pedestrian samples; bottom row: labeled segmentation ground truth.](image1)

![Fig. 5. Comparisons on different level segmentors: 1st col.: input pedestrians; 2nd and 3rd col.: segmentation confidences and results of the offline model; 4th and 5th col.: segmentation confidence and results of the temporal model; 6th and 7th col.: color model and corresponding results; 8th and 9th col.: shape model and corresponding results.](image2)

To give a quantitative analysis of the segmentation accuracy of each level segmentor, we save the intermediate result of each level segmentor when segmenting a pedestrian in 20 consecutive frames and calculate the Accuracy value compared with labeled ground truth. For the third level of our algorithm, we save the intermediate results after the first time using the color model, the first time using the shape model and the final result after the iteration process terminates. Fig. 6 gives the Accuracy value of each level segmentor in each frame from which we can further found that each level segmentor can progressively increase the segmentation accuracy.

![Fig. 6. Segmentation accuracy of each level for a pedestrian. It can be seen that the model in different level progressively improves the accuracy.](image3)

B. Comparison with Other Methods

We further compare our approach with the well-know background modeling technique and the GrabCut algorithm. The background subtraction algorithm in [16] has been proved to be very effective for segmenting moving objects in video which shares similar aim with our approach. And the GrabCut algorithm represents one typical kind of object segmentation methods which also inspires the design of the color model in our approach to a certain extent. So it would be meaningful and convenient to use them as the baseline methods.

For the background modeling method, we use the implementation in OpenCV [17] which has been widely used in the computer vision research field. In order to make the initial background model more accurate, we manually insert several background frames of the scene to help the background modeling method to build the background model, what’s more, the algorithm parameters are carefully tuned according to the video data. And for the GrabCut algorithm, we implement it as described in [9] with the pedestrian detection-and-tracking results as the initial input bounding box to guide its iteration process. The test video is the sequence EnterExitCrossing-Paths1cor.mpg and we manually label its segmentation ground truth every 5 frames and get 76 labeled frame in total. The three evaluation metrics, Accuracy, Precision and Recall are calculated at all labeled frames and the mean value of them

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Accuracy</th>
<th>Precision</th>
<th>Recall</th>
<th>Speed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Background Subtraction[16]</td>
<td>0.873</td>
<td>0.917</td>
<td>0.879</td>
<td>21fps</td>
</tr>
<tr>
<td>GrabCut[9]</td>
<td>0.892</td>
<td>0.926</td>
<td>0.902</td>
<td>7fps</td>
</tr>
<tr>
<td>Proposed Method</td>
<td>0.928</td>
<td>0.950</td>
<td>0.936</td>
<td>11fps</td>
</tr>
</tbody>
</table>

TABLE I

COMPARISON OF SEGMENTATION PERFORMANCE
are given in Table I. To compare the efficiency of the three methods, the average running speeds of them on this sequence are also given in the last column of Table I. In Fig. 7, segmentation results of the three algorithms on some typical frames are showed to give a visual comparison.

From Table I and Fig. 7, it’s not hard to find that, even with the carefully chosen background model and well-tuned parameters, the background modeling method still cannot avoid the noises and wholes within the human body. Our method can well solve these problems in most of the case. And compared with the GrabCut algorithm, our hieratical optimization process can obtain a relatively more accurate object shape boundary. Looking at the Speed column in Table I, while the test video was coded with the standard frame rate 25fps, our algorithm runs at about 11 fps which makes it possible for real-time applications after some optimization. The background subtraction method is quite efficient which runs at about 21 fps. And the GrabCut algorithm runs at about 7fps. In Fig. 8, we give more segmentation results of our method on the CAVAIR sequence TwoEnterShop2cor.mpg and other two video sequences which are much more challenging. The segmentation results on these sequences further prove the effectiveness of our method.

V. CONCLUSIONS

In this paper, we propose a novel video object segmentation framework which efficiently fuses multiple cues to hieratically segment the object out of background from a coarse level to a fine level. The whole framework is general: 1) the level segmentor has a unified representation and can be easily designed from lots of different ways; 2) we can freely modify processing procedure of the whole framework either by adding new levels, removing exiting levels or replacing existing levels with new levels to adjust the performance of the whole framework, such as accuracy, efficiency and plasticity; 3) The proposed approach can also be applied on other object classes, such as cars, faces, etc. Current system assumes that the object detector and tracker have provided the location and correspondence of the pedestrians which may not be met at all the time since the detector and tracker may fail at some frames, we will improve this by using the object segmentation result to help the detector and tracker locate and find the right target more accurately which could give birth to an integral object detection, tracking and segmentation system.

ACKNOWLEDGMENT

Prof. Haizhou Ai is partially supported by National Science Foundation of China under grant No.61075026. Mr. Shihong Lao is partially supported by ”R&D Program for Implementation of Anti-Crime and Anti-Terrorism Technologies for a Safe and Secure Society”, Special Coordination Fund for Promoting Science and Technology of MEXT, the Japanese Government.

REFERENCES