Interactive Segmentation by Image Foresting Transform on Superpixel Graphs

Paulo E. Rauber, Alexandre X. Falcão, Thiago V. Spina, Pedro J. de Rezende
Institute of Computing – University of Campinas (UNICAMP)
13083-852, Campinas, SP, Brazil.
Emails: {paulorauber|tvspina}@liv.ic.unicamp.br, {afalcao|rezende}@ic.unicamp.br

***Abstract*—There are many scenarios in which user interaction is essential for effective image segmentation. In this paper, we present a new interactive segmentation method based on the Image Foresting Transform (IFT). The method oversegments the input image, creates a graph based on these segments (superpixels), receives markers (labels) drawn by the user on some superpixels and organizes a competition to label every pixel in the image. Our method has several interesting properties: it is effective, efficient, capable of segmenting multiple objects in almost linear time on the number of superpixels, readily extendable through previously published techniques, and benefits from domain-specific feature extraction. We also present a comparison with another technique based on the IFT, which can be seen as its pixel-based counterpart. Another contribution of this paper is the description of automatic (robot) users. Given a ground truth image, these robots simulate interactive segmentation by trained and untrained users, reducing the costs and biases involved in comparing segmentation techniques.**

***Keywords*-Graph-based image segmentation, image foresting transform, robot users, interactive segmentation.**

I. Introduction

Despite the progress on automatic image segmentation of the last two decades, user interaction is still essential for effective segmentation on many domains. Object segmentation in a given image can be divided into two tasks: recognition and delineation [1]. Recognition corresponds to determining the approximate whereabouts of the objects of interest, while delineation is the precise determination of which pixels belong to each of these objects.

Since recognition is very dependent on context, humans usually outperform computers, but the latter have great potential for the minutiae involved in delineation. Accordingly, several approaches have exploited a combination of user and computer efforts [2]–[6]. A common strategy in several of these approaches, operator-assisted synergistic segmentation, consists in the creation of *seeds* (pixels labeled as belonging to some object of interest) by the user and automatic delineation by the computer. Such delineation can be corrected interactively with the addition (or removal) of seeds. Thus, the final accuracy depends on the quality of the delineation and on the effort devoted by the user to the task.

It is useful to categorize segmentation algorithms into three groups [7]: purely image-based, appearance model-based and hybrid.

The purely image-based methods delineate objects based on information that can be entirely obtained from the image and/or user input. Such methods include level sets, active boundaries, fuzzy connectedness, graph cuts, watersheds, clustering and Markov random fields [7].

In contrast, model-based methods use information about objects encoded into models to perform the segmentation. These methods include active shape models [8] and atlas-based models [9], which have been applied to segmentation of anatomic structures of the brain, given magnetic resonance images [10], [11].

As the name implies, hybrid methods combine these two approaches, attempting to overcome their individual weaknesses [12].

This paper focuses on purely image-based segmentation. In particular, the presented techniques are based on the *image foresting transform* (*IFT*). This transform is a tool for the design of operators based on optimum connectivity, and has been successfully applied to the development of algorithms for image processing [13], pattern classification [14], data clustering [15], and active learning [16].

Our segmentation technique can be summarized as follows. Firstly, the input image is oversegmented. Seed pixels defined by the user associate a label to some of these oversegmented regions (*superpixels*). A superpixel graph is created to represent the oversegmentation: each superpixel corresponds to a node and arcs connect superpixels that are adjacent in the input image. An image foresting transform is then applied to associate a label to each superpixel, exploring the connection strength between superpixels and seeds. Recently, superpixels have attracted interest for allowing faster graph-based segmentation and better feature description than fixed-size windows around pixels [17]–[19].

We have compared our technique with its pixel-based counterpart, already established in the literature [6], and found ours more effective. The new technique can also be enhanced in several ways, which are noted in the appropriate sections and can be explored by future works.

Our interest in these techniques derives from several advantages the *IFT algorithm* has over the widely used segmentation techniques based on the maximum flow algorithm, often called *graph cuts* [4], [7]. For instance, graph cut methods tend to require more seeds since they tend to favor smaller boundaries. IFT-based segmentation methods are also capable of segmenting multiple objects in almost linear time, while simultaneously segmenting more than two objects using graph

cut methods is an NP-hard problem [7]. It is interesting to note that fuzzy connectedness methods, which can be efficiently implemented using the IFT algorithm, define an *optimum cut* in a graph given a particular energy function [7].

Another contribution of this work is the development of automatic (robot) users to facilitate the evaluation of interactive segmentation methods. The idea of robot users has already been explored in the literature [20], with the main objective of avoiding the costs and biases involved in the evaluation by real users. These robots work by creating seeds from the segmentation ground truth and simulate interactive segmentation by real users. We have developed two robot users that attempt to mimic the behavior of users with expertise on the techniques presented. We also implemented one of the robots described in [20], which uses a strategy that may be more similar to that of non-expert users.

This paper is organized as follows. Section II explains the image foresting transform and is essential to the understanding of the segmentation techniques described in Section III. Section IV details the robot users and results obtained with our technique, while Section V summarizes our conclusions.

II. Image Foresting Transform

The image foresting transform (IFT) is a tool for the design of operators based on connectivity. It has been applied successfully to several image processing problems [13] and also to problems including pattern classification [14] and data clustering [15]. Given a graph and a suitable cost function, the IFT creates an *optimum-path forest* by defining a path with the lowest value ending at each node [13]. Section III describes how graphs may be created from images, while this section details the general IFT algorithm, which can be seen as a generalization of Dijkstras's shortest-path algorithm.

Let $\mathcal{G} = (\mathcal{N}, \mathcal{A})$ be a graph. We denote a path ending at t as the sequence of consecutively adjacent nodes $\pi_t = \langle t_1, t_2, \ldots, t \rangle$. We denote by $\pi_s \cdot \langle s, t \rangle$ the extension of a path π_s by an arc (s, t). A path $\pi_t = \langle t \rangle$ is said to be *trivial*.

Let f be a real-valued function that assigns a value $f(\pi_t)$ to any path π_t in $\mathcal{G}$. A path π_t is optimum if $f(\pi_t) \leq f(\tau_t)$ for any other path τ_t in $\mathcal{G}$, regardless of its origin. Note that, since paths may have arbitrary values, the optimum paths are not necessarily trivial.

Defining $\Pi_t(\mathcal{G})$ as the set of all paths in $\mathcal{G}$ that end at t, an optimum value $V(t)$ for a path ending at t is defined by Equation 1.

$$V(t) \quad = \quad \min_{\pi_t \in \Pi_t(\mathcal{G})} \{f(\pi_t)\} \tag{1}$$

The IFT solves this minimization problem by computing an optimum-path forest – a function P that assigns to each node $t \in \mathcal{N}$ either its predecessor node $P(t) \in \mathcal{N}$ in an optimum path or a distinctive marker $P(t) = nil$ when $\langle t \rangle$ is optimum. When $P(t) = nil$, t is said to be a *root*. The correctness of the IFT algorithm depends on the function f being *smooth*, as defined in [13], which also guarantees that a chain of predecessors never creates cycles. Examples of smooth functions include those used in the next sections and the typical sum of non-negative arc weights in Dijkstra's original algorithm.

The IFT algorithm for solving the minimization problem posed by Equation 1 is presented below. The root $R(t)$ associated to each node t could be obtained by following its optimum path backwards using P. However, it is more efficient to propagate the roots on-the-fly, creating the map R.

Algorithm 1. – General IFT Algorithm

Input: Graph $(\mathcal{N}, \mathcal{A})$ and connectivity function f.
Output: Predecessor map P, its connectivity value map V and its root map R.
Auxiliary: Priority queue Q and variable tmp.

```
1.  For each t ∈ N, do
2.      P(t) ← nil, R(t) ← t and V(t) ← f(⟨t⟩).
3.      If V(t) ≠ +∞, then insert t into Q.
4.  While Q ≠ ∅, do
5.      Remove s from Q such that V(s) is minimum.
6.      For each t ∈ A(s), such that V(t) > V(s), do
7.          Compute tmp ← f(π_s · ⟨s, t⟩).
8.          If tmp < V(t), then
9.              If V(t) ≠ +∞, then remove t from Q.
10.             Set P(t) ← s, R(t) ← R(s), V(t) ← tmp.
11.             Insert t in Q.
```

Lines 1–3 initialize maps to trivial paths. The nodes with finite values are inserted into Q, as root candidates. The nodes with optimum trivial paths will become roots of the forest. The main loop (lines 4–11) computes an optimum path from the roots to each node s in a non-decreasing order of value. At each iteration, a path of minimum value $V(s)$ is obtained, for some node s, when it is removed from the priority queue Q. Ties are broken in Q using a first-in-first-out policy. The remaining lines evaluate whether the path that reaches a node t through s has lower value than the current estimate for optimum path ending at t and update Q, $V(t)$, $R(t)$ and $P(t)$, accordingly.

Considerations about performance and correctness of the algorithm can be found in [13].

III. Segmentation Techniques

This section presents our segmentation method based on superpixels (sec. III-A) and the method based on pixels (sec. III-B) that we used as baseline for comparison. These segmentation methods receive as input a set of labeled pixels (seeds) for each object of interest, which can be drawn by the users using brushes of different colors on the image subject to segmentation (see Fig. 4).

At this point, it is necessary to define our notation for images. An image $\hat{I}$ is a pair $(D_{\hat{I}}, \vec{I})$, where $D_{\hat{I}} \subset Z^n$ corresponds to the image domain and $\vec{I}(t)$ assigns a vector of m scalars $I_b(t)$, $b = 1, 2, \ldots, m$, to each pixel $t \in D_{\hat{I}}$. We will consider the cases where $n = 2$ and $m = 3$. In this paper, the triple $\vec{I}(t) = (I_1(t), I_2(t), I_3(t))$ denotes a point on the *YCbCr* color space. We use this color space only because

the intensity can be obtained directly from one of the vector components. In the case of grayscale images ($b = 1$) we drop the arrow on $\vec{I}$.

We consider that the result of a segmentation is a function L that maps every pixel to one of the objects of interest.

A. Segmentation based on Superpixels

The first step in our method is to generate an oversegmentation of the original image (Fig. 1). Any method may be used for this purpose, but we choose to describe here an approach based on the IFT-watershed from grayscale markers [21].

Firstly, we compute a *gradient-like image* $\hat{G} = (D_{\hat{I}}, G)$ from the original image $\hat{I}$. The image $\hat{G}$ is obtained by normalizing the image $\hat{G}_a = (D_{\hat{I}}, G_a)$, given by Equation 2. The adjacency relation $\mathcal{A}$ relates a pixel t to its eight neighbors. The triple $(\alpha_1, \alpha_2, \alpha_3)$, with $\alpha_i \in [0, 1]$, may be used to attribute different weights to the components of the color space. In *YCbCr*, for instante, it could assign less weight to the intensity component, making $\hat{G}$ more robust to changes in illumination. Based on previous experiences dealing with illumination, we have chosen $\alpha_1 = \frac{1}{5}$ and $\alpha_2 = \alpha_3 = 1$.

$$G_a(s) = \sum_{t \in \mathcal{A}(s)} \left[\sum_{b=1}^{m} \alpha_b (I_b(s) - I_b(t))^2 \right]^{\frac{1}{2}} \tag{2}$$

The next step is an IFT-watershed from grayscale markers. In this particular IFT, we consider a graph $\mathcal{G} = (\mathcal{N}, \mathcal{A})$, where $\mathcal{N} = D_{\hat{I}}$ and $(s, t) \in \mathcal{A}$ iff t is a 8-neighbor of s. The path value function f for this transform is defined by Equation 3. The set $\mathcal{R}$ contains the roots of the optimum-path forest, found on-the-fly by the IFT algorithm. This detail avoids direct computation of the regional minima of $\hat{H}$ (defined below).

A classical watershed transform on $\hat{G}$ can be understood as a flooding of the image surface that begins at one point in each of its regional minima. The flooding from one source conquers several layers of pixels until it touches floodings from other sources at the ridges of $\hat{G}$, defining the boundaries between superpixels.

However, the size of these superpixels can be controlled by removing regional minima from $\hat{G}$ by building a component tree and filtering *basins* with volume below a certain threshold [22], resulting in an image $\hat{H} = (D_{\hat{I}}, H)$, with $H(t) \geq G(t)$ for all $t \in D_{\hat{I}}$. This threshold allows us to (indirectly) control the size of the superpixels and is a parameter for our new method.

$$f(\langle t \rangle) = \begin{cases} G(t) & \text{if } t \in \mathcal{R} \\ H(t) + 1 & \text{if } t \notin \mathcal{R} \end{cases}$$
$$f(\pi_s \cdot \langle s, t \rangle) = \max\{f(\pi_s), G(t)\} \tag{3}$$

After the transform, the relabeling of the root map R results in an image oversegmented into a set of superpixels (regions) $\mathcal{S}$. Intuitively, the root map defines, for every pixel, the flooding source that conquered it.

Irrespective of the method employed to oversegment the image into a set of superpixels $\mathcal{S}$, the next step is to associate

Fig. 1. Oversegmentation superimposed on original image $\hat{I}$.

an attribute vector $\vec{v}(s)$ to each $s \in \mathcal{S}$. For the purposes of this paper, we chose the mean color of the superpixel as the attribute vector, as defined by Equation 4.

$$\vec{v}(s) = \frac{\sum_{p \in s} \vec{I}(p)}{|s|} \tag{4}$$

The final step to complete the segmentation consists on another Image Foresting Transform. We will take the liberty of redefining some mathematical objects when there is no risk of ambiguity to keep our notation clear and consistent with Algorithm 1.

This time, we consider a graph $\mathcal{G} = (\mathcal{N}, \mathcal{A})$, where $\mathcal{N} = \mathcal{S}$, and $(s, r) \in \mathcal{A}$ if and only if a pixel from s is 4-connected to a pixel from r. Denoting by $L(s)$ the label given by the user to the superpixel s and letting $L(s) = 0$ when the superpixel s is not labeled, the connectivity function f is defined by Equation 5. The function w gives the (weighted) Euclidean distance between $\vec{v}(s)$ and $\vec{v}(t)$. The weights are the same as the used for equation 2.

$$f(\langle t \rangle) = \begin{cases} 0 & \text{if } L(t) \neq 0 \\ +\infty & \text{if } L(t) = 0 \end{cases}$$
$$f(\pi_s \cdot \langle s, t \rangle) = \max\{f(\pi_s), w(s, t)\} \tag{5}$$

Intuitively, this function makes large differences between superpixels that act as barriers for the propagation of labels. Finally, the segmentation can be obtained from the root map R that results from the IFT: we assign, for every unlabeled superpixel s, the label $L(R(s))$ belonging to the root (labeled superpixel) of its optimum path (Figs. 2-3).

Note that a user could label a superpixel with multiple labels. However, this should be infrequent, since the oversegmentation should respect the borders of the correct segmentation. We do not address this issue in our implementation and arbitrarily choose one of the labels associated to a superpixel. This can be addressed in future works by applying an oversegmentation on a finer scale (lower threshold for basins volume) in superpixels where this happens.

We believe that the possibility of using an elaborate descriptor for superpixels is a very distinctive advantage of our

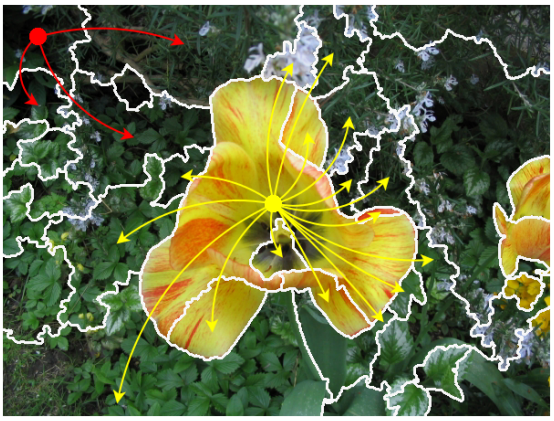

Fig. 2. Two superpixels with different labels conquer their neighbors. Superpixels are larger than usual for illustrative purposes.

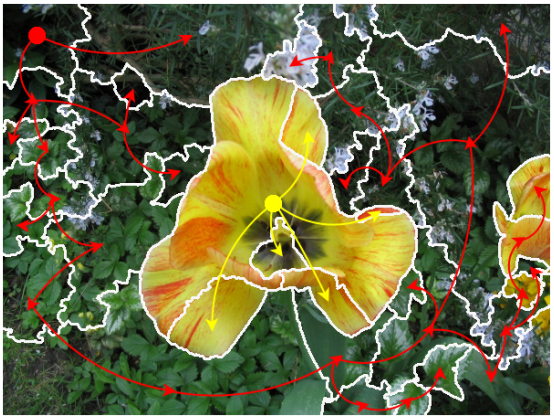

Fig. 3. The (illustrative) optimum-path forest. Note how the *red* superpixel conquered back some superpixels adjacent to the *yellow* superpixel.

Fig. 4. Seeds drawn by a user for multiple-object segmentation.

Fig. 5. Resulting multiple-object segmentation. Different hues represent different segments.

method, so we will explore this in future works. As noted in [17], instead of extracting features (attribute vectors) from a fixed-size neighborhood, texture descriptors may consider only pixels belonging to the same superpixel.

It is also important to note that this method allows multiple-object segmentation in time $O(|\mathcal{N}|\log(|\mathcal{N}|))$, which is a significant advantage as compared to methods based on *graph cuts* [7]. Figures 4 and 5 illustrate multiple-object segmentation based on our method. Although this paper focuses on 2D segmentation, our method is also capable of volumetric segmentation based on supervoxels.

Despite the efficiency of our method, in the interactive scenario, performance gains can be obtained by a differential IFT, which is capable of adapting an optimum-path forest to label changes between iterations in less time [3].

B. Segmentation based on Pixels

The pixel-based segmentation method is very similar to the method based on superpixels and, thus, may be seen as its counterpart. This method creates a graph $\mathcal{G} = (\mathcal{N}, \mathcal{A})$, where $\mathcal{N} = D_{\hat{I}}$ and $(s,t) \in A$ iff t is a 4-neighbor of s. It is important to note that this graph has considerably more nodes than its superpixel equivalent, which represents another advantage of our new method.

The function f for the transform is defined as in Equation 5. In contrast with our new method, nodes correspond to pixels and the attribute vector for a node t is defined simply as $\vec{v}(t) = \vec{I}(t)$.

This technique can be enhanced by a combination of supervised and unsupervised learning as described in [6]. However, since an adaptation of the same technique could be used to generate a better oversegmentation for the previous section, we have preferred to rescrict ourselves to this basic approach.

IV. Experiments

This section describes the experiments conducted to evaluate our method and compare it to the pixel-based one. Seeking to reduce costs and biases associated with evaluation by real users, we have developed robot users, that, given the segmentation ground truth, are capable of simulating trained and untrained users. These robots are described in Section IV-A, and the results of the experiments are described in Section IV-B.

A. Robot Users

We have implemented three types of robot users, two of which were independently developed, using different strategies

to label pixels. Even though both segmentation methods can be used for multiple-object segmentation, we have compared them in the context of binary segmentation (labeling pixels as either object of interest or background), since this is the scenario most often used to evaluate segmentation techniques. In these evaluations, the segmentation is compared to a ground truth image (a binary image that is white on the object of interest and black on the background), and a measure is used to quantify the efficacy of the segmentation method.

Each of the robots creates an ordered list $\mathcal{P}$ of seed candidates. The order defines the priority for labeling a given candidate. This list can be used in several ways, one of which is described in Section IV-B.

The *geodesic* robot (Fig. 6) is the simplest one, and was developed in [20]. The objective of this robot is to simulate the behavior of an untrained user trying to segment an image employing very little effort. Differently from [20], we do not start from pixels labeled by real users, making our method completely automatic.

When there are no labeled pixels, as in the first iteration of an interactive segmentation, the robot creates a list $\mathcal{P}$ with the geodesic center of every region of interest (object and background), computed from the ground truth. Otherwise, it computes an image $\hat{E} = (D_{\hat{I}}, E)$, with $E(t) = 0$ if pixel t was correctly labeled or $E(t) = \lambda$ if the correct label for t is λ. Its objective is to create a list with the geodesic center of every error component in this image in decreasing order of minimum distance between such center and the border of its component, i.e., larger components have higher priority for labeling.

The robot accomplishes this by using $\hat{E}$ to find every connected component with the same label through a breadth-first search. Using the Euclidean Distance Transform [13], it finds the geodesic center for every component and sorts them into a list $\mathcal{P}$, in decreasing order of minimum distance to its border.

Fig. 6. Discs centered on seed pixels generated by the geodesic robot. At each iteration, indicated by the numbers, the error components are found and a number (up to a fixed limit) of seeds is chosen in their geodesic centers.

The *superpixel* robot attempts to mimic an expert on our superpixel-based segmentation. Firstly, this robot finds superpixels on the borders of the ground truth (i.e., adjacent to superpixels with different ground truth). These superpixels are sorted in increasing order of minimum distance between their attribute vectors and the attribute vector of some adjacent superpixel on the other side of the ground truth border. Finally, the robot creates the ordered list $\mathcal{P}$ containing the geodesic centers of these superpixels. The idea is that greater similarity between adjacent superpixels with different ground truths indicates a higher risk of error for our segmentation method based on superpixels.

Fig. 7. Discs centered on seed pixels generated by the superpixel robot. Similar superpixels with different labels have higher priority for labeling. The numbers indicate the iterations described in the next subsection.

The *pixel* robot is the automatic expert on pixel-based segmentation. Its objective is to create seed pixels near pixels on a ground truth border that have low gradients in the original image, since these are places where leakages could occur.

The robot begins by computing the gradient-like image $\hat{G}$ as described in Section III-A. Then, a binary image $\hat{B} = (D_{\hat{I}}, B)$ is created, with $B(t) = 1$ if and only if $||t - q|| \leq \alpha$, for a pixel $t \in D_{\hat{I}}$, a pixel $q \in D_{\hat{I}}$ on a ground truth border, and a parameter α. Intuitively, every pixel located on a ground truth border is the center of a disc of radius α in $\hat{B}$.

For every pixel $\hat{t}$ such that $B(\hat{t}) = 1$ and $\hat{t}$ has some neighbor $\hat{q}$ with $B(\hat{q}) = 0$, the robot associates a value $v(\hat{t}) = G(z)$, where z is the nearest pixel to $\hat{t}$ that is on a ground truth border. Intuitively, each pixel on a border of $\hat{B}$ is associated to the gradient intensity of its nearest pixel on the border of the ground truth. This is similar to eroding/dilating the ground truth and associating the gradient value of the original pixels on the ground truth border to the corresponding eroded/dilated pixels.

The robot then creates a list $\mathcal{P}$ of pixels on the border of $\hat{B}$, sorted in increasing order of gradients from their associated pixels on the ground truth border.

A comparison between the behavior and efficacy of our robots and real (trained and untrained) users is desirable and could be explored in future works.

B. Results

We have chosen three datasets to compare the techniques presented: *grabcut* [23], [24], *geodesic star* [25] and *Weizmann* (single object, first human subject) [26]. These datasets

Fig. 8. Discs centered on seed pixels generated by the pixel robot. Each seed is associated to the gradient of its nearest pixel in the ground truth border. Lower gradient indicates higher priority for labeling. The numbers indicate the iterations described in the next subsection.

contain, respectively, 50, 151 and 100 natural images and their ground truths.

The experiments were conducted as follows. For a total of i iterations, given the list $\mathcal{P}$ created by a robot and a parameter n, we chose the first $n/2$ seeds inside an error component from the object and $n/2$ seeds inside an error component from the background. We correctly label a circular region (*marker*) of a given radius around these seeds, adding new seeds to the next iteration. Note that the seed candidates are generated only once by the pixel and superpixel robots, in contrast with the geodesic robot, which needs to compute the geodesic centers of the error components at each iteration.

Moreover, we consider some constraining details when drawing markers: if the marker is too near a ground truth border (as defined by a parameter), its size is reduced. This is done since the ground truth images are often imperfect and including markers too near the ground truth border could create *artificial* leakages that would not be created by real users. We also chose not to draw a marker centered on seed pixels that are already labeled, since that would reduce disproportionately the total area labeled by the pixel robot.

As in [26], we have chosen the *f-score* (also known as *f-measure* of f_1*-score*) as a measure of efficacy. Given an image $\hat{I}$ and its ground truth, we can define the following sets: true positives $\mathcal{T}_p(\hat{I})$ (set of object pixels correctly labeled in $\hat{I}$), true negatives $\mathcal{T}_n(\hat{I})$ (background pixels correctly labeled), false positives $\mathcal{F}_p(\hat{I})$ (background pixels incorrectly labeled) and false negatives $\mathcal{F}_n(\hat{I})$ (object pixels incorrectly labeled). These sets allow us to calculate the precision $P(\hat{I})$, recall $R(\hat{I})$ and f-score $F(\hat{I})$ (see Equation 6).

$$
\begin{aligned}
P(\hat{I}) &= \frac{|\mathcal{T}_p(\hat{I})|}{|\mathcal{T}_p(\hat{I})| + |\mathcal{F}_p(\hat{I})|} \\
R(\hat{I}) &= \frac{|\mathcal{T}_p(\hat{I})|}{|\mathcal{T}_p(\hat{I})| + |\mathcal{F}_n(\hat{I})|} \\
F(\hat{I}) &= 2 \cdot \frac{P(\hat{I}) \cdot R(\hat{I})}{P(\hat{I}) + R(\hat{I})}
\end{aligned}
\tag{6}
$$

For a given image $\hat{I}$, the precision $P(\hat{I})$ can be understood as the proportion of pixels *labeled as objects* that were correctly labeled, while the recall $R(\hat{I})$ is the proportion of *object pixels* that were correctly labeled. The f-score is the harmonic mean between these two measures, combining them into a single scalar that balances the different types of errors. Naturally, the f-score lies in the interval $[0, 1]$ and higher values are desirable.

We have chosen the following parameters for our experiments: for each technique, using each robot, we ran $i = 8$ iterations, choosing up to $n = 8$ pixels at each iteration. The radius of the discs centered on the seeds was five pixels, and could be reduced to one whenever the seed was too near a ground-truth border (two pixels from a ground truth border was considered a safe distance). The superpixel robot had a volume threshold of 15000. For the pixel robot, we chose $\alpha = 15$ (erosion/dilation radius). We removed a total of five images from the datasets, since they had objects of interest too small for the pixel robot. Figures 6, 7 and 8 illustrate some markers created by the robots (for the superpixel segmentation method) until the fifth iteration. Note that, for that particular image, markers created by the pixel and superpixel robots were chosen only up to the second iteration, at which point there were no longer seed candidates inside error components. The volume threshold for our superpixel segmentation, which controls superpixel size, was chosen as 150. However, the parameter space was not exhaustively searched and each image domain may benefit from different choices.

The small number of seeds per iteration allows us to see the quality of the generalizations made by each method. The seeds created by the superpixel and pixel robots are purposefully far from the ground truth border to highlight differences in delineation. The parameters were also chosen to enable for good convergence and to create interactions (subjectively) similar to what would be expected from real users. As noted earlier, empirically establishing these parameters to match real users should be further explored.

Our results are summarized in Figures 9, 10 and 11. The graphs display the mean f-score obtained by the two techniques for every combination of datasets and robots.

It is clear that our technique based on superpixels achieved higher efficacy (as defined by our measure) in the majority of cases, most notably on the Weizmann dataset and in the first iterations. In the few cases where the mean f-score was lower at some iteration, its results are not considerably inferior. For the sake of perspective, Figures 12 and 13 illustrate how meaningful a 9.5% difference in f-score can be.

It is important to emphasize that our superpixel method has even more advantages, as mentioned in Section III.

V. Conclusions

In this paper, we have described a new segmentation method based on superpixels and compared it to its pixel-based counterpart. The experiments conducted showed that our method is very effective, besides other advantages it has on efficiency and feature description. We have also described robot users

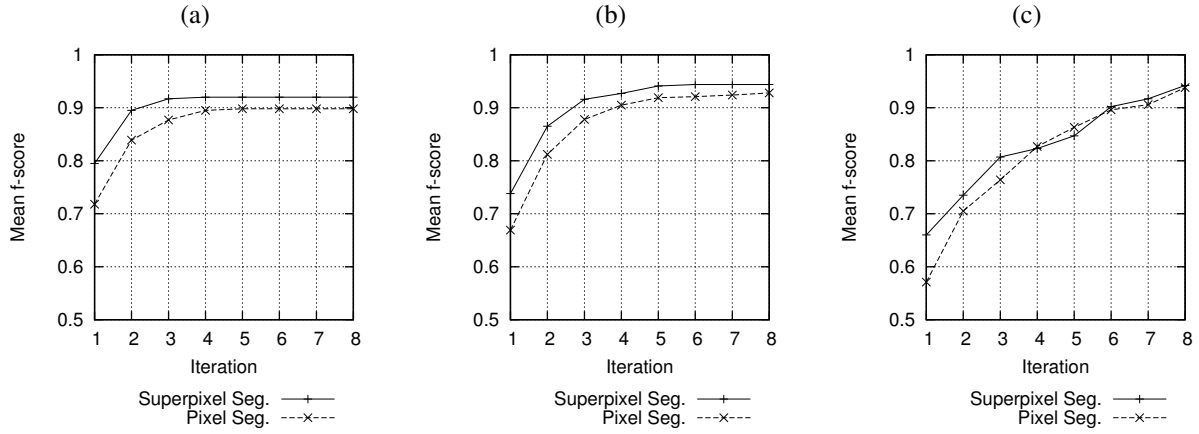

Fig. 9. Mean f-score for images in the Weizmann dataset (a) Superpixel Robot (b) Pixel Robot (c) Geodesic Robot

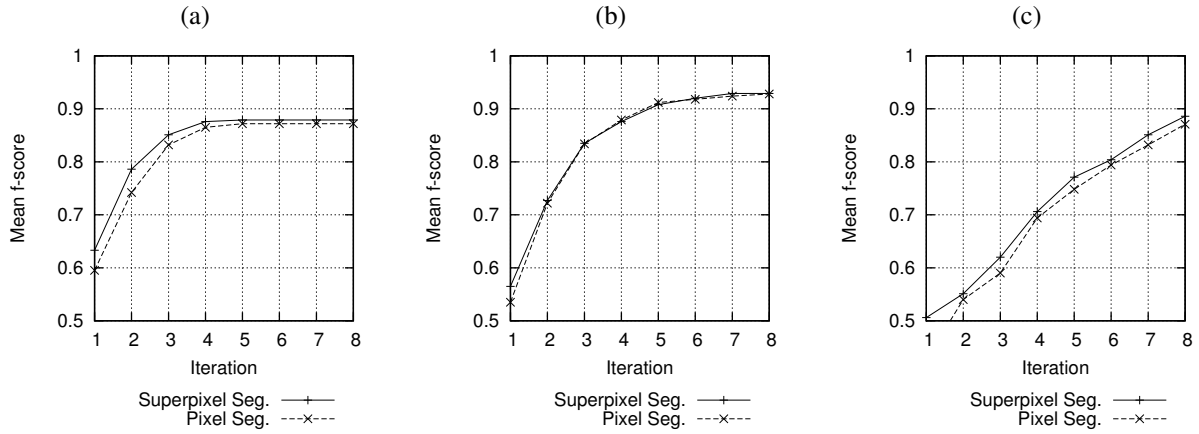

Fig. 10. Mean f-score for images in the geostar dataset (a) Superpixel Robot (b) Pixel Robot (c) Geodesic Robot

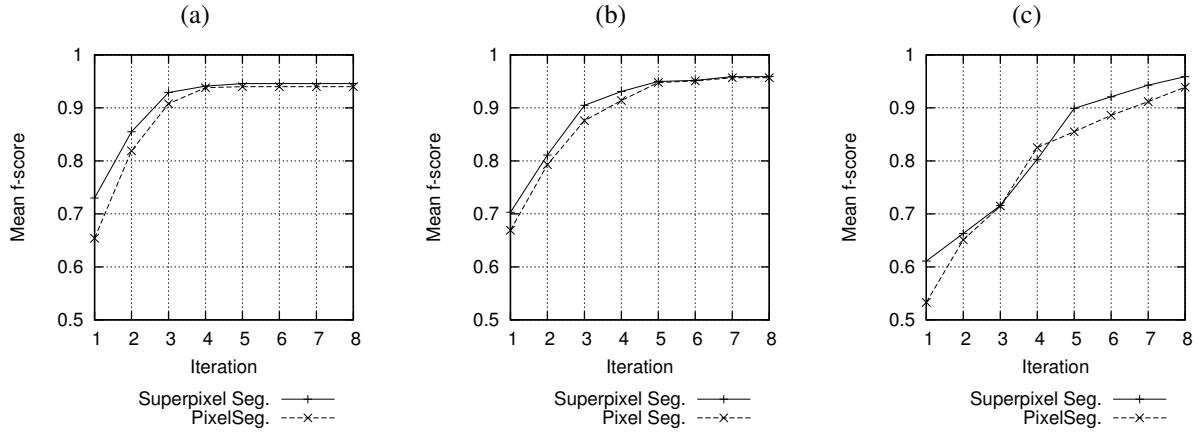

Fig. 11. Mean f-score for images in the grabcut dataset (a) Superpixel Robot (b) Pixel Robot (c) Geodesic Robot

Fig. 12. Superpixel-based segmentation, f-score: 0.985.

Fig. 13. Pixel-based segmentation, f-score: 0.89. The seeds are the same as the ones used in Figure 12.

developed to evaluate the presented methods, which we expect to see used to evaluate other interactive segmentation methods in the future. Empirically establishing the similarities between these robots and real users and comparing the efficacy of our method with other segmentation techniques described in the literature was left for future works

As highlighted in the appropriate sections, we envision, as future developments, the study of superpixel descriptors, multiscale oversegmentation for superpixels with different labels and the application of the differential IFT [3] to speed up interactive segmentation.

We would like to thank FAPESP (2007/52015-0, 2011/01434-9), CAPES and CNPq (303673/2010-9,477692/2012-5) for the financial support.

REFERENCES

[1] A. X. Falcão, J. K. Udupa, S. Samarasekera, S. Sharma, B. E. Hirsch, and R. A. Lotufo, "User-steered image segmentation paradigms: Live-wire and live-lane," *Graphical Models and Image Processing*, vol. 60, no. 4, pp. 233–260, 1998.

[2] A. Falcão, J. Udupa, and F. Miyazawa, "An ultra-fast user-steered image segmentation paradigm: live wire on the fly," *Medical Imaging, IEEE Transactions on*, vol. 19, no. 1, pp. 55–62, jan. 2000.

[3] A. Falcão and F. Bergo, "Interactive volume segmentation with differential image foresting transforms," *Medical Imaging, IEEE Transactions on*, vol. 23, no. 9, pp. 1100–1108, sept. 2004.

[4] Y. Boykov and M.-P. Jolly, "Interactive graph cuts for optimal boundary amp; region segmentation of objects in n-d images," in *Computer Vision, Eighth IEEE International Conference on*, vol. 1, 2001, pp. 105–112.

[5] A. Protiere and G. Sapiro, "Interactive image segmentation via adaptive weighted distances," *Image Processing, IEEE Transactions on*, vol. 16, no. 4, pp. 1046–1057, 2007.

[6] P. A. V. de Miranda, A. X. Falcão, and J. K. Udupa, "Synergistic arc-weight estimation for interactive image segmentation using graphs," *Comput. Vis. Image Underst.*, vol. 114, no. 1, pp. 85–99, 2010.

[7] K. Ciesielski, J. Udupa, A. Falcão, and P. Miranda, "Fuzzy connectedness image segmentation in graph cut formulation: A linear-time algorithm and a comparative analysis," *Journal of Mathematical Imaging and Vision*, vol. 44, pp. 375–398, 2012.

[8] T. F. Cootes, C. J. Taylor, D. H. Cooper, and J. Graham, "Active shape models – their training and application," *Comput. Vis. Image Underst.*, vol. 61, no. 1, pp. 38–59, 1995.

[9] K. A. Ganser, H. Dickhaus, R. Metzner, and C. R. Wirtz, "A deformable digital brain atlas system according to talairach and tournoux," *Medical Image Analysis*, vol. 8, no. 1, pp. 3–22, 2004.

[10] N. Duta and M. Sonka, "Segmentation and interpretation of mr brain images. an improved active shape model," *Medical Imaging, IEEE Transactions on*, vol. 17, no. 6, pp. 1049–1062, 1998.

[11] V. Grau, A. Mewes, M. Alcaniz, R. Kikinis, and S. Warfield, "Improved watershed transform for medical image segmentation using prior information," *Medical Imaging, IEEE Transactions on*, vol. 23, no. 4, pp. 447–458, april 2004.

[12] J. Liu and J. Udupa, "Oriented active shape models," *Medical Imaging, IEEE Transactions on*, vol. 28, no. 4, pp. 571–584, 2009.

[13] A. Falcão, J. Stolfi, and R. de Alencar Lotufo, "The image foresting transform: theory, algorithms, and applications," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 26, no. 1, pp. 19–29, 2004.

[14] J. P. Papa, A. X. Falcão, and C. T. N. Suzuki, "Supervised pattern classification based on optimum-path forest," *International Journal of Imaging Systems and Technology*, vol. 19, no. 2, pp. 120–131, 2009.

[15] L. M. Rocha, F. A. M. Cappabianco, and A. X. Falcão, "Data clustering as an optimum-path forest problem with applications in image analysis," *International Journal of Imaging Systems and Technology*, vol. 19, no. 2, pp. 50–68, 2009.

[16] P. T. Saito, P. J. de Rezende, A. X. Falcão, C. T. Suzuki, and J. F. Gomes, "Improving active learning with sharp data reduction," in *Proc. of the 20th Intl. Conf. in Central Europe on Computer Graphics, Visualization and Computer Vision (WSCG)*, 2012, pp. 27–34.

[17] J. Tighe and S. Lazebnik, "Superparsing - scalable nonparametric image parsing with superpixels," *International Journal of Computer Vision*, vol. 101, no. 2, pp. 329–349, 2013.

[18] C. Xu and J. J. Corso, "Evaluation of super-voxel methods for early video processing," in *Computer Vision and Pattern Recognition (CVPR)*, 2012, pp. 1202–1209.

[19] F. Malmberg and R. Strand, "Faster fuzzy connectedness via precomputation," in *Mathematical Morphology and its Applications to Image and Signal Processing (ISMM)*, 2013, in press.

[20] P. Kohli, H. Nickisch, C. Rother, and C. Rhemann, "User-centric learning and evaluation of interactive segmentation systems," *Int. J. Computer Vision*, vol. 100, no. 3, pp. 261–274, 2012.

[21] R. Lotufo, A. Falcão, and F. Zampirolli, "IFT-Watershed from gray-scale marker," in *Computer Graphics and Image Processing, 2002. Proceedings. XV Brazilian Symposium on*, 2002, pp. 146–152.

[22] L. Najman and M. Couprie, "Building the component tree in quasi-linear time," *Image Processing, IEEE Transactions on*, vol. 15, no. 11, pp. 3531–3539, 2006.

[23] C. Rother, V. Kolmogorov, and A. Blake, "Grabcut: interactive foreground extraction using iterated graph cuts," in *ACM SIGGRAPH 2004 Papers*. New York, NY, USA: ACM, 2004, pp. 309–314.

[24] A. Blake, C. Rother, M. Brown, P. Pérez, and P. H. S. Torr, "Interactive image segmentation using an adaptive gmmrf model," in *European Conference on Computer Vision (ECCV)*, 2004, pp. 428–441.

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

[26] S. Alpert, M. Galun, A. Brandt, and R. Basri, "Image segmentation by probabilistic bottom-up aggregation and cue integration," in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 2, 2012, pp. 315–327.