Robust Color Image Segmentation Through Tensor Voting∗

Rodrigo Moreno1, Miguel Angel Garcia2, and Domenec Puig1

1 Rovira i Virgili University, Intelligent Robotics and Computer Vision Group
Dept. of Computer Science and Mathematics, Av. Països Catalans 26, 43007 Tarragona, Spain
2 Autonomous University of Madrid, Dept. of Informatics Engineering
Francisco Tomas y Valiente 11, 28049 Madrid, Spain
{rodrigo.moreno,domenec.puig}@urv.cat, miguelangel.garcia@uam.es

Abstract

This paper presents a new method for robust color image segmentation based on tensor voting, a robust perceptual grouping technique used to extract salient information from noisy data. First, an adaptation of tensor voting to both image denoising and robust edge detection is applied. Second, pixels in the filtered image are classified into likely-homogeneous and likely-inhomogeneous by means of the edginess maps generated in the first step. Third, the likely-homogeneous pixels are segmented through an efficient graph-based segmenter. Finally, a modified version of the same graph-based segmenter is applied to the likely-inhomogeneous pixels in order to obtain the final segmentation. Experiments show that the proposed algorithm has a better performance than the state-of-the-art.

1. Introduction

Image segmentation is one of the most important stages in computer vision as a preliminary step towards further analysis and recognition stages. Its goal is to partition a given image into a set of non-overlapping homogeneous regions that likely correspond to the different objects or geometric structures that may be perceived in the scene. Although a huge amount of methods have been proposed, image segmentation is still under research since issues such as robustness or efficiency are far from being completely solved.

Region growing has been one of the most popular approaches for segmenting images. The basic idea of this approach is to grow a set of seed points by assimilating similar neighboring points. Despite its success in noiseless scenarios (e.g., [3]), methods based on this approach are usually unable to segment noisy images effectively. This is due to the difficulty to distinguish between edges and noise. However, robust edge detectors that partially get rid of this problem have recently been proposed (e.g., [6]).

On the other hand, soft color segmenters have been proven effective for complex scenarios [9]. These methods estimate for every pixel a degree of membership to every detected region in the image. However, in practice, these membership degrees only make sense for pixels close to borders. This suggests that hard segmentation (e.g., region growing) is suitable for most parts of the image, while pixels close to borders should be processed in a different way.

In this line, this paper proposes a color image segmentation technique that is robust to noise. The proposed method applies an efficient graph-based region growing strategy to the filtered image and edginess map obtained from the method proposed in [5, 6]. An edginess map can be seen as a measurement of how likely every pixel belongs to an edge. Although previous works have already combined region and edge information through a variety of methodologies [7], they usually fail in noisy scenarios, since traditional edge detectors are prone to erroneously detecting edges in noisy regions. Thus, the proposed method uses a robust edge detector in order to get rid of the problem.

The paper is organized as follows. Section 2 overviews the unified framework for both image denoising and edge detection proposed in [5, 6], which is the base of the segmentation algorithm proposed in this paper. Section 3 presents the proposed image segmentation method. Section 4 shows a comparison of the proposed algorithm with state-of-the-art methods. Finally, Sect. 5 makes some concluding remarks.

∗ This research has been partially supported by the Spanish Ministry of Science and Technology (project DPI2007-66556-C03-03), by the Commissioner for Universities and Research of the Government of Catalonia and by the European Social Fund.
2. Tensor Voting for Simultaneously Color Image Denoising and Edge Detection

In a first step, the color at every pixel is converted to the CIELAB space. Every CIELAB channel is then normalized in the range \([0, \pi/2]\). The method encodes the color, local uniformity and edginess of every pixel through three second order 2D tensors, one for each color channel. Figure 1 shows the graphical interpretation of a tensor for channel \(L\). Color, uniformity and edginess are encoded by means of \(\alpha\) and the normalized \(s_1 = (\lambda_1 - \lambda_2)/\lambda_1\) and \(s_2 = \lambda_2/\lambda_1\) saliences respectively, with \(\lambda_1\) and \(\lambda_2\) being the eigenvalues of the tensor. Since local uniformity and edginess are not available at the beginning of the process, these tensors are only initialized with color information.

In a second step, the information encoded in the tensors is propagated in a neighborhood through a convolution-like voting process. This step is independently applied to the tensors of every color channel. The voting process is carried out by means of two specially designed tensorial functions referred to as propagation functions, which take into account not only the information encoded in the tensors but also the local relations between neighbors. Two propagation functions are defined: a stick and a ball propagation function. The stick propagation function is used to propagate the most likely noiseless color of a pixel, while the ball propagation function is used to increase edginess where required. The stick and ball propagation functions from the voter pixel \(q\) to the votee pixel \(p\), \(S_c(p, q)\) and \(B_c(p, q)\), are given by:

\[
S_c(p, q) = F_{Sc}(p, q) e_{1c}(p) e_{1c}(p)^T, \tag{1}
\]

\[
B_c(p, q) = F_{Bc}(p, q) I, \tag{2}
\]

where \(e_{1c}(p)\) corresponds to the principal eigenvector of the tensor at \(p\) in channel \(c\), \(I\) is the identity matrix, and \(F_{Sc}\) and \(F_{Bc}\) are factors that modulate the strength of the votes for every color channel \(c\). These factors take into account not only the influence of the distance between the two pixels, but also their perceptual color difference, the local uniformity and their noisiness, which are derived from the tensors.

The voting process at every pixel is carried out by adding all the tensors propagated towards it from its neighbors by applying the above propagation functions. Thus, the total vote received at a pixel \(p\) for each color channel \(c\), \(T_c(p)\), is given by:

\[
T_c(p) = \sum_{q \in \text{neigh}(p)} S_c(p, q) + B_c(p, q).
\]

The voting process is applied twice. The first application is used to obtain an initial estimation of local uniformity and edginess. At the second application, the tensors are initialized with the tensors obtained after the first application.

The method yields two results: a filtered image, which is encoded through the \(\alpha\) angle of the tensors, and an edginess map that is calculated as the mean of \(s_2\) of the three tensors at every pixel. The results can be further improved in very noisy scenarios by iterating the following steps: (a) the above voting process is run, (b) every pixel’s color is replaced by the filtered image.

3. Robust Color Image Segmentation

Although region growing approaches usually have a good performance in homogeneous regions, they are also prone to errors mainly attributed to erroneous processing of inhomogeneous regions. For example, using pixels at borders as growing seeds is inconvenient, since they can lead to dummy regions generated from similar pixels close to borders [7]. The proposed method is based on the fact that pixels within homogeneous regions and pixels at borders are different in nature. Thus, they must be processed differently.

The proposed method comprises four stages. At a first stage, a graph is created in a similar way as in [3] from the filtered image processed as described in Section 2. Afterwards, pixels are classified into likely-homogeneous and likely-inhomogeneous with the help of the edginess map calculated as described in Section 2. Likely-homogeneous pixels are clustered with the graph-based segmenter of [3] in a third stage. In the final stage, likely-inhomogeneous pixels are aggregated to the presegmented image through a modified version of the same segmenter used in the third stage. These stages are explained in more detail below.

Every pixel corresponds to a vertex in the graph created at the first stage, while every pair of neighboring pixels leads to an edge whose weight is given by the color difference between those pixels. In addition, local edginess is also stored at every edge of the graph.

Likely-inhomogeneous pixels can be extracted by thresholding the edginess map. This procedure has the
advantage that not only edges are classified as likely-inhomogeneous, but also a strip of pixels at both sides of edges. Previous region growing methods are particularly prone to errors in those strips. Edginess maps obtained through robust techniques, such as tensor voting, are required for this stage, since they are the only way to correctly classify noisy pixels as likely-homogeneous in noisy scenarios.

The third and fourth stages of the proposed method are based on the graph-based segmenter described in [3], which can be summarized as follows. Let $H(C)$ be the internal difference of region $C$, defined as the largest weight of the edges in its minimum spanning tree, and $\tau(C) = k/|C|$, where $k$ is a parameter. The segmenter follows the next stages: (a) the graph edges are sorted in ascending order of weight; (b) a different region is created for every vertex in the graph; (c) for each sorted edge $E$, the following region growing strategy is applied: let $C_i$ and $C_j$ be the neighboring regions connected in the graph by means of $E$ and $w$ its weight. If $w \leq D(C_i, C_j)$, with:

$$D(C_i, C_j) = \min(H(C_i) + \tau(C_i), H(C_j) + \tau(C_j)),$$

(3)

both regions are merged and $H(C_i \cup C_j)$ is updated with $w$ in such a case.

In the third stage of the method, this graph-based segmenter is applied to the edges of the graph that do not connect likely-inhomogeneous pixels. After that, post-processing can be optionally applied in order to remove small islands surrounded by big regions.

A soft segmentation can be obtained by calculating a membership degree from the distance between every likely-inhomogeneous pixel to its nearest regions obtained after the third stage. However, a hard segmentation can also be obtained by applying the fourth stage of the method.

Finally, the fourth stage joins likely-inhomogeneous pixels to the most likely region obtained in the third stage through the same segmenter applied with the following four variations. First, the remaining edges of the graph are sorted by edginess instead of weight, since the former is more relevant than the latter for inhomogeneous pixels. However, weight is still used to merge regions, since it encodes similarity among pixels. Second, a pixel must be joined to a preexistent region in order to avoid the creation of dummy regions. Third, function $H(C)$ is not updated when a pixel is joined to region $C$ in order to give more relevance to the similarity of neighboring pixels than of neighboring regions, which is essential in the third stage. Finally, the process is iterated until both all the graph’s edges and all the likely-inhomogeneous pixels have been processed.

### Table 1. Average performance measurements for the 15 tested images

<table>
<thead>
<tr>
<th>$\sigma$</th>
<th>Metric</th>
<th>MS</th>
<th>FH</th>
<th>GC</th>
<th>TBES</th>
<th>RIS</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>PRI</td>
<td>0.73</td>
<td>0.75</td>
<td>0.73</td>
<td>0.77</td>
<td><strong>0.78</strong></td>
</tr>
<tr>
<td></td>
<td>GCE</td>
<td>0.12</td>
<td>0.20</td>
<td>0.22</td>
<td>0.31</td>
<td>0.16</td>
</tr>
<tr>
<td>30</td>
<td>PRI</td>
<td>0.54</td>
<td>0.57</td>
<td>0.74</td>
<td><strong>0.79</strong></td>
<td><strong>0.79</strong></td>
</tr>
<tr>
<td></td>
<td>GCE</td>
<td>0.31</td>
<td><strong>0.11</strong></td>
<td>0.22</td>
<td>0.32</td>
<td>0.17</td>
</tr>
</tbody>
</table>

### 4. Experimental Results

Fifteen images from the Berkeley segmentation data set [4] and their corresponding ground truths have been used in the experiments. Mean shift (MS) [1], the method in [3], referred to as FH, Graclus (GC) [2], and the method in [8], referred to as TBES, have been used in the comparisons, since they are representative of the state-of-the-art in color image segmentation. The default parameters of MS (undersegmentation), FH, TBES and the method in [5, 6] have been used. GC has been run with 20 clusters, as recommended in [4]. Parameter $k$ has been experimentally set to 800, and edginess maps have been thresholded for values greater than 20% of the maximum for the proposed method, referred to as RIS. The methods have been applied to noiseless and noisy images contaminated with Gaussian noise with a standard deviation $\sigma$ of 30.

In addition to visual inspection, performance has been evaluated by means of two metrics: the probabilistic Rand index (PRI) [10] (higher values are better) and the global consistency error (GCE) [4] (lower values are better). Table 1 shows that GC, TBES and RIS are robust since they obtain similar scores for both noiseless ($\sigma = 0$) and noisy ($\sigma = 30$) images. In addition the proposed method has the best performance in PRI in both scenarios when compared to the other methods and it is competitive in GCE.

Figure 2 shows the segmentation results for four noisy images. MS and FH have a poor performance, since they mistakenly oversegment some areas (e.g., the grass) and undersegment others (e.g., MS is unable to segment the snake). GC yields better results but its performance depends on the number of regions in the image, which is usually unknown a priori. TBES tends to mistakenly undersegment the images (e.g., it integrates the wolf with the background and is unable to segment the snake). Hence, RIS succeeds in attaining a good balance between under and oversegmentation.

---

1. Full-resolution images are available at [http://deim.urv.cat/~rivi/ris.html](http://deim.urv.cat/~rivi/ris.html)
Regarding computational cost, MS and FH were the fastest of all tested algorithms when run on an Intel Core 2 Quad Q6600 with 4GB RAM (less than 100ms), GC took around 80 seconds, and TBES was the slowest with 22 minutes and 35 seconds for every image. The efficiency of RIS is mainly determined by the method in [5, 6] (around 40 seconds), since the segmentation step of RIS took less than 100ms.

5. Concluding Remarks

A new method for robust color image segmentation has been presented. The tensors obtained by applying the method in [5, 6] are used to classify pixels into likely-homogeneous and likely-inhomogeneous. Those pixels are then segmented through a variation of the method proposed in [3]. Experiments show that the proposed algorithm is especially effective for very noisy images when compared to the methods in [1, 2, 3, 8] at a competitive computational cost. Results also suggest that robustness can be attained by using texture features or by using robust techniques, such as tensor voting. Future work includes extending this work to multiscale color image segmentation.

References