Fundamenta Informaticae 90 (2009) 1–15
DOI 10.3233/FI-2009-0001
IOS Press

Object segmentation in video via graph cut built on superpixels

Bogdan Kwolek*
Rzeszów University of Technology
Rzeszów, Poland
bkwolek@prz.edu.pl

Abstract. This paper proposes a real-time scheme for object segmentation in video. In the first stage a segmentation based on pairwise region comparison is utilized to oversegment image through extracting superpixels. Next, the algorithm applies the graph cut built on such superpixels, instead of the image pixels. Owing to the optimization is performed on a simpler graph and in consequence the object segmentation runs in shorter time. Tracking of object features over time contributes toward improved segmenting the object from one image to another. The segmentation information supports following the entire object, instead of just a few features on it. The objects are segmented correctly as complete entities, despite the high variability of the object shape and cluttered background. Experimental results illustrate the efficiency and effectiveness of the algorithm.

Keywords: Cognitive Vision Systems, Image Segmentation, Object Recognition

1. Introduction

Visual object recognition and scene interpretation are key capabilities needed both by biological organisms and for successful operation of service robots. Over the last two decades there has been meaningful progress in computer vision. At present it is possible to carry out quality control and domain specific tasks such as face detection and recognition, visual surveillance and event detection, using only image information. According to paradigm proposed in [17], a vision system is a succession of bottom up processes, which provide the means to transform sensory information to a higher abstraction level. The

*Address for correspondence: Rzeszów University of Technology, W. Pola 2, 35-959 Rzeszów, Poland

succession of processes is: segmentation, reconstruction and recognition. These three steps turn the image into a symbolic description of the scene. Marr's computational theory of human vision, which was the first complete methodology for the design of information systems, still has a great influence on the research in artificial vision.

Cognitive Informatics (CI) is a new discipline that studies the natural intelligence and the internal information processing mechanisms in the brain, and their applications in information processing [32]. A considerable progress, especially in the theoretical frameworks of cognitive informatics and denotational mathematics for CI has been done recently. Particularly, computing methodologies and technologies were developed to extend human capability, reachability, persistency, memory, and information processing speed. The cognitive models of human memory [33], particularly the Sensory Buffer Memory (SBM), Short-Term Memory (STM), Long-Term Memory (LTM), and Action-Buffer Memory (ABM), and their mapping onto the physiological organs of the brain, expose the fundamental mechanisms of cognitive informatics. A cognitive informatics based approach to automatic pattern interpretation, directed for semantic categorization has been addressed in [18].

To achieve more robust, resilient, and adaptable computer vision systems, which guarantee higher level of functionality, an emerging and multi-faceted discipline, namely cognitive vision has been introduced in the past few of years in accordance to ability to learn, adapt, balance alternative solutions during analysis and interpretation. A cognitive system is a system that can change its behaviour based on reasoning, using observed evidence and domain knowledge. Such a system is embedded in the world and interacts with environment to acquire knowledge and to carry out its mission objectives. The functional capabilities of any cognitive vision system include detection, tracking, segmentation and recognition, classification and categorization, long-term predictive capability, planning, decision taking and understanding. The key characteristic underlying all cognitive vision systems is its capability to demonstrate robust performance even in circumstances that were not foreseen by a designer of the system. Ideally, a cognitive vision system should be capable to recognize and to adapt to novel circumstances in the environment, generalize the physical layout of the scene, predict future configurations of the visual environment, and in particular predict and understand intentions of people. One important property of any cognitive system is the ability not only to recognize objects, but also to perform recognition by means of categorization.

When people look at objects, they subconsciously and effortlessly partition them into parts [3, 7]. They utilize their visual sense to segment the surrounding environment into different objects to help recognize them. Indeed, the studies [23] suggested that humans may be able to categorize objects when shown only silhouettes. Segmentation is also viewed as a fundamental problem that the visual system should apply quite early [17], before further processes such as object recognition. It is one of the crucial steps toward image understanding. Many current accounts refer to segmentation as mid-level visual process, which is an intermediate stage between initial description of raw image statistics, and succeeding matching against representations of known objects in long-term memory. Objects are represented through composition of simple elements and the relations between them [20]. The capacity of visual working memory should be understood in terms of integrated objects and conjuctions rather than specific features [16]. The recognition process is also coupled with object segmentation via stored components, which serve for top-down processes in delineating object boundaries [29].

Segmentation of object in a single image is different from segmentation of object in video. Image segmentation refers to partitioning an image into regions that are homogeneous with respect to some feature. It can yield very different results for two very similar images. In video segmentation a consistency

among segmentations of each frame is important for object segmentation. Video segmentation requires that for a given image, the segmentation achieved should relate to the segmentation of the previous image and indicate that extracted segments belong to the same objects. The segmentation can be based on color, motion, texture and depth if images are acquired from a stereo-pair. Segmentation of videos is weakened by various types of uncertainty making single cue based segmentation ineffective. Recent techniques for interactive image and video segmentation [24, 31] have demonstrated the great effectiveness of the graph cut based segmentation using both color and contrast simultaneously in a model, which has been proposed by Boykov et al. [5]. The final foreground layer is determined globally through the min-cut based optimization. However, as demonstrated in [12] the usage of only color and contrast can be insufficient in segmentation of real world video shots.

Most attention in video segmentation has been focused on motion segmentation. However, due to occlusion and disocclusion, a segmentation based on motion only results in errors at boundaries. To overcome this, the segmentation algorithm [4] has been based on Markov Random Fields (MRF) having three energy terms, of motion, intensity and boundary. The assumption behind the MRF based model is that pixels spatially close to each other have a tendency to be in the same segment. With this constraint, the holes in the segments are suppressed and regular boundary shapes are preferred. However, the computation of MRF-like constraints is computationally expensive. In [9] the need for optical flow estimation has been removed. Instead, a classifier operating jointly on intensity change and contrast has been learned from labeled data and then applied to discriminate between motion and non-motion. Next, the prior for segmentation was represented by a second order, temporal, Hidden Markov Model and spatial MRF. Finally, the layer segmentation was achieved by graph cut. However, this algorithm needs to be trained with the usage of ground-truth and is relatively slow.

In this work, a two stage segmentation approach is utilized where images are first oversegmented into superpixels, and then a graph cut built on such superpixels is employed to delineate the object of interest. Using a collection of the tracked point features the algorithm extracts the adjacent object patches, which are likely to be parts of the object of interest. Given such patches a distance function is constructed, which serves as object prior in the graph cut based final delineation of the object. The object motion is estimated using the mean locations of features with correct inter-frame correspondences. The object seed for the graph cut is constructed through the overlap of object patches that were selected on the basis of the tracked features and the object mask, shifted into predicted object position.

The proposed method is different from recent algorithms that also make use of graph cut in video segmentation. Most of the relevant work is concerned with improving the efficiency of graph cut [11], the use of various image attributes [34] and models [9, 27], which are then used in a single-step segmentation via graph cut only. In our approach we employ tracking of object fragments, and in particular, we operate not only at pixel level but also utilize superpixels as well as object patches. Superpixels are obtained through an oversegmentation of the image and they aggregate visually homogeneous pixels while respecting natural object boundaries. Superpixels were proposed in [19], where they have been used for structured scene analysis. While there exist various techniques to oversegment an image, computational cost was one of the barriers of their use in real-time algorithms. Our work is based upon a recently-developed fast segmentation method, which is described in [10]. The contribution of this work lies in a real-time object segmentation algorithm, which requires no training. Owing to object keypoint tracking it achieves reliable and consistent in time a delineation of an object.

The paper is organized as follows. The next section is devoted to the usage of graphs in image segmentation. In the first part of section 3 we show how the superpixels are constructed, whereas in the

second part we indicate how to segment the object in video using the superpixels and the tracked object keypoints. Next section on graph cut based object segmentation begins with the presentation of the energy model and then presents some results of object segmentation in video via energy minimization. In section 5, starting from presentation of the algorithm for video segmentation using graph cut built on superpixels and the support of the tracked keypoints, we present some experimental results. Section 6 is devoted to object recognition using segmented object shapes. A conclusion is drawn in the last section.

2. Graphs in image segmentation

A graph is a kind of data structure, which consists of a set of nodes (also called vertices) and a set of edges that establish relationships (connections) between the nodes. A graph $\mathcal{G}$ is defined as an ordered pair $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ stands for a set of vertices, and $\mathcal{E}$ is a set of edges connecting the vertices. In practice the adjacency lists and adjacency matrix are used as main data structures for the representation of graphs. A depth-first/breadth-first search for a path between two nodes and finding the shortest path from one node to another are typical operations on graphs.

In image processing, graphs are used mainly to represent the irregular data structures. Graph-based segmentation techniques employ a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with vertices $v_i \in \mathcal{V}$ representing image pixels or features, and edges $e_i \in \mathcal{E}$ corresponding to pairs of vertices. Each edge has a corresponding non-negative weight $w(e)$, which is a measure of dissimilarity between elements connected by that edge.

Let $\mathcal{G} = \langle \mathcal{V}, \mathcal{E} \rangle$ be a weighted graph with two distinguished vertices $\{s, t\}$ called terminals. A cut $C \subset \mathcal{E}$ is composed by a set of edges that separate the terminals in the graph $\mathcal{G} = \langle \mathcal{V}, \mathcal{E} - C \rangle$, such that $s \in \mathcal{V}^s$ and $t \in \mathcal{V}^t$. With each edge $e \in \mathcal{E}$ is associated a weight $w(e)$ based on some relationship between the examples, such as the distance/similarity between them. The cost of the cut $\mid C \mid$ equals the sum of all the vertices that are cut. The minimum cut is the cheapest one among all cuts disconnecting the terminals. One of the fundamental results of combinatorial optimization states that the minimum cut can be found by finding a maximum flow from s to t. Although a maximum flow approach is utilized in physical networks, a large variety of systems with no inherent physical network can be formulated using network and a notion of network flow. In the algorithm determining max-flow the terminals are called source and sink, respectively, and the edge weights are treated as capacities [8]. Graph cuts can be utilized to find efficient solutions for broad variety of computer vision tasks such as image restoration, correspondence in a stereo-pair, and many other problems that can be formulated in terms of minimization of the energy. Unlike iterative algorithms like simulated annealing, graph cut algorithms can not be applied to an arbitrary energy function. Instead, for energy function to be minimized a graph construction should be elaborated. The question of what energy functions can be minimized via graph cuts was addressed in [13]. The mentioned work provides several graph constructions for quite general class of energy functions, which can be employed to solve vision problems. Under most formulation of computer vision problems, the minimum energy based outcome corresponds to maximum a posteriori estimate of the solution. In the worst case the cut with minimum cost can be computed in polynomial time. In practice, for graphs with many short paths among the source and the sink the running time is nearly linear. The graph cut based combinatorial optimization for a restricted class of energies finds a global minimum near given initial solution for a given set of boundary conditions.

3. Graph-based segmentation via pairwise region comparison

Image segmentation via pairwise region comparison will be presented as the first topic in this section. The use of optical flow in order to support segmenting the entire object in image sequences will be discussed in detail later. The section demonstrates that such an algorithm is capable to generate reliable object priors for graph cut based delineation of the object of interest.

3.1. Graph-based image segmentation via pairwise region comparison

Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be a weighted undirected graph. In an approach based on pairwise region comparison a segmentation $\mathcal{S}$ is a partition of $\mathcal{V}$ into components such that every image component $\mathcal{C}$ corresponds to a connected component in a undirected graph $\mathcal{G}' = (\mathcal{V}, \mathcal{E}')$, where $\mathcal{E}' \subseteq \mathcal{E}$. The elements in a component should be similar, whereas elements in different components should be dissimilar.

In [10] an image segmentation method based on measuring the dissimilarity between the elements along the boundary of the neighboring components in relation to the dissimilarity between neighboring elements of the components has been proposed. It relates the inter-component dissimilarities to the within component dissimilarities. To achieve this the algorithm considers the internal difference $Int(\mathcal{C})$ of a component $\mathcal{C} \subseteq \mathcal{V}$

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

which is the largest weight in the minimum spanning tree $MST(\mathcal{C}, \mathcal{E})$ of the component. A spanning tree of undirected graph is a subgraph which is a tree connecting all vertices together. A minimum spanning tree of a graph is a spanning tree whose weight sum is less than or equal to the weight sum of every other spanning tree [8]. Owing to the above formula only edges of weight at least $Int(\mathcal{C})$ are taken into account during constructing a connected component $\mathcal{C}$. The method considers also the difference between the two components $\mathcal{C}_1, \mathcal{C}_2 \subseteq \mathcal{V}$ as the minimum weight between them

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

If there is no edge connection between $\mathcal{C}_1$ and $\mathcal{C}_2$ the difference takes ∞. The minimum internal difference $MInt$, which is expressed in the following manner

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

is utilized to examine if the difference between the components $Dif(\mathcal{C}_1, \mathcal{C}_2)$ is large in comparison with the minimum internal difference between $\mathcal{C}_1$ and $\mathcal{C}_2$. The threshold function $\tau(\mathcal{C})$ is constructed on the basis of the size $\mid \mathcal{C} \mid$ of the component $\mathcal{C}$ and some constant parameter κ in the following manner:

$$\tau(\mathcal{C}) = \kappa / \mid \mathcal{C} \mid. \tag{4}$$

The threshold is responsible for checking the grade to which the difference between components is larger than the minimum internal difference during an evaluation if there is evidence for a boundary in a considered pair of components.

Given the above measures of similarity, the image segmentation operates according to the following procedure:

1. Sort edges by non-decreasing edge weight.
2. Start with an initial segmentation.
3. Consider the vertices v_i and v_j, which are connected by the edge with the least weight. If v_i and v_j are already in the identical component, then continue. Otherwise merge the two components, if and only if $MInt(\mathcal{C}_1, \mathcal{C}_2) \geq Dif(\mathcal{C}_1, \mathcal{C}_2)$.
4. Repeat step 3 for all edges in order.

In a preprocessing stage the input image is smoothed via Gaussian filter (with $\sigma = 0.8$). During postprocessing the algorithm merges very small components to their neighbors.

In our implementation of the segmentation algorithm the edge set $\mathcal{E}$ consists of pairs of neighboring 8-connected pixels. The edge weight is Euclidean distance (L_2 norm) between color components in RGB space. The distance is set to be the square root of the summation of the square distances among color components.

3.2. Object segmentation in video using pairwise region comparison and optical flow

Accurate estimation of dense optical flow is computationally expensive due to an extensive search in the neighborhood of each point. In order to achieve segmentation of video in real-time as well as to preserve temporal consistency of the segmentation, our approach is based on propagating the seed features to the next frame. Without loss of generality, let us assume that a grayscale 2-dimensional images I_t and I_{t+1} are used. The two values $I_t(x, y)$ and $I_{t+1}(x, y)$ are then the grayscale values at the locations (x, y), where x and y are the two pixel coordinates of an image point. Let us consider an image point $(u_x\ u_y)$ on the first image I_t. The goal of feature tracking is to find the location $(u_x + d_x\ u_y + d_y)$ on the second image I_{t+1} such as $I_t(u_x, u_y)$ and $I_{t+1}(u_x + d_x, u_y + d_y)$ are similar. The vector determined by (d_x, d_y) establishes the image velocity. On the basis of user segmentation of the object of interest in the first frame the algorithm extracts features according to method [26] focusing on selecting good features for tracking. Tracking of such features across multiple images help us 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, our method for object segmentation in video calculates the displacement vector for a single feature. To cope with propagation of segmentation of objects with little texture, in each frame we spread evenly additional points for tracking within the extracted object segment. We perform a validity check to estimate wrong inter-frame correspondences among features. The lost feature points are then reconstructed. If the color difference between keypoints at the current and previous location is above a threshold the feature is discarded. The median direction of remaining features is calculated. All features that differ more than a predefined value from such a direction are discarded. Next, using valid features the motion of the object is determined. The tracking of features takes place on the basis of pyramidal implementation of the optical flow [15]. The optical flow is computed with subpixel accuracy on the basis of bilinear interpolation. Owing to pyramidal implementation of the optical flow the object segmentation can cope with large motions.

The initialization of the object segmentation takes place on the basis of manually extracted segments. After click on the mouse the whole segment containing the selected pixel is activated and overlaid transparently on the image. While merging of the components the segmentation algorithm checks for the number of features within the segments. If the number of features within one of the segments in the pair

is below threshold, the segment is omitted in merging. The object can consist of several segments. A single segment representing the object of interest is extracted using a collection of the tracked features.

The algorithm has been tested on several video sequences. In a sequence `WalkByShop1cor.mpeg`[1] of size 288×384 a pair of pedestrians to be segmented crosses passage zones during varying illumination conditions. Figure 1b demonstrates the superpixels that have been extracted by the segmentation algorithm. The initial segmentation of the object, which has been achieved through only one click on the computer mouse is depicted in Fig. 1c. The mentioned above test sequence is provided with manually selected bounding boxes, which are stored in XML format for each frame of the sequence. We compared such annotated boxes with the bounding boxes of the segmented objects via the algorithm. For object-id = #2 that has been segmented in frames #700 - #1000 the average overlap between boxes is something smaller than 70%.

Figure 1. Image segmentation using pairwise region comparison and optical flow. Raw input image (a), segmented image (b), manually activated segment representing the object to be tracked (c).

4. Object segmentation by energy minimization

In the following subsections the graph cut based object segmentation is discussed. First we describe the energy model. Object segmentation in video via graph cut built on pixels is shown afterwards.

4.1. Energy model

In the context of vision tasks, graph terminals correspond to the set of labels that can be assigned to pixels. Typically, there are two types of edges in the graph, namely n-links and t-links. N-links are used to connect neighboring pixels or features and they represent the neighborhood relation in the image. Their weights represent the penalty for discontinuity between pixels or features. The weight of a t-link connecting a terminal with the pixel or feature corresponds to cost of assigning the label to the pixel or feature. If edges between examples that are alike to each other are assigned a high weight, then such examples are likely to be placed in the same vertex subset determined by the globally optimal min-cut.

We define the object segmentation in video as assigning a label l_j to every image pixel p_j, $j = 1, ..., J$, where J denotes the number of pixels in each frame, and l_j is a binary variable, i.e. assuming the values 0 and 1. The energy-based objective function E, which can also be perceived as the log likelihood of the posterior distribution of a Markov Random Field [14] is formulated over the unknown

[1] Sequence downloaded from site at: `http://groups.inf.ed.ac.uk/vision/CAVIAR/`

labels l of every pixel as the sum of regional term and boundary term:

$$\begin{aligned} E(l) &= E_{data}(l) + \lambda E_{smooth}(l) \\ &= \sum_{p \in \mathcal{P}} D_p(l_p) + \lambda \sum_{\{p,q\} \in \mathcal{N}} V_{p,q}(l_p, l_q), \end{aligned} \tag{5}$$

where $\mathcal{P}$ is the set of pixels in each image, $\mathcal{N}$ is the set 8-connected pairwise neighboring pixels, $\{p, q\}$ stands for unordered set, i.e. the sum is over unordered pairs of neighboring pixels. The regularization factor λ is responsible for balancing the data term $D_p(l_p)$ and smooth cost $V_{p,q}(l_p, l_q)$. If we assume that pixels form a 2D lattice, p can be expressed in terms of its coordinates $p = (x, y)$. The above energy function can efficiently be minimized by the algorithm [6].

The pairwise smoothness energy term $E_{smooth}(l)$ is a standard Potts model that is contrast sensitive:

$$E_{smooth}(l) = \sum_{\{p,q\} \in \mathcal{N}} V_{p,q}(l_p, l_q) = \sum_{\{p,q\} \in \mathcal{N}} \frac{1}{d(p,q)} e^{-\frac{(I_p - I_q)^2}{2\sigma^2}}, \tag{6}$$

where I_p denotes the intensity value of the pixel p, $d(p, q)$ stands for the distance between pixels p and q, σ is the average intensity difference between adjacent pixels in the image and is given by $\langle \|I_p - I_q\|^2 \rangle$. The distance term in denominator is used reduce the tendency towards diagonal cuts. It permits to capture the contrast consistency only along the segmentation boundary [24]. The more similar the intensities of the adjacent pixels are, the greater the term E_{smooth} is, and thus the less likely is the object boundary between considered pixels.

The data energy term $E_{data}(l)$ evaluates the likelihood of each pixel. In previous approaches for energy minimization based segmentation this term is often calculated using Gaussian Mixture Models [5, 24, 12] or non-parametrically on the basis of color histograms [9].

4.2. Object segmentation in video using graph cut

In this subsection we show an example of the simplest application of the energy minimization to object segmentation in video. The goal is to segment the object from the background in a given image sequence. The user places the object seeds and background seeds to construct object/foreground models and to define areas that should be separated by the segmentation in the next frames. The background seeds should provide hints for the algorithm on what is not an object of interest. Using the technique discussed in Section 3.1, the person to be segmented has been extracted through only one click on the mouse, see also Fig. 1c. The interface also allows a user to switch to pixel mode and to enter seeds through a brush controlled via a mouse.

To analyze the performance of the algorithm we used the same image sequence as in Section 3.2. Figure 2 shows some object silhouettes that were automatically extracted by the algorithm. The object/foreground models were constructed on the basis of histograms. Histograms represent distributions and provide invariance to variations of object appearance in images [28]. Color histograms are invariant to translation and rotation of the object and they vary slowly with the change of angle of view and with the change in scale. The invariance and descriptive power of histograms make them very useful in object extraction. Due to the statistical nature, a color histogram can only reflect the content of images in a limited way. Histogram based techniques are effective only when the number of bins can be kept relatively low and where sufficient data amounts are available. While global histograms [28] are not

well suited for complex scenes, a better approach consists in computing histograms over local image regions. The histogram representing the target can be accommodated over time using the past color distribution and newly extracted distribution from the extracted segment. In [12] color likelihoods were modeled by Gaussian Mixture Models in RGB color space. The foreground and background mixtures were learned via Expectation Maximization (EM). In later work [9] the authors came to the conclusion that non-parametric models based on histograms can be more practical in energy based segmentation, because the discriminative power of the GMM based likelihood ratios is eroded by local minima affect.

Using solely a fixed color histograms both for foreground and background we observed an improvement in segmentation results in comparison to results presented in Section 3.2. For object-id = #2 that was segmented in frames #700 - #1000 the average overlap between boxes is about 92%. Despite only coarse manual extraction of the person, see frame #700, the algorithm was capable of extracting the person's head in most of next frames. As demonstrated, for compact objects appearing as a single connected blob, the segmentation algorithm extracts silhouettes, which can be used in classification of object category. The algorithm runs in near real-time on the 850 MHz P III PC and it takes about 0.85 sec.

Figure 2. Segmented silhouettes of the person. Starting from frame #700 every 20-th frame is shown.

5. Video segmentation using graph cut built on superpixels

To improve the efficiency of the object segmentation in video we carry out graph cut built on superpixels. The superpixels are extracted by graph based algorithm with pairwise region comparison. A graph for energy based segmentation is built on extracted superpixels, instead of image pixels. Thus, the optimization is performed on a simpler graph. The number of nodes and edges is typically reduced by more than 50 times in comparison to pixel based relevant method. For example, the algorithm determined 347 superpixels on `Foreman` image of size 174×144, see Fig. 3a. The mentioned image segmentation has been achieved for $\kappa = 5$, eq. (4). The computation time on 850 MHz P III PC is about 0.25 sec. We can observe that the algorithm identifies object boundaries quite well.

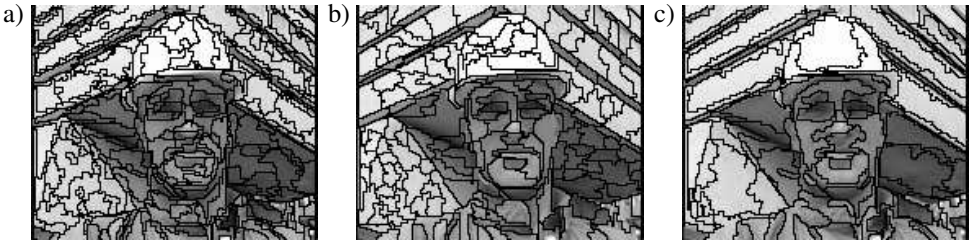

Figure 3. Segmentation of the image using pairwise region comparison for $\kappa = 5$ (a), $\kappa = 50$ (b), $\kappa = 300$ (c). The input image is depicted in Fig. 4a.

In an optional form of the object segmentation algorithm we can employ superpixels, which are computed by an algorithm identifying the watershed regions. An explanation of the morphological watershed transform can be found in [2]. The watershed algorithm [30] identifies object boundaries well and produces segments of relatively small size. The number of nodes and edges is typically reduced by more than 20 times in comparison to pixel based method. For example, the algorithm [30] determined 3440 segments on gray image of size 384×288 from Fig. 1 and 1090 segments in the first frame of the `Foreman` sequence of images of size 176×144, see also Fig. 4b. Applying our C/C++ implementation of the mentioned algorithm to `Foreman` image we achieved the segmentation in about 0.1 sec. on 850 MHz P III PC. We experimented also with the algorithm [22] for extraction of superpixels. The algorithm employs the normalized cuts [25], which use spectral clustering to exploit pairwise brightness, color and texture affinities between pixels. The normalized cut was employed to oversegment images and obtain superpixels. To enforce locality we utilized local connections only in the pairwise affinity matrix. Figure 4c depicts 350 superpixel boundaries overlaid on the image. The algorithm respects natural object boundaries very well. However, it requires considerable amount of memory and computational cost is substantially larger in comparison to pre-segmentation algorithm based on pairwise region comparison.

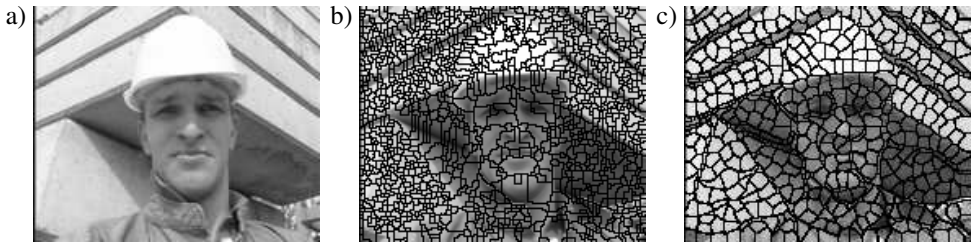

Figure 4. Superpixels obtained via watershed and normalized cuts, respectively. `Foreman` image (a), watershed based superpixels (b), normalized cut based superpixels (c).

Clustering is another technique that is widely used in image segmentation. This technique employs the fact that colors in an image tend to form clusters. Each pixel in the image is assigned to the cluster that is closest to the pixel color. We use clustering to calculate the likelihood energy in data term E_{data}. Given the superpixels that were manually separated into foreground and background in the initialization stage, the mean colors for each segment are computed. Then the clustering takes place separately for foreground and background. As a result we obtain cluster centroid locations $\{P_n^{\mathcal{F}}\}_{n=1}^{N}$ and $\{P_n^{\mathcal{B}}\}_{m=1}^{M}$, where N and M stand for the number of foreground and background clusters, respectively. At run-time, for each graph node i we compute the minimum distance from its mean color C_i to foreground and backgrounds clusters as follows: $d_i^{\mathcal{F}} = \arg\min_n \|C_i - P_n^{\mathcal{F}}\|$, $d_i^{\mathcal{B}} = \arg\min_m \|C_i - P_m^{\mathcal{B}}\|$. Such values are utilized in computation of the energy given by (5). Many different clustering algorithms exist in the present. In our algorithm we employ the well-known k-means algorithm because it acknowledged large usefulness in image segmentation and quantization [21]. The determination of the initial cluster centers plays a crucial role because the better the initial partition of data is, the faster the clustering will converge.

In practice, the graph cut built on superpixels can yield relatively good object segmentation. It takes the advantages of local color consistency, relations among neighboring segments and the benefits of global view at the object being segmented. However, like pixel based segmentation algorithms it can over-segment or under-segment the object of interest, see Fig. 5b and c, respectively. This effect inclined

the authors of work [9] to utilize a classifier that has been learned in advance at ground-truth data and then applied to discriminate between motion and non-motion. We implemented this algorithm and noticed that despite time-consuming training and parameter tuning, the mentioned algorithm in our implementation also under-segments or/and over-segments images in video. The results depicted in Fig. 5 were achieved for $N = 10$, $M = 50$ and $\kappa = 100$. For the considered `Foreman` video the watershed based segmentation algorithm yielded a far worse delineation of the face. In particular, the delineation between the face and the helmet was error-prone because of shared segments, see also Fig. 4b.

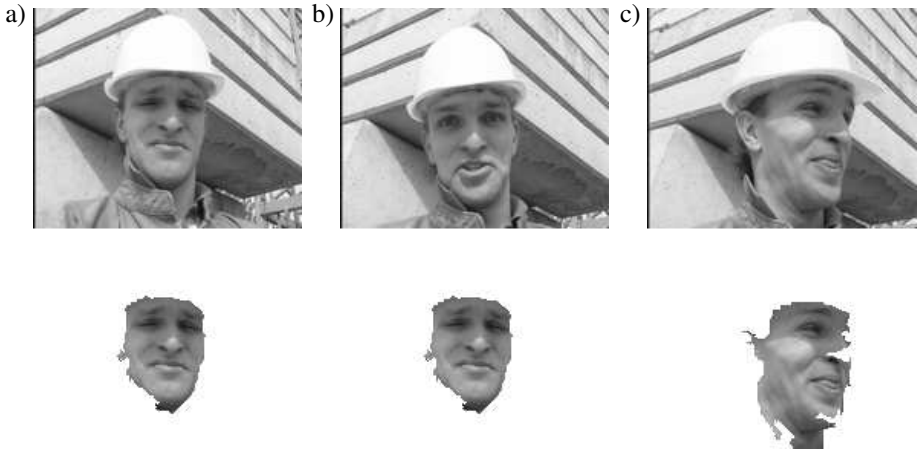

Figure 5. Video segmentation using graph cut built on superpixels. Top row: input images #31 (a), #83 (b), #108 (c). Bottom row: the segmented face.

The final algorithm for object segmentation in video employs keypoint tracking and is built on superpixels that are extracted through pairwise region comparison. Through tracking of object keypoints we identify regions likely to be parts of the object of interest. The temporal correspondence between the keypoints improves temporal coherence between segmented images. Given such regions a distance function is constructed, which serves as object prior. Using it we modify the data term E_{data}. The object seed for the graph cut is constructed as overlap of object region that was determined on the basis of the tracked features and the object mask, shifted into the predicted object location. The prediction of location of the mask is done on the basis of the object motion. It is estimated using the mean locations of features with correct inter-frame correspondences. Through analysis of the tracked features we perform a post-processing. Taking into account the number of the features on a superpixel, which has not been joined to the object by graph cut based segmentation, we can simply attach the superpixel to the object. In case of doubts we can increase the likelihood of the superpixel in question and then run the optimization once again, or even we can perform several calls as in [24] using the pre-segmentation feedback.

Some segmentation results of our method are depicted in Fig. 6. Owing to enhanced segmentation performance it is possible to reliably segment people, even if images are taken by a moving camera, see Fig 5. In this experiment the number of foreground clusters N was set to 20, and the segmentation was done using 2000 features. The remaining parameters were the same as previously. To cope with varying object appearance we extended the feature set about features from segments that were added by graph cut segmentation algorithm. At this stage we rely on the ability of the graph cut based segmentation to join small internal segments to the object.

Figure 6. Object segmentation using keypoint tracking and graph cut built on superpixels. Image #31 (a), #83 (b), #108 (c).

The algorithm has been tested on several available video sequences like: Carphone, Akijo, News. The graph cut segmentation algorithms built on pixels have inclination to over segmenting the videos. Such algorithms classify the whole parts of the background as foreground. This can occur because pixels are not natural entities. During analysis at superpixel level we can consider the tracked keypoints more effectively, take into account the geometrical relations between superpixels, objet mask from the previous frame, object prior, etc. In consequence, the over-segmentations are relatively small, comparable with errors that produces the method [9]. A drawback of our method is that the outline of the delineated object shape is not smooth.

6. Segmented shape based object recognition

The segmented shapes of the object can be utilized in object recognition through shape matching. Figure 7 depicts some object recognition results that were obtained through shape context based recognition algorithm [1]. The measurement of shape similarity is preceded by searching for correspondences between points on two shapes, and then employing the established correspondences to estimate an aligning transform. Figure 7a shows the segmented object shape in frame #700, see Fig. 2, the next image depicts a template, Fig. 7c shows the segmented shape after warping into the template, whereas the last image illustrates errors of warping. The local sum of square differences (SSD) was equal 0.07 in this illustrative experiment. The shapes were represented by 100 points sampled from contours.

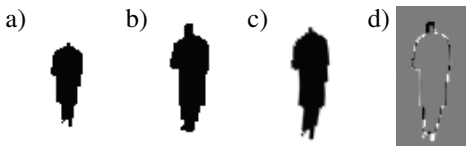

Figure 7. Shape matching and object recognition. The segmented shape (a), the template (b), the warped object shape (c), the errors of warping (d).

Another way to characterize how well the template would match to object edges at particular location in the image is by means of Chamfer distance. Given the object and template edges, see Fig. 8a, b, the distance transform is determined, see Fig. 8c. The pixel value in a distance transform image is proportional to the distance of that pixel to the edge pixel closest to it. The model is then shifted over

the distance transform image and at each shift position, the sum of distances at the model pixels is determined, and lastly the shift position producing the smallest sum is chosen as the best-match position of the model in the image, see Fig. 8d depicting the best-match position as the white cell in the matching score of size 17×11. Chamfer matching works well when the model and the image do not have considerable rotation and scaling differences.

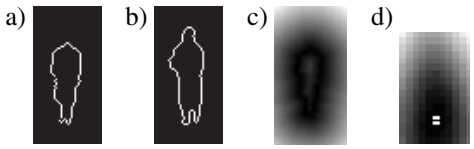

Figure 8. Shape matching using Chamfer distance. Edge image of the object (a) and the template (b), distance transform (c), best-match position marked as the white cell in the map of matching errors (d).

7. Conclusions

It this paper, a novel video segmentation scheme has been proposed. The graph cut is built on superpixels, instead of the image pixels. The superpixels are extracted using pairwise region comparison. The tracked object keypoints support segmenting the entire object in image sequences. On the basis of the tracked object keypoints we pre-segment the object. Then using such a pre-segmentation we determine a distance function, which serves as object prior in the graph cut based final delineation of the object. The object to be segmented in a sequence of images is specified manually through activating the superpixels in an initialization stage. Using the proposed algorithm, the objects are segmented correctly as complete entities, despite the high variability of the object shape and cluttered background. The algorithm segments the object in video reliably, ensures high consistency of segmentations over time, requires no extensive training, and runs in real-time on modern PC computers. We showed that the delineated shapes can be useful in object recognition via shape matching.

References

[1] Belongie, S., Malik, J., Puzicha, J.: Schape matching and object recognition using shape contexts, *IEEE Trans. on Pattern Analysis and Machine Intelligence*, **24**(24), 2002, 509–522.

[2] Beucher, S., Meyer, F.: *The morphological approach to segmentation: The watershed transformation*, Mathematical Morphology in Image Processing, 1993, 443–481.

[3] Biederman, I.: Recognition by components: A theory of human image understanding, *Psychological Review*, **94**, 1987, 115–147.

[4] Black, M.: Combining intensity and motion for incremental segmentation and tracking, *European Conference on Computer Vision*, 1992.

[5] Boykov, Y., Jolly, M.: Interactive graph cuts for optimal boundary and region segmentation of objects in n-d images, *Proc. of ICCV*, 2001.

[6] Boykov, Y., Veksler, O., Zabih, R.: Fast approximate energy minimization via graph cuts, *IEEE Trans. on Pattern Analysis and Machine Intelligence*, **23**(11), 2001, 1222–1239.

[7] Cave, C. B., Kosslyn, S. M.: The role of parts and spatial relations in object identification, *Perception*, **22**, 1993, 229–248.

[8] Cormen, T., Leiserson, C., Rivest, R.: *Introduction to algorithms*, MIT Press, 1990.

[9] Criminisi, A., Gross, G., Blake, A., Kolmogorov, V.: Bilayer segmentation of live video, *Proc. of the IEEE Computer Society Conf. on Computer Vision and Pattern Recognition*, 2006.

[10] Felzenszwalb, P. F., Huttenlocher, D. P.: Efficient graph-based image segmentation, *International Journal of Computer Vision*, **59**(2), 2004, 167–181.

[11] Juan, O., Boykov, Y.: Active graph cuts, *Proc. of CVPR*, 2006.

[12] Kolmogorov, V., Criminisi, A., Blake, A., Ross, G., Rother, C.: Bi-layer segmentation of binocular stereo video, *Proc. of CVPR*, 2005.

[13] Kolmogorov, V., Zabih, R.: What energy functions can be minimized via graph cuts?, *Proc. of European Conf. on Computer Vision*, 2002.

[14] Li, S.: *Markov Random Field modeling in computer vision*, Springer-Verlag, 1995.

[15] Lucas, B., Kanade, T.: An iterative image registration technique with an application to stereo vision, *Proc. Seventh Int. Joint Conf. on Artificial Intelligence*, Vancouver, Canada, 1981.

[16] Luck, S. J., Vogel, E. K.: The capacity of visual working memory for features and conjuctions, *Nature*, **390**(20), 1997, 279–281.

[17] Marr, D.: *Vision: a computational investigation into the human representation and processing of visual information*, W. Freeman, San Francisco, CA, 1982.

[18] Ogiela, L., Tadeusiewicz, R., Ogiela, M.: Cognitive informatics in automatic pattern understanding, *In: D. Zhang, Y. Wang, W. Kisner, Proc. the 6th IEEE Int. Conf. Cognitive Informatics*, 2007.

[19] Ohta, Y., Kanade, T., Sakai, T.: Structural method for obstacle detection and terrain classification, *Proc. of Int. Joint Conf. on Pattern Recognition*, 1978.

[20] Ommer, B., Buhmann, J. M.: Learning compositional categorization models, *ECCV*, 2006.

[21] Plataniotis, K., Venetsanopoulos, A.: *Color image processing and applications*, Springer, New York, Heidelberg, 2000.

[22] Ren, X., Malik, J.: Learning a classification model for segmentation, *In Proc. 9th Int. Conf. Computer Vision*, 2003.

[23] Rosch, E., Mervis, C. B., Gray, W. D., Johnson, D. M., Boyes-Braem, P.: Basic objects in natural categories, *Cognitive Psychology*, **8**, 1976, 382–439.

[24] Rother, C., Blake, A., Kolmogorov, V.: Grabcut - interactive foreground extraction using iterated graph cuts, *Proc. of ACM SIGGRAPH*, 2004.

[25] Shi, J., Malik, J.: Normalized cuts and image segmentation, *IEEE Trans. on Pattern Analysis and Machine Intelligence*, **22**(8), 2000, 888–905.

[26] Shi, J., Tomasi, C.: Good features to track, *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition*, Seattle, WA, 1994.

[27] Sun, J., Zhang, W., Tang, X., Shum, H.-Y.: Background cut, *Proc. of European Conf. On Computer Vision*, 2006.

[28] Swain, M. J., Ballard, D. H.: Color indexing, *Int. Journal of Computer Vision*, **7**(1), 1991, 11–32.

[29] Ullmann, S.: Object recognition and segmentation by a fragment-based hierarchy, *Trends in Cognitive Sciences*, **11**(2), 2006, 58–64.

[30] Vincent, L., Soille, P.: Watersheds in digital spaces: An efficient algorithm based on immersion simulations, *IEEE Trans. on Pattern Analysis and Machine Intelligence*, **13**(6), 1991, 583–598.

[31] Wang, J., Bhat, P., Colburn, R. A., Argawala, M., Cohen, M. F.: Interactive video cutout, *Proc. of ACM SIGGRAPH*, 2005.

[32] Wang, Y.: The real-time process algebra (RTPA), *The Int. J. of Annals of Software Engineering*, **14**, 2002, 235–274.

[33] Wang, Y., Wang, Y.: On cognitive informatics models of the brain, *IEEE Transactions on Systems, Man, and Cybernetics*, **36**(2), 2006, 16–20.

[34] Xu, N., Bansal, R., Ahuja, N.: Object segmentation using graph cuts based active contours, *Proc. of CVPR*, 2003.