SEMI-AUTOMATIC OBJECT TRACKING IN VIDEO SEQUENCES BY EXTENSION OF THE MRSST ALGORITHM

Marko Esche, Mustafa Karaman, Thomas Sikora

Communication Systems Group, Technische Universität Berlin
Sekr. EN1, Einsteinufer 17, D-10587 Berlin, GERMANY
{esche, karaman, sikora}@nue.tu-berlin.de

ABSTRACT

The objective of this work is to investigate a new approach for object segmentation in videos. While some amount of user interaction is still necessary for most algorithms in this field, these can be reduced making use of certain properties of graph-based image segmentation algorithms. Based on one of these algorithms a framework is proposed, that tracks individual foreground objects through arbitrary video sequences and partly automates the necessary corrections required from the user. Experimental results suggest, that the proposed algorithm performs well on both low- and high-resolution video sequences and can even cope with motion blur.

1. INTRODUCTION

Both fast segmentation of real world objects in still images and tracking of theses objects throughout a video sequence have many applications in video processing and editing [1]. While a lot of progress has been made concerning the extraction of binary object masks for single frames, automatic generation of such masks for an entire video remains mainly unsolved. Since dynamic foreground objects both need to be correctly identified and can also undergo sudden changes in shape and color, most video segmentation tools rely on user interaction in order to produce accurate results. In this paper a new framework for the tracking of foreground objects in video sequences based on the *Modified Recursive Shortest Spanning Tree Algorithm* (MRSST) is presented. It incorporates the interactive Object Contour Extraction method proposed by Adamek and O'Connor in [2]. Additionally, a new approach to identify corresponding image regions in consecutive frames is integrated using a *binary partition tree* (BPT) for each frame. The remainder of the paper is structured as follows. Section 2 briefly revisits the segmentation of color images using the MRSST and its applicability to the task of object extraction. Both a general outline of the proposed algorithm and detailed descriptions of two of its main components are given in Section 3. Experimental results and objective evaluation measures are provided in section 4. Section 5 concludes the paper with a short discussion and future directions.

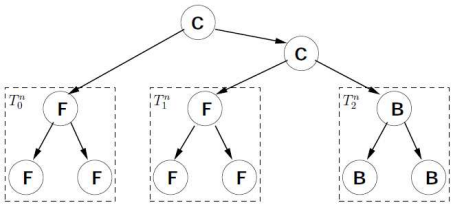

Fig. 1. After the processing of the scribbles added by the user, local homogeneous subtrees of the BPT are formed (marked with dashed rectangles), denoted here by T_0^n, T_1^n and T_2^n.

2. OBJECT EXTRACTION AND THE MRSST

Among the various image segmentation approaches graph-based algorithms, such as the one presented by Cooray et al. in [3], have lately received significant attention. This is partly due to their ability to represent image segments of various sizes as nodes in a BPT. Among them is the MRSST presented in [4], which introduces so-called syntactical features in order to associate image regions with features such as spatial compactness and shape regularity. Initially the MRSST treats every pixel of an image as a node in a graph that is connected via weighted edges to its four direct neighbors. The segmentation of the image is achieved by iteratively merging those nodes that are connected by the least-cost link, which also enforces the recalculation of associated graph edges and their weights. During this process the BPT is created by storing the order in which nodes are merged and establishing a father-child relationship between the newly created node and the two merged ones. The output of the MRSST in this case is a BPT, whose root node contains the entire image while the leafs are the individual pixels of the image. In [2] Adamek and O'Connor proposed a method that allows to quickly extract arbitrary foreground shapes from a BPT by letting the user draw so-called scribbles on the image. Both foreground (**F**) and background (**B**) scribbles are passed to the leafs of the BPT in the form of competing labels. These are then iteratively propagated up the BPT. When the algorithm tries to assign both labels to the same node, that node is marked as

a conflict node (**C**) and all its parent nodes are marked with the conflict label as well. This results in the formation of homogeneously labeled local subtrees that either belong exclusively to foreground or background. An example of a labeled BPT is given in figure 1, where each subtree represents an individual spatial region of its own. By adding further scribbles to the image the resulting foreground mask can be refined further.

3. PROPOSED ALGORITHM

The new object tracking algorithm, that is proposed in this paper, consists of a three-stage approach as outlined by figure 2 for a video sequence of N frames. Initially a BPT is created for the first frame of the sequence which is then labeled by the user in order to correctly identify the foreground object to be tracked, which results in the extraction of a shape S^n. The labeling is done using the method described in [2]. For every consecutive frame a BPT is also created. An initial estimate of the shape S^{n+1} of the tracked object in the next frame is determined by matching local subtrees of the BPT among neighboring frames. More details on this technique are provided in section 3.1. Having obtained this estimate, the object contour C^n is transferred into the next frame and used further to correct the object contour C^{n+1} of the predicted object shape S^{n+1}. See section 3.2 for details. Afterwards the predicted object shape is presented to the user, who can now add further labels to refine the object shape. An approach with a similar workflow, which however does not make use of a graphical image representation and therefore relies solely on the object contour has been proposed in [5].

3.1. Identification of Corresponding Subtrees

Before describing the actual algorithm for transferring the current object shape S^n into the next frame, a number of definitions have to be made. The areas in pixels occupied by shapes S^n and S^{n+1} are denoted as a_S^n and a_S^{n+1} respectively. The area associated with a subtree in the current frame is consequently denoted by $a_{T_i}^n$. A similar notation is used to describe the center pixel of a shape S in the current frame $\vec{c}_S^{\,n}$ or the center pixel of a subtree in the next frame $\vec{c}_{T_i}^{\,n}$. In order to evaluate the suitability of a subtree T_j^{n+1} for inclusion in the new object shape S^{n+1}, the previously determined object shape is transferred into the next frame using the motion vector given in equation 1.

$$\vec{v} = (\vec{c}_S^{\,n+1} \cdot a_S^{n+1} + \vec{c}_S^{\,n} \cdot (a_S^n - a_S^{n+1}))/a_S^n - \vec{c}_S^{\,n+1} \quad (1)$$

The choice of this motion vector ensures that the predicted object shape S^{n+1} is initially placed at the same location as the previous object shape, see figure 3 for a simple example. Once more local subtrees have been assigned to the new object shape, the location of the predicted object shape is adapted to fit to the new object location. Based on the

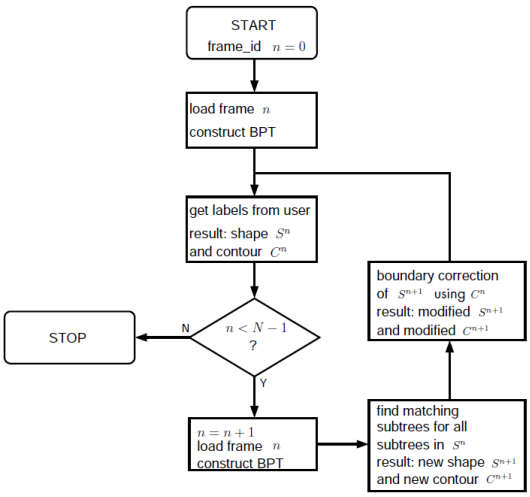

Fig. 2. General structure of the algorithm.

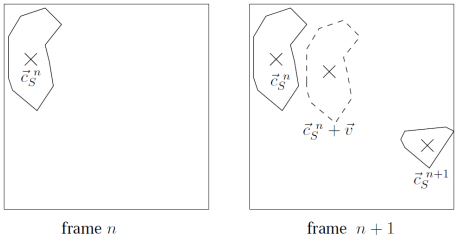

Fig. 3. *left*: object shape in the previous frame, *right*: partially constructed object shape for the current frame with center $\vec{c}_S^{\,n+1}$ and the moved reference shape with center $\vec{c}_S^{\,n} + \vec{v}$.

transferred version of S^n it is now possible to compute a relative overlapp $c_{ov}(T_j^{n+1})$ between every subtree in the current frame and the entire object shape in the previous frame. Additionally, a color cost $c_{co}(T_j^{n+1})$ is assigned to every local subtree in the new frame. It is computed as the smallest euclidean distance in *CIEL*u*v** color space between the average color of T_j^{n+1} and any subtree T_i^n in the previous frame within a search radius $r(T_j^{n+1})$. Initially the search radius is set to $\frac{1}{5}\sqrt{a_{T_j}^{n+1}}$. A greedy algorithm that tests every subtree in the new frame for inclusion in the new object shape is now developed: During every iteration the subtree with the smallest inclusion cost as given in equation 2 is chosen and included in the new object shape.

$$c_{inc}(T_j^{n+1}) = \begin{cases} \infty, & \text{if } \begin{array}{l} c_{co}(T_j^{n+1}) > c_{co}^{max} \\ \vee\, c_{ov}(T_j^{n+1}) < c_{ov}^{min} \end{array} \\ \frac{c_{co}(T_j^{n+1})}{c_{co}^{max}} + 1 - c_{ov}(T_j^{n+1}), & \text{else} \end{cases} \quad (2)$$

Additionally, the subtree T_k^n in the previous frame with the smallest color distance to the included subtree T_l^{n+1} is removed from the previous object shape to ensure that the color distributions of S^n and S^{n+1} remain identical. The thresh-

olds c_{co}^{max}, c_{ov}^{max} and r^{max} used in equation 2 are needed to control the behaviour of the inclusion algorithm. When a subtree receives an infinite inclusion cost, then it is split into its respective shildren which are then examined during the next iteration, thus making use of the BPT's structure to quickly find a best match. Should it not be possible to include any candidate subtree during the current iteration, because no subtree received a finite inclusion cost, then the three thresholds are adapted as given by the the following stage description:

1. Inclusion of subtrees T_j^{n+1} with at least 50% overlapp, small color difference and little individual segment motion $r(T_j^{n+1})$
2. Inclusion of subtrees T_j^{n+1} with at least 50% overlapp, bigger color difference and more segment motion
3. General inclusion of all segments T_j^{n+1}, that have a color distribution similar to T_i^n and lie somewhere within the entire region of interest

3.2. Boundary Correction

Since object deformations usually manifest themselves as a modification of the object boundary, it is necessary to adopt the object contour C^n to the new object shape S^{n+1}. The strategy proposed here is to first move the previous object shape into the current frame. Afterwards every pixel along the original object contour is examined. For a square patch of size $l \times l$, $l = \frac{1}{4}\sqrt{a_S^n}$, around the pixel location in the previous frame the contrast measure given in equation 3 is computed, where L_F, u_F, v_F and L_B, u_B, v_F are the three components of the average colors of foreground and background inside the patch, respectively.

$$con = \frac{(L_F - L_B)^2 + (u_F - u_B)^2 + (v_F - v_B)^2}{3 \cdot 255 \cdot 255} \quad (3)$$

Should the contrast between background and foreground be higher than 0.1% then a local scribble model as shown in figure 4 is built. The scribble model is basically a distribution

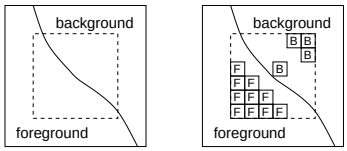

Fig. 4. Original object contour (*left*) and adapted automatic scribbles (*right*). **B** and **F** indicate background and foreground labels, respectively.

of labels over the square patch mentioned above, that is assumed to extract the correct object shape, if these automatic scribbles were added to an unlabeled BPT. The actual distribution is chosen with respect to possible contour deformations and only the one background pixel closest to the center

Fig. 5. *left*: initially predicted object shape (outlined im red), that was constructed using the subtree matching approach; *center*: automatically generated scribbles (green for foreground, red for background); *right*: corrected object shape, after the application of the scribbles.

of the patch is expected to remain a background pixel. A best match for the $l \times l$ patch is now found by performing a fullsize blocksearch around the initial contour location. Once the best match has been identified, the previously obtained automatic scribbles are added to the BPT of the new frame around the corrected location of the considered contour pixel. An example for such a boundary correction step is provided in figure 5. Of particular interest in the displayed frame is the flag that occupies the lower right part of the image and is correctly identified as part of the background despite having the same color as the foreground object.

4. EVALUATION

The propsed algorithm has been implemented in *C++* and tested on the CIF test sequences *House*, *Highway* and *Group*. It was also tested on one sequence each from the following movies in DVD resolution: *Harry Potter and the Sorcerer's Stone*, *Planet Earth* (BBC Documentary) and *Star Wars - Episode IV*. Some of these also include global camera motion or moving background objects. Keyframes and extracted foreground objects are shown in figure 6. In order to objectively evaluate the algorithm, all automatically generated object masks were compared with manually segmented groundtruth masks. For each frame precision (p), recall (r) and f-measure (f) were computed. In addition, the number of user interactions (u) and the number of manually labeled pixels (l) per frame were recorded. In this context, a user interaction is defined as a single connected scribble in either foreground or background color. The per-frame values for p, r, f, u and l for the *House* sequence are given in figure 7. Due to the foreground object (a person walking from right to left) entering the scene during the first frames of the sequence satisfactory results are only achieved from frame five onwards, when the person is visible in its entirety. The average measures for all sequences are given in table 1. In addition, the percentage of frames, for which less than 4 or less than 6 scribbles were needed (rel_4 and rel_6 respectively) are provided. Of

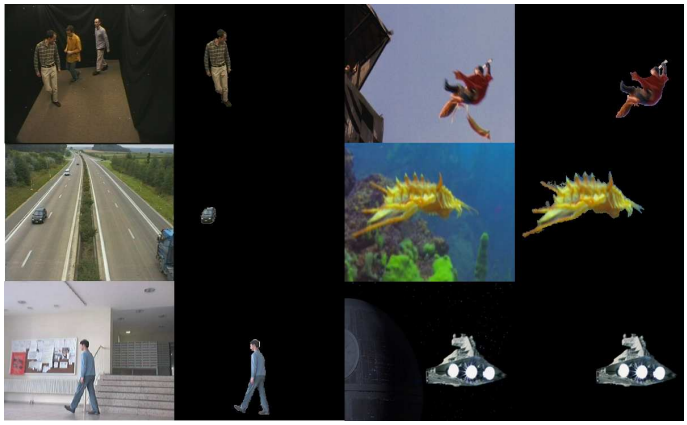

Fig. 6. Sample frames and extracted foreground objects for all tested sequences. For the high -resolution sequences displayed in the right columns only a magnified region of interest is shown.

Sequence	p	r	f	rel_4	rel_6
Group	93.0%	89.3%	91.0%	70.7%	92.7%
Highway	92.7%	89.2%	90.1%	88.2%	94.1%
House	93.3%	86.4%	88.7%	70.3%	94.6%
Harry Potter	87.8%	94.2%	90.8%	54.2%	80.6
Planet Earth	92.3%	92.0%	92.1%	67.5%	95.8%
Star Wars	97.6%	98.3	97.9	100%	100%

Table 1. Average precision, recall and f-measure per sequence. rel_4 and rel_6 indicate the percentage of frames for which < 4 respectively < 6 scribbles where necessary.

particular interest is the *Harry Potter* sequence throughout which motion blur is frequently present. Nevertheless, the dual approach performs comparably as well as for the other sequences. A comparison with similar interactive tracking approaches has not been conducted yet, since, to the knwoledge of the authors, a general framework for objectively measuring the amount of user interaction still needs to be established.

5. SUMMARY

The main objective of this work was to develop a new interactive object tracking approach based on the MRSST. In this paper a two stage algorithm has been described, that tracks individual foreground objects in video sequences by matching local subtrees of the BPT among neighboring frames and by generating an automatic set of foreground and background labels for each consecutive frame. It has been shown, that the algorithm performs comparatively well for both high- and low-resolution videos. In addition, no restrictions have been placed on movement, shape or variability of the tracked foreground object, which makes the algorithm applicable for arbitrary videos and a wide range of real-world objects. Future

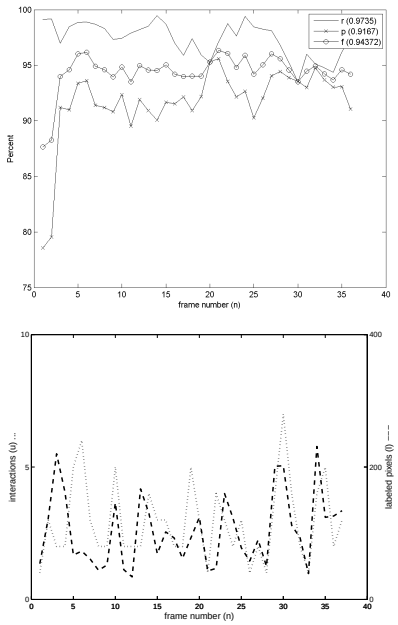

Fig. 7. *top*: precision, recall and f-measure values for the automatically generated object masks for the *House* sequence; *bottom*: user interactions needed to correct these masks

work will include the incorporation of a background subtraction approach in order to reduce the amount of misclassification done during the first stage of the algorithm.

6. REFERENCES

[1] A. Alatan, L. Onural, M. Wollborn, R. Mech, E. Tuncel, and T. Sikora, "Image sequence analysis for emerging interactive multimedia services - the european cost 211 framework," *IEEE Transations on Circuits and Systems for Video Technology*, vol. 8, pp. 802–813, 1998.

[2] Tomasz Adamek and Noel E. O'Connor, "Interactive object contour extraction for shape modeling," in *1st International Workshop on Shapes and Semantics*, 2006.

[3] S. Cooray, Noel O'Connor, Sean Marlow, Noel Murphy, and Thomas Curran, "Semi-automatic video object segmentation using recursive shortest spanning tree and binary partition tree," in *WIAMIS*, 2001.

[4] Tomasz Adamek, Noel E. O'Connor, and Noel Murphy, "Region-based segmentation of images using syntactic visual features," in *WIAMIS*, 2005.

[5] Xue Bai, Jue Wang, David Simons, and Guillermo Sapiro, "Video snapcut: robust video object cutout using localized classifiers," in *SIGGRAPH '09: ACM SIGGRAPH 2009 papers*, 2009.