Interactive Image Segmentation Based on Synthetic Graph Coordinates

Costas Panagiotakis[a,*], Harris Papadakis[b], Elias Grinias[d], Nikos Komodakis[c], Paraskevi Fragopoulou[b,1], Georgios Tziritas[c,*]

[a]*Department of Commerce and Marketing, Technological Educational Institute of Crete, 72000 Ierapetra, Crete, Greece*
[b]*Department of Applied Informatics and Multimedia, Technological Educational Institute of Crete, Heraklion, Crete, Greece, PO Box 140*
[c]*Computer Science Department, University of Crete, Greece, P.O. Box 2208*
[d]*Department of Geoinformatics and Surveying, Technological Educational Institute of Serres, 62124 Serres, Greece*

Abstract

In this paper, we propose a framework for interactive image segmentation. The goal of interactive image segmentation is to classify the image pixels into foreground and background classes, when some foreground and background markers are given. The proposed method minimizes a min-max Bayesian criterion that has been successfully used on image segmentation problem and it consists of several steps in order to take into account visual information as well as the given markers, without any requirement of training. First, we partition the image into contiguous and perceptually similar regions (superpixels). Then, we construct a weighted graph that represents the superpixels and the connections between them. An efficient algorithm for graph clustering based on synthetic coordinates is used yielding an initial map of classified pixels. This method reduces the problem of graph clustering to the simpler problem of point clustering, instead of solving the problem on the graph data structure, as most of the known algorithms from literature do. Fi-

*Corresponding author

Email addresses: `cpanag@staff.teicrete.gr` (Costas Panagiotakis), `adanar@epp.teicrete.gr` (Harris Papadakis), `elgrinias@gmail.com` (Elias Grinias), `komod@csd.uoc.gr` (Nikos Komodakis), `fragopou@ics.forth.gr` (Paraskevi Fragopoulou), `tziritas@csd.uoc.gr` (Georgios Tziritas)

[1]Paraskevi Fragopoulou is also with the Foundation for Research and Technology-Hellas, Institute of Computer Science, 70013 Heraklion, Crete, Greece.

nally, having available the data modeling and the initial map of classified pixels, we use a Markov Random Field (MRF) model or a flooding algorithm to get the image segmentation by minimizing a min-max Bayesian criterion. Experimental results and comparisons with other methods from the literature are presented on LHI, Gulshan and Zhao datasets, demonstrating the high performance and accuracy of the proposed scheme.

Keywords: image segmentation, interactive image segmentation, network coordinates, community detection, Markov Random Field

1. Introduction

Image segmentation is a key step in many image-video analysis and multimedia applications. Segmentation of color textured images has become a necessity for many applications, such as content based image retrieval, object recognition-tracking [1] and in 3D modeling [2].

The objective of image segmentation is to extract the foreground objects out of the cluttered background. Typically, it is used to partition images into regions that are in some sense homogeneous, or have some semantic significance, thus providing high level information about scenes to subsequent processing stages [3]. However, the automatic segmentation problem is ill-posed and complex, as more than one partition can satisfy generic objective criteria and can be accepted by human experts. Thus, despite the plethora of methodologies for image segmentation, a general-purpose algorithm that addresses the whole range of segmentation problems and applications does not exist.

According to interactive image segmentation, which is a special case of image segmentation, unambiguous solutions, or segmentations satisfying subjective criteria, could be obtained, since the user gives some markers on the regions of interest and on the background. Thus, under interactive segmentation, the semantic objects are extracted with high accuracy, since the high-level information that is needed to traverse the "semantic-gap" between homogeneous regions and perceived objects is provided by the given markers. Fig. 1 illustrates an example

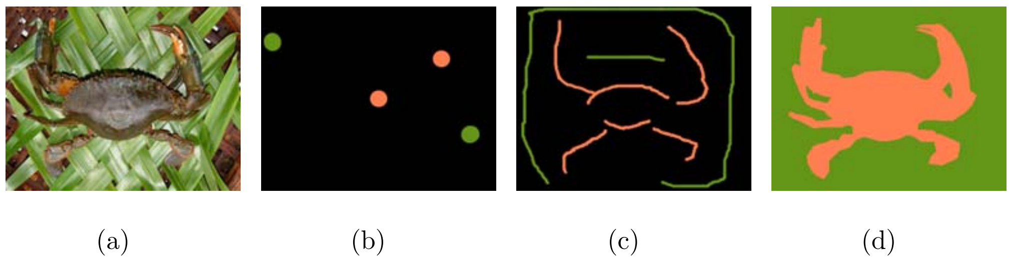


Figure 1: **(a)** Original image, **(b), (c)** given markers (point and scribble) and **(d)** the ground truth image.

of an original image, two types of markers and the segmentation ground truth. Some algorithms are also sensitive on the shape of the given markers that usually are points (Fig. 1(b)) or scribbles as in Fig. 1(c).

Automatic and interactive image segmentation have been widely studied in the literature. Most of image segmentation algorithms satisfy a subset of the following constraints yielding valid segmentations [1]:

- *Edge constraints* that refer to correlation between the segmentation boundaries and the image edges. In [4], the detection of edges and boundaries is done using features that correspond to characteristic changes in brightness and texture associated with natural boundaries, combined with a classifier that is trained using human labeled images as ground truth. In [5], the proposed segmentation algorithm consists of generic machinery for transforming the output of any contour detector into a hierarchical region tree, reducing the problem of image segmentation to that of contour detection.

- *Shape constraints* lead to boundary regularization, while hard shape conditions, assuming prior knowledge, could address specific rigid or deformable object localization. In [6], a variational level set framework for segmentation and tracking of the Left Ventricle has been proposed that can account for global shape consistency as well as for local deformations.

- *Region constraints* refer to pixel grouping according to class properties and similarity criteria. The objective is to obtain regions that are uniform and homogeneous with respect to the selected features. This is one of the most significant constraints that should be satisfied, especially by the interactive image segmentation algorithms that have a-priory knowledge about the classes [7].

- *Topology constraints* should limit the number of connected components. In [8], a new metric has been proposed that tolerates disagreements over boundary location, penalizes topological disagreements and can be used directly as a cost function for learning boundary detection.

During the last decade, a large number of interactive image segmentation algorithms have been proposed in the literature. Hereafter, we briefly present some popular interactive image segmentation algorithms. In [9] hard topological constraints are used to reduce the search space of feasible segmentations minimizing an energy function via max-flow/mincut algorithms. In [10], weighted geodesic distances to user-provided scribbles are computed, from which the image is automatically segmented. The weights are based on spatial and/or temporal gradients, considering the statistics of the pixels scribbled by the user. In [11], a new shape constraint based method for interactive image segmentation has been proposed using Geodesic paths. The authors introduce Geodesic Forests, which exploit the structure of shortest paths in implementing extended constraints. The system is evaluated by means of a “robot user” to measure the amount of interaction required in a precise way.

In [12], discriminative learning methods have been used to train conditional models for both region and boundary based on interactive scribbles. In the region model, the authors use two types of local histograms (histogram of color (HoC) and histogram of oriented gradients (HoG) [13]) with different window sizes to characterize local image statistics around a specific pixel. In the boundary model, the authors use 12-bin boundary features by applying gradient filters (4 different scales) to each color component. In [2], interactive segmentation has been used for 3D modeling of urban buildings. Architectural elements (e.g. walls and windows) are detected using an interactive segmentation algorithm based on Gaussian mixture models and Expectation Maximization methods.

In [14], a two step segmentation algorithm has been proposed that first obtains a binary segmentation and then applies matting on the border regions to obtain a smooth alpha channel. The proposed segmentation algorithm is based on the minimization of the Geodesic Active Contour energy. In [15], a Level set approach for interactive segmentation has been proposed that has been tested on grayscale images. In [16], a Bayesian network (BN) model for both automatic and interactive image segmentation has been proposed. A multilayer BN is constructed from an oversegmentation to model the statistical dependencies among superpixel regions, edge segments, vertices, and their measurements. Image segmentation is generated by the most probable explanation inference of the true states of both region and edge nodes from the updated BN.

According to the interactive segmentation algorithm proposed in [7], first, all the labeled seeds are independently propagated for obtaining homogeneous connected components for each of them. The given scribbles should be of approximately circular shape. Then, the image is divided into blocks which are classified according to their probabilistic distance from the classified regions, and a topographic surface for each class is obtained. Finally, two algorithms (MRF and priority multi-label flooding) for regularized classification based on the topographic surface have been proposed. In this research, we have used as well the MRF and priority multi-label flooding algorithms to classify the initial map.

Most of the approaches from the literature are heuristic, or they try to optimize a criterion that may not be appropriate for interactive segmentation. Furthermore, some of them they need specific shape of initial markers, or they require a training set. The proposed method consists of several steps taking into account visual information as well as the given markers, without the requirement of training. Initially, we construct a weighted graph that represents the superpixels and the connections between them. The proposed method solves the graph clustering problem using the synthetic network coordinates that are automatically estimated by a distributed

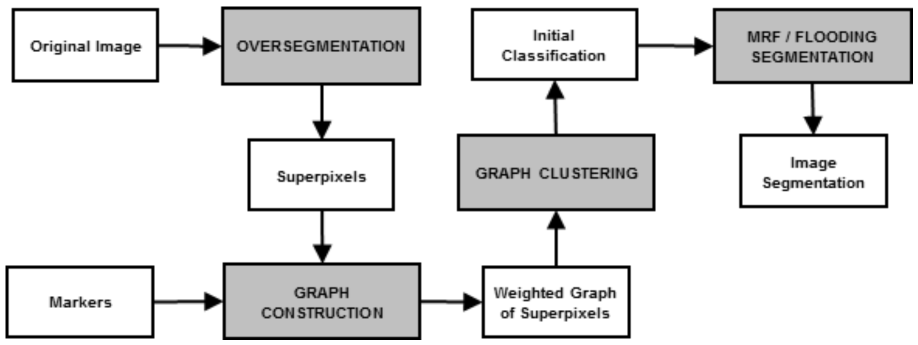


Figure 2: Scheme of the proposed system architecture.

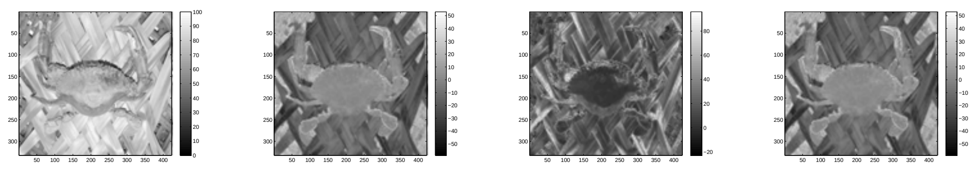
algorithm based on interactions between neighboring nodes. The use of network coordinates reduces the problem of graph clustering to the simpler problem of point clustering, instead of solving the problem on the graph data structure, as most of the known algorithms from literature do. Finally, the image segmentation is provided by a Markov Random Field (MRF) model or a flooding algorithm minimizing a min-max Bayesian criterion. This criterion has been also successfully used in image segmentation problem [7, 1], resulting in higher performance results when compared to other methods from literature. In addition, in this work there is no constraint on the shape of initial markers.

The rest of the paper is organized as follows: Section 2 gives the definition of the interactive segmentation problem and the proposed visual modeling. Sections 3 and 4 present the proposed algorithm. The experimental results are given in Section 6. Finally, conclusions and discussion are provided in Section 7.

2. Problem Definition

2.1. Overview

The problem of interactive image segmentation that we tackle in this research is defined as follows: The user provides markers on the regions of interest (foreground) and on the background. Taking into account the initial markers, the goal is to classify the rest of image pixels into foreground and background classes.



(a) (b) (c) (d)

Figure 3: The description of visual information of Fig. 1(a): **(a)**, **(b)**, **(c)** The three Lab color components and **(d)** the textureness.

Similarly with the interactive segmentation algorithm proposed in [7], the proposed method can be divided into two steps:

- In the first step (see Section 3), we partition the image into superpixels using the oversegmentation algorithm proposed in [17]. Then, we construct a weighted graph that represents the superpixels and the connections between them, taking into account the given markers and visual information. An initial map of classified pixels is provided by an efficient algorithm for graph clustering based on synthetic coordinates.
- In the second step (see Section 4), the image segmentation is provided by a Markov Random Field (MRF) model or a flooding algorithm. These methods have been proposed in [7].

Fig. 2 illustrates the scheme of the proposed system architecture. Due to this methodology, high performance results are obtained, without applying any constraint on the markers' shape, number and size (see Fig. 1). Hereafter, we present in detail the proposed methodology.

2.2. Visual information

In this work, the description of visual content consists of Lab color components for color distribution and textureness for texture content. This approach has been also used in [1]. In order to compute textureness [1], we first obtain the principal image component Y from the three RGB components. Let Y_m be the output of a

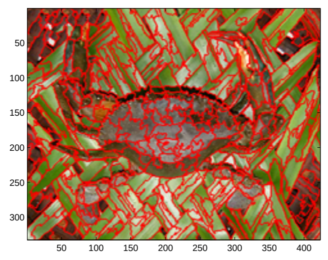


Figure 4: Estimated superpixels for the image of Fig. 1(a).

median filter, in our implementation of size 5×5. The textureness is measured by applying again the same median filter on $|Y - Y_m|$. Therefore, at every image pixel, in addition to the three Lab color components, we have a measure of textureness that is sensitive to local high-frequency content, while respecting strong edges. Fig. 3 depicts the four image components that have been used to describe the visual information of Fig. 1(a).

3. Coarse Segmentation

3.1. Oversegmentation algorithm

Initially, we partition the image into superpixels using the oversegmentation algorithm proposed in [17]. This method robustly segments the given image into small contiguous and perceptually similar (homogeneous) regions, called superpixels. In the next steps of our method, we have used superpixels instead of pixels, since the number of superpixels is sufficiently lower (about 100 times lower) than the number of pixels and each superpixel belongs either to the foreground or the background class. Let $SP = \{s_i, i \in \{1, ..., N\}\}$ denote the set of N superpixels. Fig. 4 illustrates the estimated set SP produced by the oversegmentation algorithm proposed in [17], for the image of Fig. 1(a).

This initialization is actually the initial step of the proposed algorithm. However, in the fine segmentation step (see Section 4), we relax the constraint that each superpixel should belong to either the foreground or the background classes, by performing a fine, pixel-level segmentation. Copying with the case that a superpixel contains both a foreground and a background marker, (which is almost impossible to happen since superpixels are perceptually homogeneous regions with small area), a solution is to split the superpixel under consideration into two regions according to the given markers.

3.2. Graph of superpixels

In the next step, we construct a graph G that represents the superpixels and the connections between them, taking into account the given markers and visual information. According to the given markers, two superpixels can either be connected, meaning that they belong to the same class or be disconnected, meaning that they belong to different classes. Thus, the nodes (superpixels) in this graph are connected with edges of two types in order to take into account the two types of relations between superpixels:

- the E_C edges that connect two superpixels belonging to the same class.
- the E_D edges that connect two superpixels belonging to different classes.

In the second step of the algorithm, the visual information and the superpixels' proximity is used to create the set of edges E_C until G becomes a connected graph. The visual distance $d_v(s_i, s_j)$ between two superpixels s_i and s_j, $i,\ j \in \{1, ..., N\}$ is given by the Mallows distance [18] of the three color components in Lab color space and for the textureness measure of the corresponding superpixels (see Equation 1). The distance is efficiently computed under the assumption of independent data components. For each component, data are sorted and the mean absolute difference is used to measure the distance [1, 19]. Let $V_k(s_i)$, $k \in \{1, 2, 3, 4\}$ be the corresponding sorted vector of the k-th component. The first three components denote the three color components (Lab) and the last corresponds to textureness. The superpixel that has the least number of pixels is oversampled, so that the two vectors $V_k(s_i)$ and $V_k(s_j)$ end up with the same number of elements. In the special case that a superpixel has more elements than a constant threshold (e.g.

500), we perform subsampling on the corresponding vector, in order to maintain the computational cost of $O(1)$ for the estimation of $d_v(s_i, s_j)$. Without loss of generality, let s_j be the superpixel with the least number of pixels. Then, the visual distance $d_v(s_i, s_j)$ between two superpixels s_i and s_j is given by the following Equation:

$$d_v(s_i, s_j) = \frac{1}{4 \cdot |s_i|} \sum_{k=1}^{4} ||V_k(s_i) - V_k(s_j)||_1, \tag{1}$$

where $|s_i|$ denotes the number of pixels of s_i and $||.||_1$ denotes the Manhattan (L_1) norm.

Let G' be the weighted graph of superpixels, so that two superpixels s_i and s_j are connected with an edge of weight $d_v(s_i, s_j)$ if and only if they are neighbors, meaning that they share a common boundary. Then, the proximity distance $d_p(s_i, s_j)$ between superpixels s_i and s_j is given by the length of the shortest path from s_i to s_j in graph G'. Since $d_v(s_i, s_j)$ is a metric using the L_1 norm, due to the triangular inequality, it holds that $d_v(s_i, s_j) \leq d_p(s_i, s_j)$. Using Johnson's algorithm [20], it holds that the computational cost to compute all shortest paths on G' is $O(N \cdot \log(N) + N \cdot E)$ which is simplified to $O(N^2)$, since the number of edges (E) of the graph of superpixels is $O(N)$.

The proposed distance between superpixels s_i and s_j that efficiently combines the visual and proximity distances is given by Equation 2:

$$d(s_i, s_j) = \sqrt{d_p(s_i, s_j)} \cdot d_v(s_i, s_j) \tag{2}$$

The use of the square root on the proximity distance is explained by the fact that the visual distance is more important than the proximity distance. The product is selected because $d(s_i, s_j)$ should get high values only if both of the distances get high values. In addition, $d_p(s_i, s_j)$ also depends on the geometric position of the superpixels in the image and on their visual content dissimilarity, while $d_v(s_i, s_j)$ only depends on their visual dissimilarity. Thus, the use of $\sqrt{d_p(s_i, s_j)}$ can be explained as a weight in the Equation, in order to increase the visual distance of superpixels in case there exist "visual obstacles" between them, even if they are similar in visual content. According to this Equation, the distance between two neighboring superpixels is given by $d_v^{\frac{3}{2}}(s_i, s_j)$.

The choice of a specific distance formula for Equation 2 is not so crucial, since $d_p(s_i, s_j)$ also depends on visual difference. We have tested alternatives formulas on LHI dataset like

- $d(s_i, s_j) = d_v(s_i, s_j)$
- $d(s_i, s_j) = d_p(s_i, s_j)$
- $d(s_i, s_j) = d_p(s_i, s_j) \cdot d_v(s_i, s_j)$
- $d(s_i, s_j) = d_p(s_i, s_j) + d_v(s_i, s_j)$

getting slightly lower performance (between 1% to 4%) compared to Equation 2.

The pseudo-code of the procedure that computes the two sets of edges, E_C and E_D for graph G, is given by Algorithm 1. The input of the algorithm is the set of superpixels SP, the distances $d(s_i, s_j)\ \forall i, j \in \{1, ..., N\}$ and each intersection between superpixel s_i and the given markers $marker(s_i), i \in \{1, ..., N\}$. Function $class(.)$ returns the class (foreground or background) of the given marker pixels. E_C and E_D are initialized to the corresponding edges according to the given markers (see lines 1-12 of Algorithm 1). The $\frac{N\cdot(N-1)}{2}$ pairs of distances $d(.,.)$ are sorted and stored in vector vec_{sorted} (see line 20 of Algorithm 1). Finally, we add the sorted edges of vec_{sorted} on E_C set until G becomes a connected graph (see lines 23-31 of Algorithm 1). $Conn(s_i)$ and $Conn(s_j)$ correspond to the degree (connections) of s_i and s_j, respectively. G should be a connected graph in order to be able to execute the Vivaldi algorithm [21] that generates the superpixels' synthetic coordinates. In order to keep the graph balanced (almost equal degree per node) with respect to the nodes' degree, we have used an upper limit on node degree ($MaxConn = 10$). In Section 6.1, we report some experiments that show how the performance of the proposed framework is affected by the choice of $MaxConn$. The method has computational cost $O(N^2 \cdot \log(N))$.

```
input  : $SP = \{s_i, i \in \{1, ..., N\}\}$, $d(.,.)$, $marker(s_i), i \in \{1, ..., N\}$
output: $E_C, E_D$
$E_C = E_D = \emptyset$
for $i = 1$ to $N$ do
    for $j = i + 1$ to $N$ do
        if $marker(s_i) \neq \emptyset \wedge marker(s_j) \neq \emptyset$ then
            if $class(marker(s_i)) == class(marker(s_j))$ then
                $E_C = E_C \cup (s_i \sim s_j)$
            else
                $E_D = E_D \cup (s_i \sim s_j)$
            end
        end
    end
end
$vec = \emptyset$
for $i = 1$ to $N$ do
    for $j = i + 1$ to $N$ do
        $vec = vec \cup (s_i, s_j)$
    end
end
$vec = vec - E_D - E_C$
$vec_{sorted} = sort(vec, d(.,.))$
$L = \frac{N \cdot (N-1)}{2}$
$MaxConn = 10$
for $k = 1$ to $L$ do
    $(s_i \sim s_j) = vec(k)$
    if $G(SP, E_C \cup E_D) is\ Connected$ then
        $return$
    end
    if $max(Conn(s_i), Conn(s_j)) < MaxConn$ then
        $E_C = E_C \cup (s_i \sim s_j)$
    end
end
```

Algorithm 1: The graph construction algorithm.

3.3. Synthetic coordinates

Network coordinate systems were designed to predict latencies between Internet hosts, without the need for explicit measurements using probe queries. These algorithms assign synthetic coordinates to hosts, so that the distance between two hosts' coordinates provides an accurate latency prediction between them. This technique provides to applications the ability to predict round trip time with less measurement overhead than probing. Vivaldi [21] is a fully decentralized, lightweight, adaptive network coordinate algorithm that predicts Internet latencies with low error. Vivaldi uses the Euclidian coordinate system (in n-*dimensional* space, where n is a parameter) and the associated distance function. In our implementation, we have used $n = 20$. It holds that n should be high enough for the Vivaldi algorithm to be able to place the points in $\Re^n$ so that the point distances are satisfied. On the other hand, if n is too high (e.g. analogous to the number of points), this will make the system to be over-learned. In addition, this will (linearly) increase the computational cost of Vivaldi algorithm. We have reported (see Section 6.1) some experiments that show that the performance of the proposed framework is slightly affected by the choice of n.

Conceptually, Vivaldi simulates a network of physical springs, placing imaginary springs between pairs of network hosts [22]. The Vivaldi algorithm is quite efficient concerning its computational cost since it is a distributed algorithm [22]. It can be executed in $O(N \cdot n)$ when a centralized implementation is used.

Each node x participating in Vivaldi maintains its own coordinates $p(x) \in \Re^n$ (the position of node x that is a point in the n-*dimensional* space). Initially, all node coordinates are set at the origin. Periodically, each node communicates with another node (selected among a small set of nodes known to it). Each time a node communicates with another node, it measures its distance and learns that node's coordinates. Subsequently, the node allows itself to be moved a little by the corresponding imaginary spring connecting them. When Vivaldi converges, any two nodes' Euclidian distance will match their actual distance, even though those nodes may never had any communication.

In our case, the input to the Vivaldi algorithm [21] is a weighted graph, where the weights correspond to the nodes distances in a virtual $n-$dimensional Euclidean space. We have used the weights 0.0 and 1000.0 for E_C and E_D, respectively. These weights correspond to the Euclidean distance between the virtual position of superpixels, that is used by the Vivaldi algorithm [21] to position the superpixels in a virtual space (the $n-$dimensional Euclidean space $\Re^n$) generating synthetic coordinates. It holds that when the algorithm converges the Euclidean distance of any two superpixel positions approximates the actual distance between those superpixels (edge weight). In other words, superpixels (nodes) of the same class will be placed closer in space than superpixels of different classes, thus forming natural clusters in space. Although in the original application of Vivaldi, the actual distances were the latencies between Internet hosts, recently, we have successfully applied Vivaldi on the problem of locating communities on real and synthetic dataset graphs [23, 22].

3.4. Initial map of classified pixels

Having estimated a synthetic coordinate (position) $p(s_i) \in \Re^n$ for each superpixel $s_i, i \in \{1, ..., N\}$ of the graph, we can use a clustering algorithm in order to cluster a subset of superpixels into foreground and background classes, providing this way an initial map of classified pixels Map. The proposed algorithm creates the initial map by merging the superpixels that have been placed in proximity in $\Re^n$, meaning that they should belong to the same class. In this research, we have used a hierarchical clustering algorithm that recursively finds clusters in an agglomerative (bottom-up) mode.

The pseudo-code of this method, which classifies a subset of superpixels into foreground and background classes, providing Map, is given in Algorithm 2. The input of the algorithm is the set of superpixels SP and the position of superpixels produced by Vivaldi. The function $marker(s_i)$ returns the intersection between superpixel s_i and the given markers and function $class(.)$ returns the class (foreground or background) of the given marker pixels.

```
input  : $SP = \{s_i, i \in \{1, ..., N\}\}$, $p(.)$
output: $Map(.)$
$NumClusters = 0$, $IC = INC = \emptyset$
for $i = 1$ to $N$ do
    if $marker(s_i) \neq \emptyset$ then
        $IC = IC \cup \{s_i\}$, $Map(s_i) = class(marker(s_i))$
    else
        $INC = INC \cup \{s_i\}$, $Map(s_i) = UNKNOWN$
    end
end
for $i = 1$ to $|IC|$ do
    for $j = 1$ to $|IC|$ do
        $Dist(i,j) = \begin{cases} |p(s_i) - p(s_j)|_2 & , j > i \\ \infty & , j \leq i \end{cases}$
    end
end
$Dist1D = sort1D(Dist)$
$set = \{i \in \{2, ..., \frac{|IC| \cdot (|IC|-1)}{2}\} : Dist1D(i) \in [100, 300]\}$
$T = 200$
if  $|set| > 1$ then
    $T = Dist1D(argmax(Dist1D(set) - Dist1D(set-1)) + set(1))$
end
$i = 1$, $ICM = [1 : |IC|]$
while $Dist1D(i) < T$ do
    $[u, v] = \{(a, b) : Dist(a, b) = Dist1D(i)\}$
    if  $Map(s_u) = Map(s_v)$ then
        $set1 = \{k \in \{1, ..., |IC|\} : ICM(k) = ICM(u)\}$
        $set2 = \{k \in \{1, ..., |IC|\} : ICM(k) = ICM(v)\}$
        $C_1 = \frac{\sum_{i \in set1} A(i) \cdot p(IC(i))}{\sum_{i \in set1} A(i)}$
        $C_2 = \frac{\sum_{i \in set2} A(i) \cdot p(IC(i))}{\sum_{i \in set2} A(i)}$
        if $|C_1 - C_2|_2 < T$ then
            $ICM(set2) = ICM(v)$
        end
    end
    $i = i + 1$
end
$ICMU = unique(ICM)$
for $i = 1$ to $|ICMU|$ do
    $set = \{k \in \{1, ..., |IC|\} : ICMU(i) = IC(u)\}$
    $C = \frac{\sum_{i \in set} A(i) \cdot p(IC(i))}{\sum_{i \in set} A(i)}$
    for $j = 1$ to $N$ do
        $D(i,j) = |C - p(s_j)|_2$
    end
end
$D1D = sort1D(D)$, $i = 1$
while $D1D(i) < T$ do
    $[u, v] = \{(a, b) : D(a, b) = D1D(i)\}$
    $c = class(marker(s_{ICMU(a)}))$
    if  $Map(s_v) = UNKNOWN$ then
        $Map(s_v) = c$
    end
    $i = i + 1$
end
```

Algorithm 2: The proposed graph clustering algorithm.

Map is initialized according to the given markers (see lines 2-8 of Algorithm 2). Firstly, we merge the labeled points of the same class (foreground or background) according to their distances (see lines 9-34 of Algorithm 2) in order to reduce the number of clusters. When two clusters c_1 and c_2 are merged, we replace them by a new cluster that is placed in the mass mean of c_1 and c_2 (see lines 26, 27 and 37 of Algorithm 2). This means that its synthetic coordinates are given by the weighted mass mean of c_1 and c_2 taking into account the number of points of c_1 and c_2. Then, the distances between the labeled clusters and the unlabeled points are computed (see lines 35-41 of Algorithm 2). The proposed method sorts the unlabeled points (superpixels) according to their distance from the labeled clusters (see line 42 of Algorithm 2). Thus, we start with each point as a separate cluster with "UNKNOWN" label and we successively merge a cluster c_1 with "UNKNOWN" label with the closest labeled cluster c_2, if the distance between c_1 and c_2 is lower than a predefined threshold T according to their synthetic coordinates. If no such pair of clusters exists, the algorithm terminates.

T is computed by the histogram of distances between all points' pairs that belong in the range $[100, 300]$ (see lines 16-19 of Algorithm 2). The default value for T is 200. We have used these values, since it holds that the distance between the points of the same class is close to zero and between the points of different classes is close to 1000.0 (see Section 3.3). Therefore, T is given by the value that better discriminates the two distributions (distances between points of the same class and between points of different classes) from the histogram of distances. The computational cost of the proposed clustering algorithm is $O(N^2 \cdot n + N^2 \cdot \log(N))$, due to the computation of the distances between all points' pairs and to the sorting phase.

4. Fine Segmentation

4.1. Postprocessing of initial map

The graph clustering algorithm constructs an initial map of classified pixels. Usually, it holds that the image borders and especially the pixels close to the four image corners belong to the background class. We have used the following simple rule that takes into account this property, providing a new initial map of classified pixels. A pixel is classified to background class if it belongs to an unclassified superpixel and

- its distance from the closest image border is less than 1% of image diagonal
- its distance from the closest image corner is less than 7% of image diagonal.

Using the criterion of "unclassified superpixel" the proposed heuristic works well in cases where an object intersects the image boundary. Finally, we perform erosion on the classified superpixels using a disk of 2 pixels radius, in order to be able to correct some boundary errors of the oversegmentation algorithm.

4.2. A min-max classification criterion

This section describes the proposed criterion that has been used in the fine segmentation step. This criterion has been also used in [7, 1]. Let $S = \bigcup_{l=1}^{L} S_l$ be the set of those initially classified pixels estimated by the graph clustering algorithm described in Section 3.4. For any unclassified pixel s we can consider all the paths linking it to a classified set or region. A path $C_l(s)$ is a sequence of adjacent pixels $\{s_0, ..., s_{n-1}, s_n = s\}$. It holds that all pixels of the sequence are unlabeled, except s_0 which has label l. The cost of a particular path is defined as the maximum cost of a pixel classification according to the Bayesian rule and along the path

$$Cost(C_l(s)) = \max_{i=1...n} d_l^B(s_i) \tag{3}$$

Therefore, for each l, a topographic surface on a discrete grid is defined, considering 4-connected pixels. The initially classified pixels are defined to be at zero level, while the height of the unclassified pixels is given by the Bayesian rule.

Finally, the classification problem becomes equivalent to a search for the shortest path given the above cost, since we can define the distance of any unclassified pixel from the different classes as being the lowest height to climb for reaching site

s,

$$\delta_l(s) = \min_{C_l(s)} Cost(C_l(s)) \tag{4}$$

Therefore the decisions are topologically constrained. If we consider the graph of unclassified sites with four-connections and the classified connected components, we can define an edge weight as follows,

$$w(s_{i-1}, s_i) = \max(d_l^B(s_{i-1}), d_l^B(s_i)) \tag{5}$$

The paths defined by Equation 5 belong to the MST of the graph defined above and the computation of $\delta_l(s)$ necessitates the construction of this MST [24]. In the next two Subsections, two algorithms based on the principle of the $min - max$ Bayesian criterion for classification, are briefly presented. These algorithms have been proposed in [7, 1].

4.3. Image segmentation using MRF (ILFMA)

According to the Independent Label Flooding MRF-based mininization Algorithm (ILFMA), at first, distances $\delta_l(s)$ of Equation 4 are computed by L independent flooding procedures. The initially classified pixels, composing spatially connected regions, are grown by iteratively considering neighboring pixels. The growing procedure is applied to compute the distances $\delta_l(s)$. Among all neighboring pixels in set S_l that are unclassified and of unknown distance from label l, the nearest pixel is found, according to Equation 4. Growing proceeds until no more pixels can be added to the expanding regions, because their propagating contour reaches only pixels with different initial labels.

Given the region growing measurements derived in Equation 4, we then propose to optimize a discrete MRF in order to decide what the final class labels should be. In this manner, we aim at capturing the local interactions between pixels, which will help us refine and correct the class labels that were assigned during the previous stage of the algorithm. The problem can be formulated as follows: we seek to assign a class label l_s (from a discrete set of class labels L) to each node

$s \in V$ of a lattice graph, so that the following energy (E) is minimized:

$$E = \sum_{s \in V} \delta_l(s) + \sum_{(s,z) \in \varepsilon} w(l_s, l_z) \tag{6}$$

where ε is the set of graph edges.

To minimize the MRF energy E, we make use of the primal-dual method proposed in [25], which casts the MRF optimization problem as an integer program and then makes use of the duality theory of linear programming in order to derive solutions that have been proved to be almost optimal.

The theoretical worst-case complexity for the MRF optimization algorithm is $O(M^2)$ [26]. In practice, the above optimization method is extremely fast (see [26], [27] for a comparison with the state-of-the-art methods) and furthermore it provably provides solutions that are extremely close to the optimum for a wide class of NP-hard energies. For the problems considered in this paper, the typical running time of the above algorithm was not more than a second per image.

4.4. Image segmentation using flooding algorithm (PMCFA)

The Priority Multi-Class Flooding Algorithm (PMCFA), that is analytically described in [1], imposes strong topology constraints. All initially classified regions are propagated simultaneously and most likely decisions are taken as soon as possible. All the contours of initially classified regions are propagated towards the space of unclassified image pixels, according to similarity criteria, which are based on the class label and the segmentation features. Contour pixels s are sorted according to their dissimilarity $\delta_{l(R)}(s)$ from the class label $l(R)$ of regions R they adjoin and at each step, a group of contour pixels of minimum dissimilarity are set to the label of the class to which they most probably belong. The implementation of the flooding algorithm uses quantization of distances and priority lists to obtain a quasi-linear computational complexity [1].

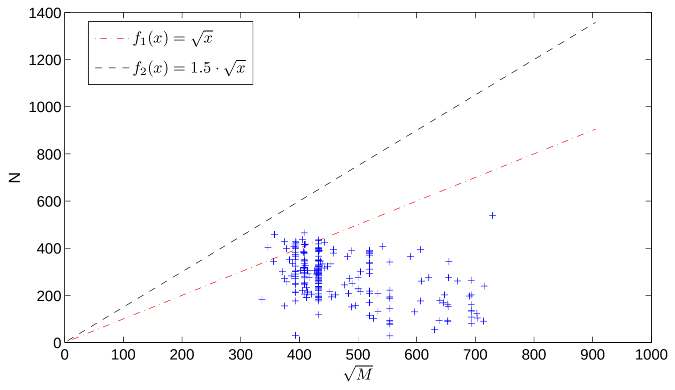


Figure 5: The number of superpixels and the corresponding square root of image pixels.

5. Computational Complexity

This section provides the analysis of the computational cost of the proposed scheme. Let M and N denote the number of image pixels and superpixels, respectively.

- The computational cost of the first step of the oversegmentation algorithm (see Section 3) is $O(M \cdot log(M))$ [17].
- The construction of the weighted graph and the execution of Vivaldi algorithm have $O(N^2 \cdot \log(N))$ and $O(N \cdot n)$ computational costs, respectively (see Sections 3.2 and 3.3). This means that the total cost of this part is $O(N^2 \cdot \log(N))$, since $n << N$.
- The computational cost of the graph clustering based on synthetic coordinates is $O(N^2 \cdot n + N^2 \cdot \log(N))$ (see Section 3.4).
- Finally, the image segmentation is provided by a Markov Random Field (MRF) model or a flooding algorithm (see Section 4). The computational cost for the flooding algorithm is quasi-linear [7, 1]. In the worst case, the computational cost for the Markov Random Field (MRF) based algorithm is $O(M^2)$ (worst case) [26].

Therefore, the computational complexity of the proposed scheme is $O(M \cdot \log(M) + N^2 \cdot n + N^2 \cdot \log(N))$, when the flooding algorithm is used. However, the number of superpixels is sufficiently lower than the number of pixels. Fig. 5 depicts the number of superpixels and the corresponding square root of image pixels ($\sqrt{M}$) computed on 222 images that have been used in the experimental results (see Section 6). Moreover, we have plotted functions $f_1(x) = \sqrt{x}$ and $f_2(x) = 1.5 \cdot \sqrt{x}$ using black dashed and red dashdot lines, respectively. According to this experiment, it holds that the 90% of cases have N lower than $\sqrt{M}$, while under any case N is lower than $1.5 \cdot \sqrt{M}$. Consequently, the assumption that $N = O(\sqrt{M})$ is true and the total computational cost of the proposed scheme is simplified to $O(M \cdot (n + \log(M)))$, when the flooding algorithm is used. On the other hand, the computational complexity of the proposed scheme is simplified to $O(M^2)$, when the Markov Random Field (MRF) model is used.

6. Experimental Results

We have tested our method on the following three datasets (LHI, Gulshan and Zhao interactive segmentation benchmarks). SGC-ILFMA and SGC-ILFMA have been compared with algorithms from the literature according to the reported results of [12], [11] and [28].

- The LHI interactive segmentation benchmark [29, 30]. This benchmark consists of 21 natural images with ground-truths and three types of users' scribbles for each image. This means that 63 different segmentation results have been obtained (see Section 6.2).

- The Gulshan interactive segmentation benchmark that consists of 151 natural images used in [11]. The Gulshan dataset consists of the GrabCut dataset (49 images) augmented with images from the PASCAL VOC09 segmentation challenge (99 images) and 3 images from the alpha-matting dataset [11] (see Section 6.3).

- The Zhao interactive segmentation benchmark [28]. The dataset is composed of five categories(animal, artifact, building, human, plant). Each category contains 10 representative images (see Section 6.4). So, this benchmark consists of 50 natural images with ground-truths and four types of users' scribbles for each image. This means that 200 different segmentation results have been obtained.

Similarly with [12], in order to measure the algorithms' performance two evaluation criteria of accuracy have been used: the Region precision (RP) and the Boundary precision (BP). Let SR and GT denote a segmentation result and a ground truth image, respectively. RP is defined as the fraction of the number of pixels that belong to the intersection of SR and GT ($|SR \cap GT|$) divided by the number of pixels that belong to the union of SR and GT ($|SR \cup GT|$).

$$RP = \frac{|SR \cap GT|}{|SR \cup GT|} \tag{7}$$

RP measures an overlap rate between a result foreground and the corresponding ground truth foreground. A higher RP indicates a better segmentation result, while the optimal segmentation is achieved when $RP = 1$, a value, which is obtained when $SR = GT$. BP calculates an inverse Chamfer distance [12, 31] between a result contour BSR and the corresponding ground truth contour BGT:

$$\begin{aligned} BP &= \frac{1}{D(BSR, BGT)} && (8) \\ D(BSR, BGT) &= \frac{1}{|BSR|} \cdot \sum_{u \in BSR} \min_{v \in BGT} |u - v|_2 && (9) \end{aligned}$$

where $|BSR|$ denotes the number of pixels of result contour BSR. It holds that higher BP values correspond to better segmentation results. In Equation 8, it appears that a division by zero possible if the boundaries align exactly. A solution on this problem is to set $BP = 2 \cdot |BSR|$, only if $D(BSR, BGT) = 0$. Since, when $D(BSR, BGT) > 0$, the maximum value of BP is $|BSR|$.

In what follows, the proposed methods using the Markov Random Field (MRF) model and the flooding algorithm are denoted as SGC-ILFMA and SGC-PMCFA, respectively.

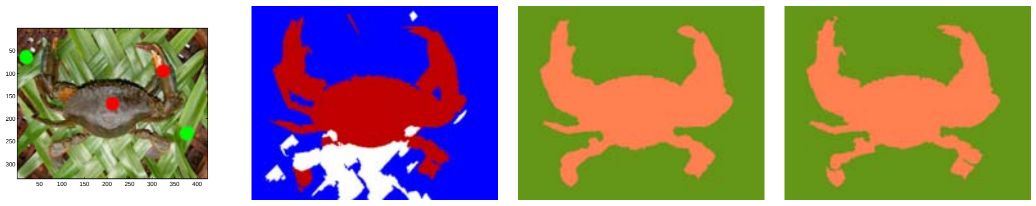
Figs. 6, 7, 8 depict intermediate (initial map of classified pixels) and final segmentation results of the proposed methods (SGC-ILFMA and SGC-PMCFA) on several images of the LHI, Gulshan and Zhao interactive segmentation benchmarks, respectively. We graphically depict the given markers on the original images using red color for foreground and green color for background, respectively. The red and blue coloring of intermediate results correspond to foreground and background classes, respectively. The white color pixels of intermediate results correspond to unclassified pixels. In Figs. 6(f), 6(r) and 7(b) the initial map of classified pixels also corresponds to the final segmentation, since the foreground and background classes are well discriminated using the initial marker information. In most of the cases both of the proposed algorithms give high performance results. In 7(j) and 7(n), although the visual information of the given markers does not suffice to discriminate the foreground and background classes, the proposed methods give good performance results. In any case, the results of the SGC-ILFMA and SGC-ILFMA algorithms are almost the same.

A demonstration of the proposed method is given in [32]. This page also contains the results of the SGC-ILFMA and SGC-PMCFA algorithms on the three datasets used herein and related bibliography.

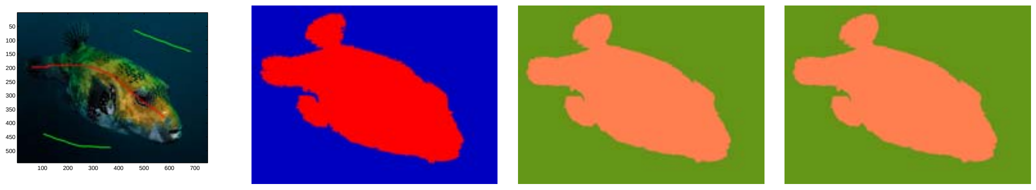
6.1. Influence of parameters

This section examines the sensitivity of the proposed method on the two parameters used, namely n and $MaxConn$, using a subset of 60 images from the used datasets.

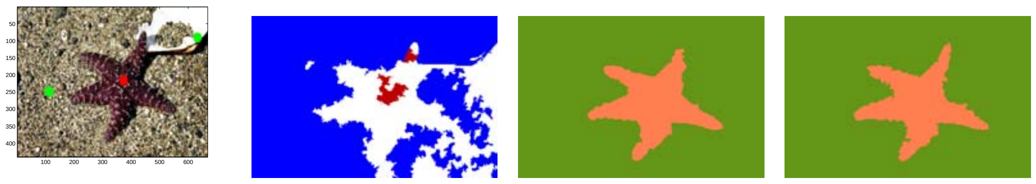
In order to prove the efficiency of our selection ($n = 20$) and to measure how this choice affects performance, we have tested different values of n, ($n \in \{5, 10, 20, 30, 100\}$) measuring the region precision (see Table 1). It holds that we get almost the same results when $n \geq 10$, while the worst results are obtained if we set $n = 5$. According to these results, the choice of n slightly affects the performance of the algorithms, under the constraint that $n \geq 10$. In [21], more details about the choice of dimension of the Euclidean space, are found.



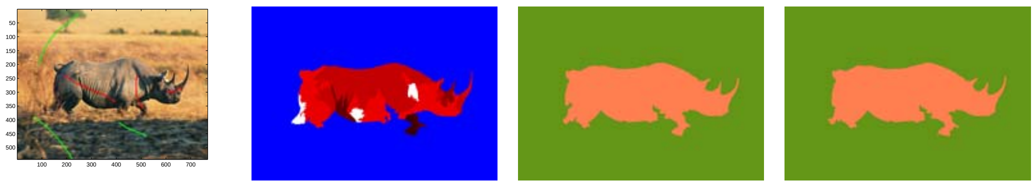
(a) (b) (c) (d)



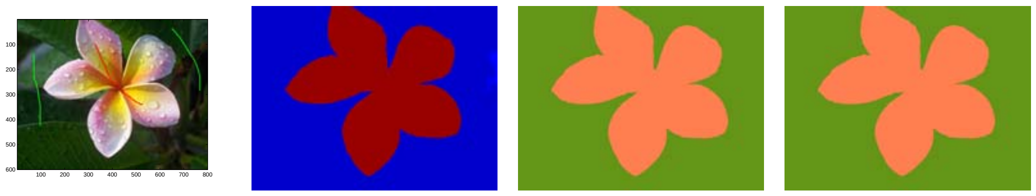
(e) (f) (g) (h)



(i) (j) (k) (l)



(m) (n) (o) (p)



(q) (r) (s) (t)

Figure 6: **(a), (e), (i), (m), (q)** Original images with markers from the LHI dataset. **(b), (f), (j), (n), (r)** Initial map of classified pixels. **(c), (g), (k), (o), (s)** Final segmentation results of the SGC-ILFMA method. **(d), (h), (l), (p), (t)** Final segmentation results of the SGC-PMCFA method.

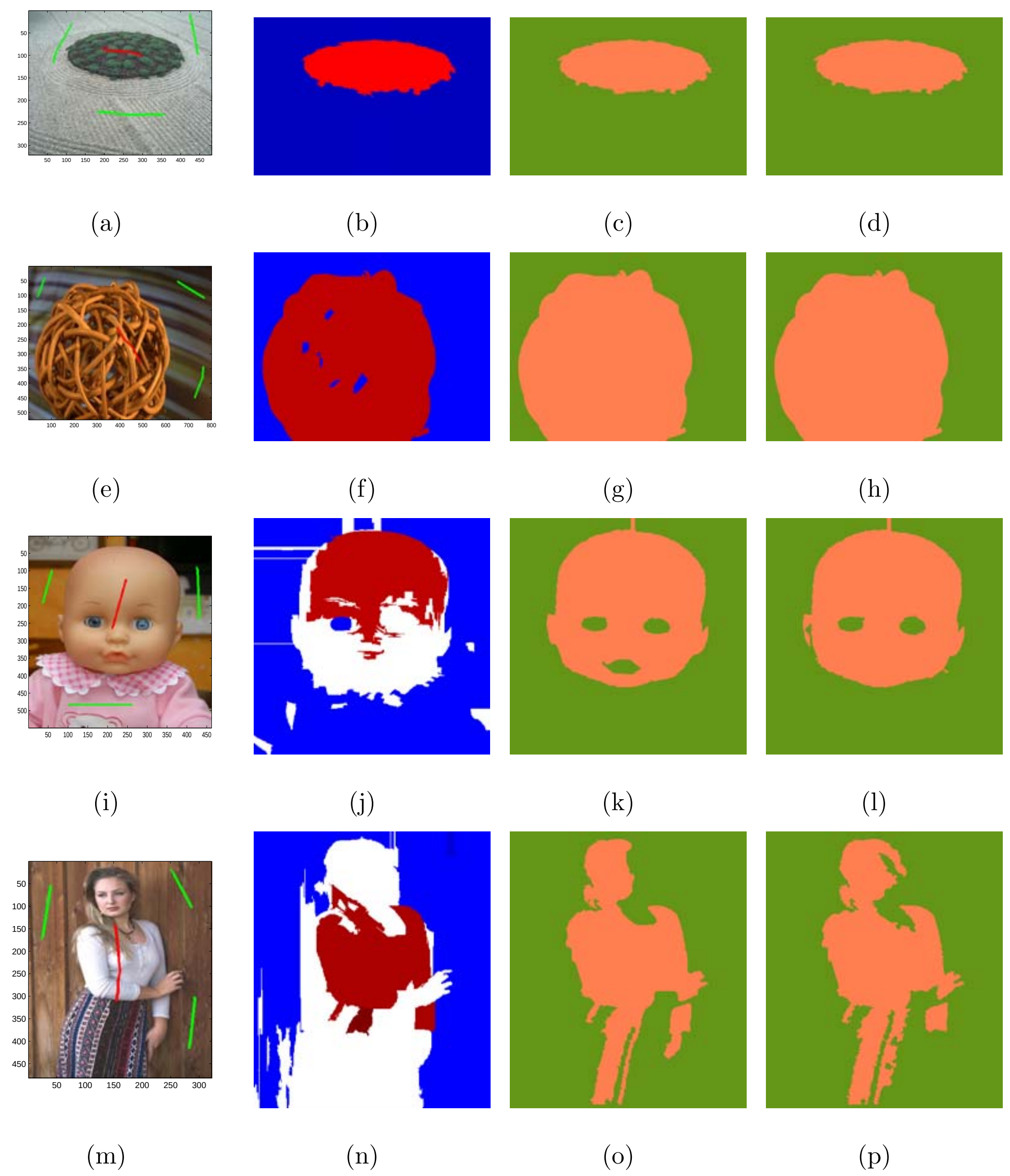


Figure 7: **(a), (e), (i), (m), (q)** Original images with markers from the Gulshan dataset. **(b), (f), (j), (n), (r)** Initial map of classified pixels. **(c), (g), (k), (o), (s)** Final segmentation results of the SGC-ILFMA method. **(d), (h), (l), (p), (t)** Final segmentation results of the SGC-PMCFA method.

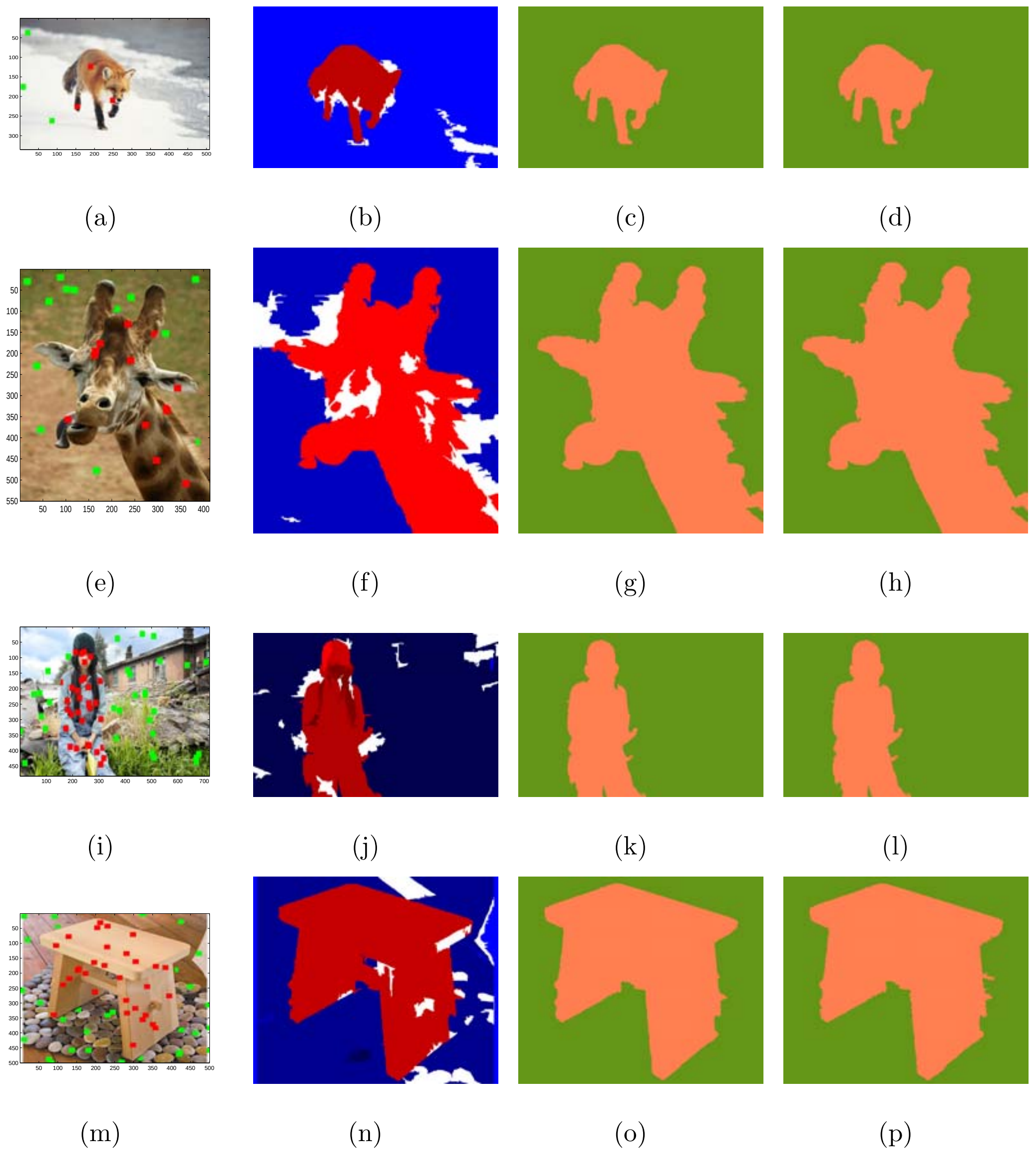


Figure 8: **(a), (e), (i), (m)** Original images with markers from the Zhao dataset. **(b), (f), (j), (n)** Initial map of classified pixels. **(c), (g), (k), (o)** Final segmentation results of the SGC-ILFMA method. **(d), (h), (l), (p)** Final segmentation results of the SGC-PMCFA method.

Method	$n = 5$	$n = 10$	$n = 20$	$n = 30$	$n = 100$
$SGC - ILFMA$	82.40%	83.54%	83.42%	83.44%	83.73%
$SGC - PMCFA$	82.93%	83.56%	83.78%	83.76%	83.88%

Table 1: The region precision of SGC-ILFMA and SGC-PMCFA algorithms using different values for n.

Method	6	8	10	12	14
$SGC-ILFMA$	83.27%	83.42%	83.42%	81.26%	81.30%
$SGC-PMCFA$	83.49%	83.79%	83.78%	82.46%	82.14%

Table 2: The region precision of SGC-ILFMA and SGC-PMCFA algorithms using different values for $MaxConn$.

Method	**SGC-ILFMA**	**SGC-PMCFA**	**CO3**	**Unger et al.**	**Bai et al.**
RP	**85.44%**	85.18%	79%	73%	50%
BP	0.365	**0.369**	0.21	0.16	0.05

Table 3: The region precision (RP) and boundary precision (BP) of SGC-ILFMA, SGC-PMCFA, CO3 [12], Unger et al. [14], Bai et al. [10] algorithms over the LHI dataset.

Concerning the parameter $MaxConn$, we have tested different values of $MaxConn$, ($MaxConn \in \{6, 8, 10, 12, 14\}$) measuring the region precision (see Table 2). It holds that we get almost the same results when $MaxConn \in [6, 10]$. Slightly lower performance is obtained when $MaxConn \geq 12$, due to the fact that the resulting graph has significant differences on the degree of nodes.

6.2. Comparison with other algorithms on the LHI dataset

We have compared the proposed methods with three other algorithms from the literature: CO3 [12], Unger et al. [14] and Bai et al. [10], using the reported results of [12]. Table 3 depicts the mean region precision (RP) and boundary precision (BP) of SGC-ILFMA, SGC-PMCFA, CO3, Unger et al., Bai et al. algorithms over the LHI dataset (63 images). The proposed methods SGC-ILFMA and SGC-ILFMA outperform the other methods. The SGC-ILFMA and SGC-PMCFA algorithms give quite similar, high accuracy for this benchmark. ILFMA provides smoother boundaries than SGC-PMCFA. The third highest performance results are given by the CO3 [12] method.

6.3. Comparison with other algorithms on the Gulshan dataset

In Gulshan dataset, there exist several images where the visual information of given markers does not suffice to discriminate the foreground and background classes (e.g. see Figs. 6(n) and 6(r)). So, in [11], extra markers have been added by a robot user according to well-defined rules, that model the way in which residual error in segmentation is progressively reduced in an interactive system. In this work, we have used this dataset without additional markers in order to examine the limits of the proposed algorithms in quite difficult cases. When no additional markers are used, the highest region precision (RP) is 58.9% (see Fig. 12 in [11]) obtained by the GSCseq algorithm [11]. The corresponding RP of the proposed methods SGC-ILFMA and SGC-PMCFA is 57.7% and 59.5%, respectively. Hence, according to these results SGC-PMCFA slightly outperforms the rest of the methods for this benchmark.

6.4. Comparison with other algorithms on the Zhao dataset

In addition, we have compared the proposed methods with four other algorithms from the literature Bai et al. [10], Couprie et al. [33], Grady [34] and Noma et al. [35] using the reported results of [28] on Zhao dataset. According to [28], Couprie et al., Grady and Noma et al. algorithms give the highest performance on Zhao dataset.

Table 4 depicts the mean region precision (RP) on four different simulation levels of SGC-ILFMA, SGC-PMCFA, Bai et al., Couprie et al., Grady and Noma et al. algorithms over the Zhao dataset. The four examples of Fig. 8 (8(b), 8(f), 8(j) and 8(n)) correspond to the four simulation levels (types of users' scribbles) of Zhao dataset. The higher the level, the more markers are added in order to simplify the problem. The proposed methods (SGC-ILFMA and SGC-ILFMA) clearly outperform the other methods at the first three simulation levels that correspond to the most difficult cases (less markers). At the fourth simulation level, it holds that the proposed methods and the Couprie et al. and Grady yield similar results outperforming the other methods. The average performances are 81.1%, 81.8%, 60.8%, 73.5%, 72.3% and 71.8% for SGC-ILFMA, SGC-PMCFA, Bai et al., Couprie et al., Grady and Noma et al. methods, respectively. Therefore, the results with highest performanceare are clearly obtained by the SGC-ILFMA and

Method	**SGC-ILFMA**	**SGC-PMCFA**	**Bai et al.**	**Couprie et al.**	**Grady**	**Noma et al.**
1	66.7%	**68.4%**	36%	50%	46%	49%
2	84.1%	**84.5%**	53%	72%	71%	69%
3	85.3%	**85.7%**	74%	84%	84%	82%
4	88.1%	**88.6%**	80%	88%	88%	87%

Table 4: The region precision (RP) on four different simulation level of SGC-ILFMA, SGC-PMCFA, Bai et al. [10], Couprie et al. [33], Grady [34] and Noma et al. [35] algorithms over the Zhao dataset.

SGC-PMCFA algorithms. The third highest performance results are given by the Couprie et al. [33] method that gives similar results with the Grady and Noma et al. method.

7. Conclusion

In this paper, we proposed an interactive image segmentation algorithm. According to the proposed framework, the problem of interactive image segmentation has been solved in two steps taking into account image visual information, proximity distances as well as the given markers. In the first step, we constructed a weighted graph of superpixels and we clustered this graph based on a synthetic coordinates algorithm in order to create an initial map of classified superpixels. In the second step, we have used a Markov Random Field model or a flooding algorithm for getting the final, pixel-level, image segmentation. Experimental results and comparisons with other methods from the literature on the LHI, Gulshan and Zhao datasets demonstrate the high performance of the proposed scheme. The proposed method can yield high performance results under different shapes of the initial markers.

Acknowledgments

This research has been partially co-financed by the European Union (European Social Fund - ESF) and Greek national funds through the Operational Program

"Education and Lifelong Learning" of the National Strategic Reference Framework (NSRF) - Research Funding Programs: ARCHIMEDE III-TEI-Crete-P2PCOORD and THALIS-NTUA-UrbanMonitor.

References

[1] C. Panagiotakis, I. Grinias, G. Tziritas, Natural image segmentation based on tree equipartition, bayesian flooding and region merging, IEEE Transactions on Image Processing 20 (8) (2011) 2276 – 2287.

[2] L. Simon, O. Teboul, P. Koutsourakis, N. Paragios, Random exploration of the procedural space for single-view 3d modeling of buildings, International journal of computer vision 93 (2) (2011) 253–271.

[3] K. McGuinness, N. O'Connor, A comparative evaluation of interactive segmentation algorithms, Pattern Recognition 43 (2) (2010) 434–444.

[4] D. Martin, C. Fowlkes, J. Malik, Learning to detect natural image boundaries using local brightness, color, and texture cues, IEEE Transactions on Pattern Analysis and Machine Intelligence 26 (5) (2004) 530–549.

[5] P. Arbelaez, M. Maire, C. Fowlkes, J. Malik, Contour detection and hierarchical image segmentation, IEEE Transactions on Pattern Analysis and Machine Intelligence (99) (2011) 1–1.

[6] N. Paragios, A level set approach for shape-driven segmentation and tracking of the left ventricle, IEEE Transactions on Medical Imaging 22 (6) (2003) 773–776.

[7] I. Grinias, N. Komodakis, G. Tziritas, Flooding and mrf-based algorithms for interactive segmentation, in: International Conference on Pattern Recognition (ICPR), 2010, pp. 3943–3946.

[8] V. Jain, B. Bollmann, M. Richardson, D. Berger, M. Helmstaedter, K. Briggman, W. Denk, J. Bowden, J. Mendenhall, W. Abraham, et al., Boundary learning by optimization with topological constraints, in: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2010, pp. 2488–2495.

[9] Y. Boykov, G. Funka-Lea, Graph cuts and efficient nd image segmentation, International Journal of Computer Vision 70 (2) (2006) 109–131.

[10] X. Bai, G. Sapiro, Geodesic matting: A framework for fast interactive image and video segmentation and matting, International journal of computer vision 82 (2) (2009) 113–132.

[11] V. Gulshan, C. Rother, A. Criminisi, A. Blake, A. Zisserman, Geodesic star convexity for interactive image segmentation, in: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2010, pp. 3129–3136.

[12] Y. Zhao, S. Zhu, S. Luo, Co3 for ultra-fast and accurate interactive segmentation, in: Proceedings of the international conference on Multimedia, ACM, 2010, pp. 93–102.

[13] N. Dalal, B. Triggs, Histograms of oriented gradients for human detection, in: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Vol. 1, 2005, pp. 886–893.

[14] M. Unger, T. Pock, W. Trobin, D. Cremers, H. Bischof, Tvseg-interactive total variation based image segmentation, in: In: British Machine Vision Conference (BMVC), 2008.

[15] D. Cremers, O. Fluck, M. Rousson, S. Aharon, A probabilistic level set formulation for interactive organ segmentation, Proc. of the SPIE Medical Imaging.

[16] L. Zhang, Q. Ji, A bayesian network model for automatic and interactive image segmentation, IEEE Transactions on Image Processing 20 (8) (2011) 2582–2593.

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

[18] C. Mallows, A note on asymptotic joint normality, The Annals of Mathematical Statistics 43 (2) (1972) 508–515.

[19] E. Levina, P. Bickel, The earth mover's distance is the mallows distance: some insights from statistics, in: IEEE International Conference on Computer Vision (ICCV), Vol. 2, 2001, pp. 251–256.

[20] D. Johnson, Efficient algorithms for shortest paths in sparse networks, Journal of the ACM (JACM) 24 (1) (1977) 1–13.

[21] F. Dabek, R. Cox, F. Kaashoek, R. Morris, Vivaldi: A decentralized network coordinate system, in: Proceedings of the ACM SIGCOMM '04 Conference, Vol. 34, 2004, pp. 15–26.

[22] H. Papadakis, C. Panagiotakis, P. Fragopoulou, Locating communities on real dataset graphs using synthetic coordinates, Parallel Processing Letters (PPL) 20 (1).

[23] H. Papadakis, C. Panagiotakis, P. Fragopoulou, Local community finding using synthetic coordinates, in: International Conference on Future Information Technology (FutureTech 2011), 2011.

[24] I. Grinias, N. Komodakis, G. Tziritas, Bayesian region growing and mrf-based minimization for texture and colour segmentation, in: International Workshop on Image Analysis for Multimedia Interactive Services, 2007.

[25] N. Komodakis, G. Tziritas, Approximate labeling via graph cuts based on linear programming, Pattern Analysis and Machine Intelligence, IEEE Transactions on 29 (8) (2007) 1436–1453.

[26] N. Komodakis, G. Tziritas, N. Paragios, Performance vs computational efficiency for optimizing single and dynamic mrfs: Setting the state of the art with

primal-dual strategies, Computer Vision and Image Understanding 112 (1) (2008) 14–29.

[27] N. Komodakis, G. Tziritas, N. Paragios, Fast, approximately optimal solutions for single and dynamic mrfs, in: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2007, pp. 1–8.

[28] Y. Zhao, X. Nie, Y. Duan, Y. Huang, S. Luo, A benchmark for interactive image segmentation algorithms, in: IEEE Workshop on Person-Oriented Vision (POV), 2011, pp. 33–38.

[29] LHI interactive segmentation benchmark.
URL `http://www.imageparsing.com/interactivesegmentation.html`

[30] B. Yao, X. Yang, S. Zhu, Introduction to a large-scale general purpose ground truth database: methodology, annotation tool and benchmarks, Lecture Notes in Computer Science 4679.

[31] A. Thayananthan, B. Stenger, P. Torr, R. Cipolla, Shape context and chamfer matching in cluttered scenes, in: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Vol. 1, 2003, pp. 127–133.

[32] Webpage with results of SGC-ILFMA and SGC-ILFMA algorithms.
URL `http://www.csd.uoc.gr/~cpanag/DEMOS/intImageSegmentation.htm`

[33] C. Couprie, L. Grady, L. Najman, H. Talbot, Power watersheds: A new image segmentation framework extending graph cuts, random walker and optimal spanning forest, in: IEEE 12th International Conference on Computer Vision, 2009, pp. 731–738.

[34] L. Grady, Random walks for image segmentation, IEEE Transactions on Pattern Analysis and Machine Intelligence 28 (11) (2006) 1768–1783.

[35] A. Noma, A. Graciano, L. Consularo, R. Cesar-Jr, I. Bloch, A new algorithm for interactive structural image segmentation, arXiv preprint arXiv:0805.1854.