Foreground segmentation via segments tracking

Bogdan Kwolek

Rzeszów University of Technology, 35-959 Rzeszów, Poland
bkwolek@prz.edu.pl

Abstract. In this paper we propose a video segmentation algorithm that in the final delineation of the object employs the graph-cut. A partitioning of the image based on pairwise region comparison is done at the beginning of each frame. A set of keypoints is tracked over time via optical flow to extract regions, which are likely to be parts of the object of interest. The tracked keypoints contribute towards better temporal coherence of the object segmentation. A probabilistic occupancy map of the object is extracted using such initial object segmentation and a probabilistic shape model. The map is utilized in a classifier that operates both on pixels and regions. The aim of the classifier is to extract a trimap consisting of foreground, background and unknown areas. The trimap is employed by graph-cut. The outcome of the graph-cut is used in on-line learning of the shape model. The performance of the algorithm is demonstrated on freely available test sequences.

1 Introduction

Automatic segmentation of monocular video is a very challenging task. The graph-cut technique of Boykov and Jolly [1] has stimulated an explosion of interest toward automatic segmentation, rising quickly to become one of the most influential and leading algorithms for interactive image segmentation [2]. Segmentation of object in video is different from a segmentation of object in a single image. Video segmentation should preserve temporal consistency among segmentations and in consequence should indicate that extracted segments belong to the same objects. Recent research in this area led to compelling, real-time algorithms, either based on motion [3] or stereovision [4].

Extraction of foreground objects in monocular video is less constrained and often requires additional assumptions. Research termed as layer motion segmentation [5][6] received considerable interest of the computer vision community in the past decades. The general objective of methods falling into this category is to automatically extract motion coherent regions. However, such methods are computationally expensive and often require off-line learning as well as batch processing of the whole sequences. Segmentation of objects in video under assumption that the background is static [7][8] may yield confused results due to moving background objects, shadows, etc.

Most of the current approaches to image/video segmentation rely primarily on image-based criteria, such as color, gray level, or texture regularity of image

regions in context of smoothness and continuity of bounding outlines. In bottom-up approaches an image is first segmented into regions and then identification which regions correspond to single object takes place with regard to continuity. In complementary approach referred to as top-down segmentation, prior knowledge about an object, such as its possible shape, color, or texture is utilized to guide the segmentation. The difficulties inherent in pure low-level segmentation stimulated elaboration of top-down, class-specific segmentation algorithms [9]. In the algorithm [10] the training is done on a set of learned pictorial structures (LPS). The learned models are then used in a detection cascade. The OBJ-CUT algorithm provides excellent segmentations by the use in min-cut a good initialization, which was obtained by LPS.

In [1] the video segmentation is achieved by passing a complete set of frames to the algorithm and treating the entire sequence as 3D grid of pixels. Motion information has been used for video segmentation for a long time [11][7][12]. The motion segmentation algorithms assume that the foreground and background objects have unlike motion patterns. The objective of methods termed as layer-based motion segmentation is to automatically extract motion coherent regions [7][5]. However, certain methods falling into this category require off-line learning and batch processing of the whole video [7][5]. Occlusions make the motion based segmentation algorithms prone to errors, in particular at object boundaries. Therefore some work has been done to elaborate methods dealing with such errors [5]. The segmentation algorithm [11] is based on Markov Random Fields (MRF) that are built on three energy terms, namely motion, intensity and boundary. The assumption behind the MRF based model is that pixels spatially close to one another have a tendency to be in an identical layer. With the help of MRF the holes in the segments are suppressed and regular boundary shapes are preferred. However, the calculation of MRF-like constraints is computationally expensive. In [13] a simple body model and MRF are utilized in video segmentation at the blob level. Temporal consistency over segmented blobs is maintained by the usage of weighted bipartite matching. However, while segmentation of videos with real-world content, some blobs can be merged into single one, whereas a single blob can be split into multiple blobs and in the last resort it can disappear. The mentioned phenomena make blob tracking a challenging task. In [3] a spatio-temporal derivatives based model, which has been learned off-line from labeled data is employed in a classifier operating jointly on intensity change and contrast to distinguish between motion and non-motion. The priors for segmentation are represented by a second order, temporal Hidden Markov Model and spatial MRF. Finally, the layer segmentation is done by graph-cut. However, this algorithm needs to be trained by the use of ground-truth and is relatively slow.

In the algorithm presented in this work, the final delineation of the object is done via graph-cut. Using the object mask extracted via graph-cut a probabilistic shape model is learned on-line. A partitioning of the image based on pairwise region comparison takes place at the beginning of each frame. Using a collection of the tracked point features we extract adjacent segments, which

are likely to be parts of the object of interest. Next, using a component consisting of such segments a distance function is constructed. Given the motion of the object, which is determined on the basis of the tracked features an adjustment of the probabilistic shape model to likely location of the object takes place. Afterwards, we extract a probabilistic occupancy mask of the object. It is constructed on the basis of the adjusted probabilistic model of the object shape and the object distance function. A classifier operating on (i) probabilistic occupancy map, (ii) mean colors of the segments and (iii) the quantity of the tracked features in each segment constructs a trimap consisting of foreground, background and unknown areas. The graph-cut responsible for final delineation of the object is built on such a trimap. The smoothness term in the energy model is adapted gradually according to the occupancy map. The segments only partially covered by probabilistic occupancy map or holding too few features are classified as background. In particular, some segments arising during changes of object appearance have typically no successfully tracked features or the number of features is too small to assign such segment to the object. But with support of the classifier such feature-less object fragments obtain keypoints for tracking. At last such features contribute significantly to better object segmentation as well as temporal coherence in segmentation.

The proposed method is different from recent algorithms making use of graph-cut in video segmentation. Most of the relevant work is concerned with improving the efficiency of graph-cut [14], the use of various image attributes [15] and models [3][8], which are used in a single-step segmentation via graph-cut only. In our approach, in several segmentation steps we employ tracking of object fragments, a classification at intermediate level, on-line learning of the object shape, and in particular, we operate not only at pixel level but also use regions.

The rest of the paper is organized as follows. In the next Section we briefly describe video segmentation via pairwise region comparison. Section 3. is devoted to on-line learning of a probabilistic model of the object shape. The video segmentation via graph-cut built on the tracked object fragments and the learned shape model is explained in Section 4. In Section 5. we report results, which were obtained in experiments. Finally, some conclusions follow in the last Section.

2 Spatio-temporal figure-ground segmentation via pairwise region comparison and point matching

Figure-ground segmentation refers to a delineation of a region in an image such that it contains the object of interest. Spatio-temporal segmentation consists in delineating one or multiple objects from images rather than partitioning video into disjoint regions. Video segmentation should keep temporal consistency among segments in consecutive frames. This means that segments of a given image should relate to the segments of the previous one such that they do not belong to different objects.

Exact estimation of optical flow is computationally costly due to extensive search in the neighborhood of every image pixel. Therefore, in order to achieve

segmentation of video in real-time as well as to preserve temporal consistency of the segmentation, our algorithm relies on propagating the object seed features to the next frame. A set of adjacent blobs selected manually or automatically in the initialization step constitutes an object to be tracked. In each frame the algorithm first partitions the image into regions, then using the tracked features it identifies the image regions that correspond likely to the object of interest.

The input image is partitioned into segments using method from work [16]. It selects edges from an undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, in which every pixel corresponds to a node $v_i \in \mathcal{V}$ and neighboring pixels are connected by unidirected edges $(v_i, v_j) \in \mathcal{E}$. The segmentation consists in partitioning of vertices $\mathcal{V}$ into components such that each component corresponds to a connected component in the graph $\mathcal{G}' = (\mathcal{V}, \mathcal{E}')$, where $\mathcal{E}' \subseteq \mathcal{E}$. The evidence for a boundary between components is based on (i) intensity differences along the boundary and (ii) intensity differences between neighboring pixels within regions.

The internal difference $Int(\mathcal{C})$ of a component $\mathcal{C} \subseteq \mathcal{V}$ is defined in the following manner:

$$Int(\mathcal{C}) = \max_{e \in MST(\mathcal{C},\mathcal{E})} w(e), \tag{1}$$

and it is the largest weight in the minimum spanning tree $MST(\mathcal{C}, \mathcal{E})$ of the component. A component remains connected if it consists of edges with weights at least $Int(\mathcal{C})$. The difference between two components $\mathcal{C}_1, \mathcal{C}_2 \subseteq \mathcal{V}$ is defined as the minimum weight edge in the following manner:

$$Dif(\mathcal{C}_1, \mathcal{C}_2) = \min_{v_i \in \mathcal{C}_1, v_j \in \mathcal{C}_2, (v_i, v_j) \in \mathcal{E}} w(v_i, v_j). \tag{2}$$

In case when there is no edge connecting $\mathcal{C}_1$ and $\mathcal{C}_2$ the difference takes ∞. The minimum internal difference $MInt$, which is defined as follows:

$$MInt(\mathcal{C}_1, \mathcal{C}_2) = \min(Int(\mathcal{C}_1) + \tau(\mathcal{C}_1),\ Int(\mathcal{C}_2) + \tau(\mathcal{C}_2)) \tag{3}$$

is employed to verify if there is a boundary between a pair of components, where $\tau(\mathcal{C}) = \kappa / \mid \mathcal{C} \mid$, and κ is a threshold. The running time of the algorithm is nearly linear and the algorithm is fast in practice. In a preprocessing stage a Gaussian filter has been applied in order to smooth the images to be segmented. During postprocessing a merging of very small components to their neighbors has been realized. In our implementation the edge set $\mathcal{E}$ has been constructed from pairs of neighboring 8-connected pixels. The edge weights assume the values of the Euclidean distance (L_2 norm) between color components in RGB space. Figure 1 depicts some segmentation results of the `Foreman` image for two different thresholds parameters κ, which were set to $\kappa = 50$ and $\kappa = 160$, respectively. We can observe that despite similar colors of the background and the helmet, for $\kappa = 50$ the algorithm quite accurately delineated the foreground boundary.

Owing to graph based representation of the segmented image a single click on the mouse can activate or deactivate the whole segments. Such active segments are overlaid transparently on the input image. They constitute a very

Fig. 1. Image segmentation via pairwise region comparison using different values κ.

simple and intuitive way to adjust the initial object segmentation to the user liking. Given the user delineation of the object in the first frame, the algorithm extracts object keypoints using method [17], which focuses on selecting good features for tracking. Next, some additional point features are spread evenly to cover the whole object of interest. During generation of such features a predefined minimum distance to the closest feature is taken into account. At this stage the algorithm verifies if each segment contains sufficient number of point features with respect to its area. The complete feature set consists of both such keypoints and good features. Tracking of the feature set across images allows the algorithm to find the object from one image to another. On the other side, the segmentation information supports following the entire object instead of just a few features on it. Instead of calculating an optical flow for the whole image the algorithm calculates the local displacement vector for each feature. The mean location of all valid features is used in calculating the object motion. A validity checking is performed to estimate wrong inter-frame correspondences among point features. If the color difference between keypoints at the current and previous location is above a threshold the feature is rejected. The median direction of remaining features is calculated. All features that differ more than a predefined value from such direction are rejected. Next, using the valid features the motion of the object is calculated. The tracking of point features is done on the basis of pyramidal implementation of the optical flow [18]. The optical flow is computed with sub-pixel accuracy on the basis of bilinear interpolation. Owing to pyramidal implementation of the optical flow the object segmentation can cope with large motions.

3 On-line learning of probabilistic model of the shape

The object segmentation obtained by graph-cut may drift and thus evolve into arbitrary shapes due to clutter and ambiguous backgrounds. Severe segmentation ambiguities may occur when the images are noisy or the illumination changes. Recent approaches also employ spatial Gaussian mixture models (GMM) in video segmentation [7]. However, most existing algorithms do not learn on-line or relies on small motion assumption. Hence, they assume that the mixture models from the current frame can be applied to segmentation of the next frame. In our approach a dynamic Gaussian prior imposes preferences for shapes similar to examples that have been seen just before, as it propagates hypotheses about the

object shape. In particular, the proposed dynamic Gaussian prior can also encode the knowledge about the initial shape of the object. In the final delineation of the object the GMM model can give an estimate of the desired segmentation. At this stage it prevents the object shape from evolving into the arbitrary shape in the course of segmentation, particularly when there is a significant overlap between foreground and background pixels.

In our approach the compact object shape consists of N points $\{\xi_i\}_{i=1}^N$ of dimensionality $d = 2$. The shape is modeled with a mixture of K components. The likelihood that a pixel j belongs to the object can be expressed in the following manner:

$$p(\xi_j) = \sum_{k=1}^{K} p(k)p(\xi_j|k) \tag{4}$$

where $p(k)$ is the prior of the k-th component and $p(\xi_j|k)$ is the conditional density function. In the sequel, for probability density function consisting of Gaussians, the parameters in (4) are denoted as $p(k) = \pi_k$, $p(\xi_j|k) = \mathcal{N}(\xi_j; \mu_k, \Sigma_k)$, where μ_k and Σ_k are parameters of the model. The conditional density function $p(\xi_j|k)$ takes the following form:

$$p(\xi_j|k) = \frac{1}{(2\pi)^{\frac{d}{2}}|\Sigma_k|^{\frac{1}{2}}}\, e - \frac{(\xi_j - \mu_k)^T \Sigma_k^{-1}(\xi_j - \mu_k)}{2} \tag{5}$$

where d is dimensionality, $\{\pi_k, \mu_k, \Sigma_k\}_{k=1}^K$ are parameters of the model and denote prior, mean vector and covariance matrix, respectively. Given the initial object shape the full-covariance Gaussian mixture can be learned off-line using iterative Expectation Maximization (EM) algorithm. The k-means clustering algorithm can be used to provide a rough estimate of the model parameters. In the EM algorithm the parameters are estimated iteratively as follows:

E-step:
$$p_{k,j}^{(i+1)} = \frac{\pi_k^{(i)} \mathcal{N}(\xi_j; \mu_k^{(i)}, \Sigma_k^{(i)})}{\sum_{l=1}^{K} \pi_l^{(i)} \mathcal{N}(\xi_j; \mu_l^{(i)}, \Sigma_l^{(i)})} \tag{6}$$

M-step:
$$\pi_k^{(i+1)} = \frac{\sum_{j=1}^{N} p_{k,j}^{(i+1)}}{N} \tag{7}$$

$$\mu_k^{(i+1)} = \frac{\sum_{j=1}^{N} p_{k,j}^{(i+1)} \xi_j}{\sum_{j=1}^{N} p_{k,j}^{(i+1)}} \tag{8}$$

$$\Sigma_k^{(i+1)} = \frac{\sum_{j=1}^{N} p_{k,j}^{(i+1)} (\xi_j - \mu_k^{(i+1)})(\xi_j - \mu_k^{(i+1)})^T}{\sum_{j=1}^{N} p_{k,j}^{(i+1)}} \tag{9}$$

In each iteration i the algorithm computes the average log-likelihood as follows:

$$\mathcal{L}^{(i)} = \frac{1}{N} \sum_{j=1}^{N} \log \pi_k^{(i)} \mathcal{N}(\xi_j; \mu_k^{(i)}, \Sigma_k^{(i)}) \tag{10}$$

The algorithm stops if $\mathcal{L}^{(i+1)}/\mathcal{L}^{(i)} < th$, where th is a predefined value.

Let us assume that in time t the learning algorithm outlined above converged in iteration ii. Given the parameters $\{E_k^{(ii)}, \mu_k^{(ii)}, \Sigma_k^{(ii)}\}_{k=1}^K$, which were obtained in the iteration ii and stand for the cumulated posterior probabilities $E_k^{(ii)} = \sum_{j=1}^N p_{k,j}^{(ii)}$, mean vector and covariance matrix, respectively, we define the following parameters: $E_{\mu_k}^{(ii)} = \mu_k^{(ii)} \sum_{j=1}^N p_{k,j}^{(ii)}$, $E_{\Sigma_k}^{(ii)} = \Sigma_k^{(ii)} \sum_{j=1}^N p_{k,j}^{(ii)}$. Assuming that posterior probabilities remain the same, i.e. the parameters $E_k^{(ii)}$ do not change their values, when in time $t+1$ new data $\{\xi_j\}_{j=1}^M$ are considered by the model, the model can be updated according to the following steps:

$$\text{E-step:} \qquad p_{k,j}^{(i+1)} = \frac{\pi_k^{(i)} \mathcal{N}(\xi_j; \mu_k^{(i)}, \Sigma_k^{(i)})}{\sum_{l=1}^K \pi_l^{(i)} \mathcal{N}(\xi_j; \mu_l^{(i)}, \Sigma_l^{(i)})} \tag{11}$$

$$\text{M-step:} \qquad \pi_k^{(i+1)} = \frac{E_k^{(ii)} + \sum_{j=1}^M p_{k,j}^{(i+1)}}{N+M} \tag{12}$$

$$\mu_k^{(i+1)} = \frac{E_{\mu_k}^{(ii)} + \sum_{j=1}^M p_{k,j}^{(i+1)} \xi_j}{E_k^{(ii)} + \sum_{j=1}^M p_{k,j}^{(i+1)}} \tag{13}$$

$$\Sigma_k^{(ii,i+1)} = (\mu_k^{(ii)} - \mu_k^{(i+1)})(\mu_k^{(ii)} - \mu_k^{(i+1)})^T \tag{14}$$

$$\Sigma_k^{(i+1)} = \frac{E_{\Sigma_k}^{(ii)} + E_k^{(ii)} \Sigma_k^{(ii,i+1)} + \sum_{j=1}^M p_{k,j}^{(i+1)} (\xi_j - \mu_k^{(i+1)})(\xi_j - \mu_k^{(i+1)})^T}{E_k^{(ii)} + \sum_{j=1}^M p_{k,j}^{(i+1)}} \tag{15}$$

The update of the model stops when the ratio of the average log-likelihoods from iteration i and $i+1$ is below a predefined threshold.

Figure 2 depicts the ability of GMMs to model the compact object shapes. The tests have been carried out using the standard `Foreman` test sequence. In the input images #1 and #2 (left column) the ellipses resulting from the shape model are overlaid transparently. The shape model has been trained via the EM, initialized by k-means, using pixel locations taken from an object map (see image in top row and middle column). Then the shape model has been updated via the use of pixel locations from an object map reflecting a segmented foreground (bottom row). Looking at the ellipses at bottom left part of the object map, which cover the missing part of the torso we can observe how the model adapts to the changes of the shape. This capability can also be observed in the object probability map that is depicted in the third column. It can be observed that due to the missing part of the torso the likelihood is something smaller in this part of the image.

The object likelihood has been constructed as a product of two mixtures of Gaussians. It is constructed under assumption that object color is not dependent on its location. The first mixture consists of 20 spatial Gaussians. The second one approximates the color distribution in RGB color space. The fixed color model has been built using 10 Gaussians. By the change of significance of spatial part

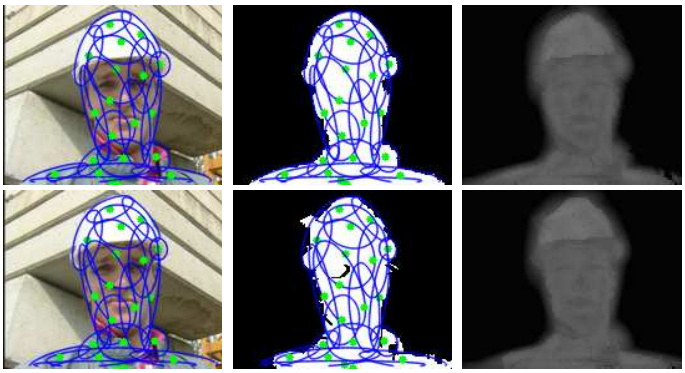

Fig. 2. Modeling the object shape with color-spatial GMMs.

in the color-spatial probabilistic model we can accommodate the model to the object speed, shape variability, etc. Examples shown in Fig. 3 illustrate the effect of balancing between color and spatial part in the model.

Fig. 3. Balancing between color and the spatial part in the object model.

Using the current object shape, the object likelihood map can be constructed in several ways. As discussed above, it can be composed on the basis of the current frame and former one. Given the motion of the object the spatial model from the previous frame can be projected forward in time trough adjustment of $\mu^{(ii)}$ parameter. Given the pose of the object undergoing segmentation the shape model from the first frame can be employed in the update of the model. In most cases the first and simplest approach has proven to be effective and sufficient.

4 Video segmentation via graph-cut using tracked object fragments and learned shape model

The image segmentation $\mathcal{X}$ can be uniquely determined by minimum cut $\mathcal{C}$ on the graph $\mathcal{G}$ with two terminal nodes such that the terminals are separated on the induced graph $\mathcal{G} = (\mathcal{V}, \mathcal{E} \backslash \mathcal{C})$ [1]. The minimization is done using Gibbs energy function that is formulated over the unknown labels $\mathcal{X}$ as the sum of data term

and boundary term:

$$E(\mathcal{X}) = E_{data}(\mathcal{X}) + \lambda E_{smooth}(\mathcal{X}) = \sum_{i \in \mathcal{V}} E_1(x_i) + \lambda \sum_{\{i,j\} \in \mathcal{N}} E_2(x_i, x_j), \quad (16)$$

where $\mathcal{V}$ is the set of all vertices in graph $\mathcal{G}$, whereas $\mathcal{N}$ is the neighborhood formed from the vertex connectivity. The regularization factor λ balances the data term E_1 and the smooth cost E_2. The data term E_1 is a likelihood energy, which expresses cost of assigning label x_i to node i. The prior term E_2 encodes the cost when the adjacent nodes i and j have the labels x_i and x_j, respectively. The min-cut of the weighted graph determines the segmentation that best separates the object from its background. In our work we use the recent max-flow/min-cut implementation from work [19].

The weight of a t-link connecting a terminal with the pixel corresponds to cost of assigning the label to this pixel. There are two t-links for each pixel. The weights of these links depend on the state of the object trimap. If a pixel belongs to object, the value of background t-link is set to 0, whereas the foreground t-link is set to K. If a pixel has been classified as background the value of background t-link is set to K and the value of foreground t-link is set to 0. The value of K is set as follows: $K = 1 + \max_{i \in \mathcal{V}} \sum_{j:\{i,j\} \in \mathcal{N}} E_2(x_i, x_j)$. To set the weights of t-links for pixels of unknown state we use the probabilities obtained from two separate GMMs. The first GMM is used to model the distribution of background color. The second one is the model used in determining the object likelihood, discussed in Section 3. The data cost is the negative log likelihood of the color given the label. The smoothness term is a standard Potts model: $E_2(x_i, x_j) = \mathrm{dist}(i,j)^{-1} \exp(-\beta \|C(i) - C(j)\|^2) + \lambda_2$, where $\mathrm{dist}(i,j)$ stands for the distance between pixels i and j, $C(i)$, $C(j)$ are RGB colors of two neighboring pixels i and j, λ_2 is set to 10. β is a robust parameter that is set to $(\langle \|C(i) - C(j)\|^2 \rangle)^{-1}$ [2], where the expectation stands for an average over the image. The distance term $\mathrm{dist}(\cdot)$ is used to reduce a tendency of the segmentation algorithm towards diagonal cuts, whereas the aim of λ_2 is to discard small and high-contrasted regions.

In order to generate the trimap with very likely object pixels we employ a classifier that operates both on pixels as well as on segments. The outcome of the classifier is a binary mask of the object, background pixels and pixels of unknown state. We utilize a decision tree classifier, which input consists of: the graph constructed during segmentation via pairwise region comparison in order to calculate the quantities given by (2) and (3), the number of valid features on each segment, the probabilistic occupancy map of the object, and the quantities $\tilde{d}_k^{\mathcal{F}}$, $\tilde{d}_k^{\mathcal{B}}$ that are explained below.

As we already mentioned, at the initialization stage we manually extract foreground and background segments. For each segment we compute the mean colors and the clustering takes place separately for foreground and background. As a result we obtain cluster centroid colors $\{P_n^{\mathcal{F}}\}_{n=1}^{N}$ and $\{P_n^{\mathcal{B}}\}_{m=1}^{M}$, where N and M stand for the number of foreground and background clusters, re-

spectively. At this stage we use the well-known k-means algorithm because it acknowledged its usefulness in image segmentation [20]. At run-time, just before the classification, for each candidate segment k we compute the minimum distance from its mean color C_k to foreground and backgrounds clusters as follows: $d_k^{\mathcal{F}} = \arg\min_n \|C_k - P_n^{\mathcal{F}}\|$, $d_k^{\mathcal{B}} = \arg\min_m \|C_k - P_m^{\mathcal{B}}\|$. Then we compute the mentioned quantities: $\tilde{d}_k^{\mathcal{F}} = d_k^{\mathcal{F}}/(d_k^{\mathcal{F}} + d_k^{\mathcal{B}})$, $\tilde{d}_k^{\mathcal{B}} = d_k^{\mathcal{B}}/(d_k^{\mathcal{F}} + d_k^{\mathcal{B}})$.

The graph-cut extracts a binary map representing the object of interest. In the course of segmentation it adapts the value of β with reference to the probabilistic occupancy map of the object. The larger the distance of the considered background pixel to the object, the smaller the probability of the cut is. Using pixels belonging to the object map the probabilistic model of the object shape is learned on-line as discussed in Section 3. With the support of the object mask the keypoints with tentative matches are replaced by newly generated keypoints.

5 Experiments

To test the effectiveness of the proposed algorithm we performed various experiments on several test sequences. We tested feature tracking via Kalman filter and a linear motion model. The confidence measure that the candidate keypoint corresponds to the tracked one was built on: (i) scaled differences between gradient vectors at the tracked and candidate locations, scaled by the standard deviation of the gradient, (ii) difference between the predicted keypoint location and the candidate keypoint location, scaled by the standard deviation in position, (iii) cross-correlation between color patches. The standard deviation was used to express the reliability of the measure and was extracted from the covariance matrix of the Kalman filter. The method permits reliable keypoint matching. However, computation burden of this method is significant because for each tracked keypoint it adds every candidate keypoint to the list of possible matches. Our research findings show that for our algorithm the optical-flow based feature tracking is effective and sufficient.

At image partitioning stage we tested the well-known mean-shift segmentation algorithm [21]. This algorithm determines spatio-temporal segments with better coherence. But such stable segments were important in an earlier version of the algorithm, which has only been based on feature tracking and graph-cut in the final segmentation. The segments, which are obtained by the current method usually correspond to large homogenous fragments of object parts. Given an initial set of image pixels corresponding to the visible area of an object, we can track it over subsequent frames, even if the viewpoint changes and different parts of the object come into and out of view. It is worth to note that the computation time of the mean-shift based segmentation is several times larger in comparison to the method currently used.

Figure 4 depicts some segmentation results that were obtained using the `Foreman` test sequence. The person is segmented as complete entity, despite non-stationary background, ambiguous and cluttered background as well as variability of the object shape.

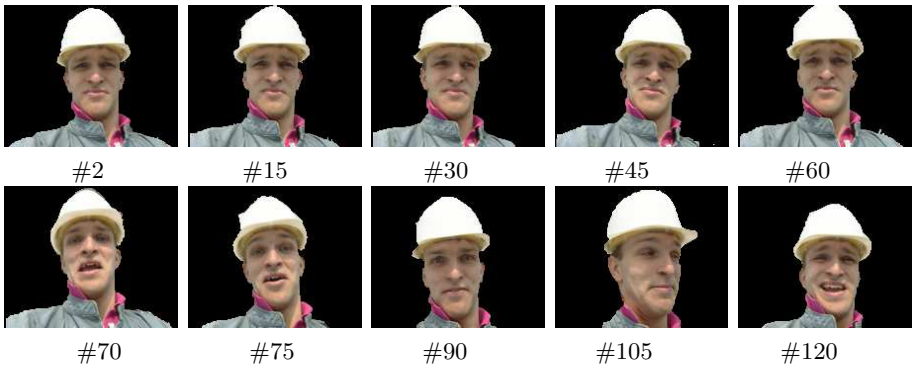

Fig. 4. Video segmentation using `Foreman` test sequence.

In order to conduct comparison tests we implemented the algorithm [3]. This method requires training with ground-truth of different environments. Moreover, the weights of the four energy terms need to be tuned for the segmented sequence. Experiments done at `AT` sequence[1] showed that the discussed method does not work well if the foreground has similar colors as the background, in particular when there is little motion in the foreground. Figure 5 shows some segmentation results, which were obtained by our method on the mentioned above test sequence. The number of background pixels with similar color to the foreground

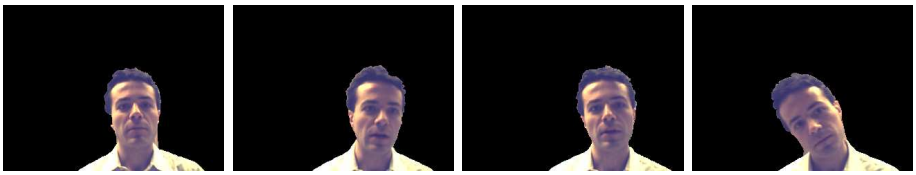

Fig. 5. Video segmentation using `AT` sequence.

color, which were labeled as the object is far smaller. Our experiments demonstrated that the objects are segmented as complete entities, despite considerable variability of the object shape as well as cluttered background. The method can even handle situations in which object boundary shares similar colors with the background.

6 Conclusions

The strength of our algorithm lies in multi-stage segmentation. In several segmentation steps we employ tracking of object fragments, classification at inter-

[1] Available on: http://research.microsoft.com/vision/cambridge/i2i

mediate level, on-line learning of the object shape, and in particular, we operate not only at pixel level but also utilize regions. The algorithm is robust to image noise and can achieve accurate region boundaries. One of the future research directions of the presented approach is to explore the object delineation via a graph-cut built on pre-segmented images as a means to reduce the computational cost.

References

1. Boykov, Y., Jolly, M.: Interactive graph cuts for optimal boundary and region segmentation of objects in n-d images. In: Proc. of ICCV. (2001) 105–112
2. Rother, C., Blake, A., Kolmogorov, V.: Grabcut - interactive foreground extraction using iterated graph cuts. In: Proc. of ACM SIGGRAPH. (2004) 332–343
3. Criminisi, A., Gross, G., Blake, A., Kolmogorov, V.: Bilayer segmentation of live video. In: Proc. of CVPR. (2006) 53–60
4. Kolmogorov, V., Criminisi, A., Blake, A., Ross, G., Rother, C.: Bi-layer segmentation of binocular stereo video. In: Proc. of CVPR. (2005) 1186–1193
5. Xiao, J.J., Shah, M.: Motion layer extraction in the presence of occlusion using graph cut. In: Proc. of CVPR, Washington, D.C. (2004) II:972–979
6. Kumar, P., Torr, P., Zisserman, A.: Learning layered motion segmentations of video. In: Proc. of ICCV. (2005) 33–40
7. Wren, C., Azarbayejani, A., Darrell, T., Pentland, A.: Pfinder: real-time tracking of the human body. PAMI **19** (1997) 780–785
8. Sun, J., Zhang, W., Tang, X., Shum, H.Y.: Background cut. In: Proc. of European Conf. On Computer Vision. (2006) 628–641
9. Leibe, B., Leonardis, A., Schiele, B.: Combined object categorization and segmentation with implicit shape model. In: LNCS, Springer, vol. 4170. (2004) 508–524
10. Kumar, M., Torr, P., Zisserman, A.: Obj cut. In: Proc. of CVPR. (2004) 18–25
11. Black, M.: Combining intensity and motion for incremental segmentation and tracking. In: European Conf. on Computer Vision. (1992) 485–493
12. Khan, S., Shah, M.: Object based segmentation of video using color, motion and spatial information. In: Proc. of CVPR. (2001)
13. Park, S., Aggarwal, J.K.: Segmentation and tracking of interacting human body parts under occlusion and shadowing. In: Proc. of Workshop on Motion and Video Computing. (2002) 105–111
14. Juan, O., Boykov, Y.: Active graph cuts. In: Proc. of CVPR. (2006) 1023–1029
15. Xu, N., Bansal, R., Ahuja, N.: Object segmentation using graph cuts based active contours. In: Proc. of CVPR. (2003) 46–53
16. Felzenszwalb, P.F., Huttenlocher, D.P.: Efficient graph-based image segmentation. Int. J. of Computer Vision **59** (2004) 167–181
17. Shi, J., Tomasi, C.: Good features to track. In: Proc. of CVPR. (1994) 593–600
18. Lucas, B., Kanade, T.: An iterative image registration technique with an application to stereo vision. In: Proc. Int. Joint Conf. on Artificial Intell. (1981) 674–679
19. Boykov, Y., Kolmogorov, V.: An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. PAMI **26** (2004) 1124–1137
20. Plataniotis, K., Venetsanopoulos, A.: Color Image Processing and Applications. Springer, New York, Heidelberg (2000)
21. Comaniciu, D., Meer, P.: Mean shift: A robust approach toward feature space analysis. PAMI **24** (2002) 603–619