IFT-SLIC: A general framework for superpixel generation based on simple linear iterative clustering and image foresting transform

Eduardo Barreto Alexandre*, Ananda Shankar Chowdhury†, Alexandre Xavier Falcão‡, Paulo A. Vechiatto Miranda*
*Institute of Mathematics and Statistics (IME), Dept. of Computer Science,
University of São Paulo (USP), São Paulo, SP, Brazil. Email: eduardob@ime.usp.br, pmiranda@vision.ime.usp.br
†Dept. of Electronics and Telecommunication Engineering,
Jadavpur University, Kolkata, India. Email: aschowdhury@etce.jdvu.ac.in
‡Institute of Computing, Dept. of Information Systems,
University of Campinas, Campinas, SP, Brazil. Email: afalcao@ic.unicamp.br

***Abstract*—Image representation based on superpixels has become indispensable for improving efficiency in Computer Vision systems. Object recognition, segmentation, depth estimation, and body model estimation are some important problems where superpixels can be applied. However, superpixels can influence the efficacy of the system in positive or negative manner, depending on how well they respect the object boundaries in the image. In this paper, we improve superpixel generation by extending a popular algorithm — Simple Linear Iterative Clustering (SLIC) — to consider minimum path costs between pixel and cluster centers rather than their direct distances. This creates a new Image Foresting Transform (IFT) operator that naturally defines superpixels as regions of strongly connected pixels by choice of the most suitable path-cost function for a given application. Non-smooth connectivity functions are also explored in our IFT-SLIC approach leading to improved performance. Experimental results indicate better superpixel extraction using the proposed approach as compared to that of SLIC.**

***Keywords*-Simple Linear Iterative Clustering; Image Foresting Transform; Superpixel; unsupervised segmentation.**

I. Introduction

Unsupervised over-segmentation of an image, commonly called *superpixels*, is a convenient way to partition an image into relevant regions that can together represent objects. This partition can greatly reduce the computational time of the algorithms, by replacing the rigid structure of the pixel grid [1]. A superpixel can be defined as a compact region of similar and connected pixels, which locally represent a same image structure. The similarity measure can be defined in numerous ways, by using intensity, color, texture and position as features. Since the pixels contained in the same superpixel are considered equal by definition, superpixels primitives have some advantages over simple pixel primitives, like computational efficiency, since that the number of primitives are greatly reduced at the superpixel level. This brings great opportunities to alleviate Computer Vision pipelines overhead.

Despite the efficiency gain, a superpixel image representation can greatly affect, positively or negatively, the efficacy of the algorithms. Hence, it is crucial that the superpixels respect the object boundaries in the image, such that one object can be precisely defined by a set of superpixels. A good superpixel generation algorithm should possess the following desirable properties [1]:

1) **Ability to adhere to image boundaries:** The methods must respect and preserve the local structures presented in the image, since the objective of a superpixel is to represent some object or its parts in an image;
2) **Flexibility in the number of superpixels it generates:** The methods should ideally allow the customization of the desired number of superpixels, in order to prevent undersegmentation — i.e., segmenting the image in too few regions, so one superpixel would eventually contain two or more objects;
3) **Efficiency:** They need to be generated in the fastest way possible, so they don't add too much overhead to the rest of the pipeline limiting its benefits, and they must be straightforward to extend to higher dimensions;
4) **Hard segmentation:** The superpixels should not overlap each other. Each pixel must be assigned to a single superpixel;
5) **Compactness:** Superpixels should be constrained to have uniform size and shape. The ability to control the compactness of the superpixels is important. Compact, regular superpixels are often desirable because their bounded size and few neighbors form a more interpretable graph and can extract more locally relevant features.

In this paper, we extend one of the most popular superpixel generation algorithms, called *Simple linear iterative clustering* (SLIC) [1], in an *Image Foresting Transform* (IFT) [2] framework. This extension gives us a greater freedom to utilize it in a wider variety of scenarios, by choice of a more suitable path-cost function for each given application. We call this extension IFT-SLIC. SLIC essentially adapts the k-means algorithm for superpixel generation. Since k-means is based on the direct distances between pixel and cluster centers, similar pixels may not group into one compact region, even locally, and the problem is somehow addressed in SLIC. We change the distance function to be the minimum path cost in a derived image graph, such that superpixels are naturally defined as compact regions of strongly connected pixels. This result, not only improves the quality of the superpixels according to the aforementioned properties, but also reduces superpixel generation to the choice of a suitable path-cost function for a given application, and we exemplify that for natural and medical image segmentation. In the context of unsupervised segmentation of images, the methods by IFT usually consider only smooth functions [3]. The proposed version of IFT-SLIC also breaks new ground by considering non-smooth connectivity functions, which are more adaptive to cope with problems of inhomogeneity [4], and can fit to the image features more effectively.

The rest of the paper is organized in the following manner: In Section II, we discuss some previous methods, showing their strengths and limitations. Section III shows the original *Simple linear iterative clustering* (SLIC) and the *Image Foresting Transform* (IFT). The proposed extension of SLIC, named as IFT-SLIC, is presented in Section IV. In Section V, we discuss the experimental results. The paper is concluded in Section VI with an outline for directions of future research.

II. Related work

Superpixel generation is a vastly studied area, especially due to the fact that every segmentation algorithm can potentially generate superpixels.

Mean Shift and Quick Shift [5] are examples of mode-seeking algorithms that are used to generate superpixels [6] even though their main purpose is to generate a direct segmentation of the image. Mean Shift works by recursively moving data points in the pixel feature space until it reaches a dome of a density function, similarly, Quick Shift creates a tree of nearest-neighbor data points that increase the density value to reach the dome. However, these methods do not offer an explicit control over the amount of superpixels or their compactness.

The graph based segmentation approaches in [7] and [8] can also be used for extraction of superpixels. The method in [7] uses minimum spanning tree whereas the approach in [8] is based on normalized cuts. However, it has been observed that [7] produces superpixels with very irregular shapes and sizes and [8] is one of the slowest methods for the extraction of superpixels.

Another non-specialized algorithm used for superpixels is the classic Watershed [9]. As its name suggests, the idea is to create various watersheds by simulating a flooding process, starting from the local minima of the gradient of the image [10]. Each catchment basin represents a connected component in the segmentation, which consequently represents a superpixel as well. The problem with the above algorithm is that it does not offer any way to directly control the size or compactness of the superpixels, thereby violating Properties 2 and 5. Nevertheless, Property 2 could be amended by the usage of extinction values from a component tree [11]. Other methods treat data clustering as an Optimum-Path Forest (OPF) problem [3]. This corresponds to a dual definition of the IFT-Watershed [10], but running on a different graph and starting from the local maxima of a density function.

Other authors focus specifically on superpixels output, as in [12], where a geometric-flow-based algorithm is proposed. This algorithm organizes its superpixels in a lattice-like structure. Superpixels are generated by a curve evolution of a set of seeds points, regularly placed onto the image. Using some constraints, this process obtains superpixels that fulfill all the superpixel properties. However, according to Achanta et al. [1], the Turbopixel method [12] is among the slowest algorithms examined and exhibits relatively poor boundary adherence. Some papers, like [13], [14] and [15], generate superpixels in a certain geometrical order which creates a real regular lattice. The advantage of having a lattice is that the generated superpixels have the same relationship to its neighbors as simple pixels, simplifying its adaptation to methods which take advantages of neighborhood analysis. This lattice structure differs from the one in [12] which lacks a well-defined neighborhood.

III. Technical background

A. SLIC Algorithm

Simple linear iterative clustering (SLIC) [1] adapts a k-means clustering approach to efficiently generate superpixels. SLIC superpixels correspond to clusters in the $labxy$ feature space. It has two parameters, the desired number of approximately equally sized superpixels k, and a parameter m to offer control over their compactness. Its complexity is linear in the number of pixels N, and independent of the number of superpixels k.

For color images, the SLIC algorithm has the following steps:

- Firstly, the input image is converted to the $CIELAB$ color space.
- Then, a total of k' [1] initial cluster centers $C_i = [l_i\ a_i\ b_i\ x_i\ y_i]^T$ are sampled on a regular grid spaced $S = \sqrt{N/k}$ pixels apart.
- Optionally, the centers may be moved to the lowest gradient position in a 3×3 neighborhood, to avoid initialization in a noisy pixel.
- Next, in the assignment step, each pixel is associated with the nearest cluster center according to a distance measure D, but considering only the centers whose search region of $2S \times 2S$ pixels overlaps its location.
- After that, an update step adjusts the cluster centers to be the mean $[l\ a\ b\ x\ y]^T$ vector of all the pixels belonging to the cluster.
- The assignment and update steps are then repeated for a total of 10 iterations.

[1] SLIC does not guarantee the exact number k of desired superpixels. Only k' initial centers are actually used, where k' is an approximate value of k ($k' \approx k$), according to their source code.

- At the end, some disjoint pixels that do not belong to the same connected component as their cluster center may remain. Therefore, a post-processing step to enforce connectivity is applied, by assigning a distinct label to each connected component [2].

The distance measure D is given by:

$$D = \sqrt{{d_c}^2 + \left(\frac{d_s}{S}\right)^2 m^2} \quad (1)$$

where m gives the relative importance between color distance (d_c) and spatial distance (d_s). When m is large, the resulting superpixels are more compact, whereas, when m is small, we have a better adhesion to the image boundaries, but with less regular size and shape.

B. Image Foresting Transform (IFT)

An image can be interpreted as a graph $G = (\mathcal{I}, \mathcal{A})$ whose nodes are the image pixels in its image domain $\mathcal{I} \subset Z^n$, and whose arcs are the pixel pairs (s,t) in $\mathcal{A}$ (e.g., 4-neighborhood, or 8-neighborhood, in case of 2D images). The *adjacency relation* $\mathcal{A}$ is a binary relation on $\mathcal{I}$. We use $t \in \mathcal{A}(s)$ and $(s,t) \in \mathcal{A}$ to indicate that t is adjacent to s.

For a given image graph $G = (\mathcal{I}, \mathcal{A})$, a path $\pi_t = \langle t_1, t_2, \ldots, t_n = t\rangle$ is a sequence of adjacent pixels with terminus at a pixel t. A path is *trivial* when $\pi_t = \langle t\rangle$. A path $\pi_t = \pi_s \cdot \langle s,t\rangle$ indicates the extension of a path π_s by an arc (s,t). When we want to explicitly indicate the origin of a path, the notation $\pi_{s \rightsquigarrow t} = \langle t_1 = s, t_2, \ldots, t_n = t\rangle$ may also be used, where s stands for the origin and t for the destination node. A *predecessor map* is a function P that assigns to each pixel t in $\mathcal{I}$ either some other adjacent pixel in $\mathcal{I}$, or a distinctive marker nil not in $\mathcal{I}$ — in which case t is said to be a *root* of the map. A *spanning forest* is a predecessor map which contains no cycles — i.e., one which takes every pixel to nil in a finite number of iterations. For any pixel $t \in \mathcal{I}$, a spanning forest P defines a path π_t^P recursively as $\langle t\rangle$ if $P(t) = nil$, and $\pi_s^P \cdot \langle s,t\rangle$ if $P(t) = s \neq nil$.

A *connectivity function* computes a value $f(\pi_t)$ for any path π_t, usually based on arc weights. A path π_t is *optimum* if $f(\pi_t) \leq f(\tau_t)$ for any other path τ_t in G. By taking to each pixel $t \in \mathcal{I}$ one optimum path with terminus t, we obtain the optimum-path value $V(t)$, which is uniquely defined by $V(t) = \min_{\forall \pi_t \text{ in } G}\{f(\pi_t)\}$. The *Image Foresting Transform* (IFT) [2] takes an image graph $G = (\mathcal{I}, \mathcal{A})$, and a path-value function f; and assigns one optimum path π_t to every pixel $t \in \mathcal{I}$ such that an *optimum-path forest* P is obtained — i.e., a spanning forest where all paths are optimum. However, f must be *smooth* [2], otherwise, the paths may not be optimum.

The cost of a trivial path $\pi_t = \langle t\rangle$ is usually given by a handicap value $H(t)$, while the connectivity functions for non-trivial paths follow a path-extension rule. For example:

$$\begin{aligned} f_{\max}(\pi_s \cdot \langle s,t\rangle) &= \max\{f_{\max}(\pi_s), w(s,t)\} && (2)\\ f_{sum}(\pi_s \cdot \langle s,t\rangle) &= f_{sum}(\pi_s) + w(s,t) && (3)\\ f_{euc}(\pi_{r \rightsquigarrow s} \cdot \langle s,t\rangle) &= \|t - r\|^2 && (4) \end{aligned}$$

where $w(s,t) \geq 0$ is a fixed arc weight.

Recently, methods based on Image Foresting Transform (IFT) with non-smooth connectivity functions have been used successfully in the context of supervised image segmentation [16], [17], [18], [4]. Non-smooth functions comprise a less restricted class of connectivity functions, allowing advances, such as the incorporation of boundary polarity [16], [17], the use of shape constraints [18], and the better handling of images with inhomogeneity problems [4], but practically there are no studies of their application in the context of unsupervised segmentation of images.

IV. IFT-SLIC

Similar to SLIC, we start with the same selection of k' initial cluster centers $C_i = [l_i\ a_i\ b_i\ x_i\ y_i]^T$, which are sampled on a regular grid spaced $S = \sqrt{N/k}$ pixels apart.

The main difference with SLIC lies in the assignment step. Instead of using an adaptive k-means clustering approach, we consider the computation of an IFT with the non-smooth connectivity function f_D, which is based on the path-cost function $f_{\sum|\triangle I|}$ from [4] that uses the sum of the absolute value of relative intensities. These functions are justified by the theoretical and experimental resutls presented in [4][3].

The initial cluster centers $C_i = [l_i\ a_i\ b_i\ x_i\ y_i]^T$ define a set of seeds $\mathcal{S}$, such that for each pixel $r \in \mathcal{S}$ at coordinate (x_r, y_r), we have a corresponding cluster center $C_j = [l_j\ a_j\ b_j\ x_j\ y_j]^T$ and $(x_j, y_j) = (x_r, y_r)$.

Note that the path-cost function f_D plays the same role as the distance measure D in the SLIC.

$$\begin{aligned} f_D(\pi_t = \langle t\rangle) &= \begin{cases} 0 & \text{if } t \in \mathcal{S} \\ +\infty & \text{otherwise} \end{cases} \\ f_D(\pi_{r \rightsquigarrow s} \cdot \langle s,t\rangle) &= f_D(\pi_s) + (\|I(t) - I_r\| \cdot \alpha)^\beta + \|s,t\| \end{aligned}$$

where $I(t)$ is the color vector at pixel t, i.e., $I(t) = [l_t\ a_t\ b_t]^T$, and I_r is the color vector of the cluster center of seed r (i.e., $I_r = [l_j\ a_j\ b_j]^T$ where $C_j = [l_j\ a_j\ b_j\ x_j\ y_j]^T$ and r is at the coordinate (x_j, y_j)).

At the end of the assignment step, each cluster/superpixel will be represented by its respective tree in the spanning forest (i.e., the predecessor map P) computed by the IFT.

After that, an update step adjusts the cluster centers. Differently from SLIC, which considers the mean $[l\ a\ b\ x\ y]^T$ vector of all the pixels belonging to the cluster, we take for the (x,y) the coordinate of the cluster's pixel closest to the mean position. The idea is to avoid the selection of an updated position that lies outside its cluster.

[2]Moreover, according to their source code, if a certain component is too small, it is merged with a previously found adjacent component. So, the initial number k' of superpixels changes in the end as segments are added or removed.

[3]For instance, the functions f_D and $f_{\sum|\triangle I|}$ are more adaptive to cope with problems of inhomogeneity, which are common in MR images of 3 Tesla [4].

The assignment and update steps are then repeated for a total of 10 iterations. IFT-SLIC does not require a post-processing step as the connectivity is already guaranteed by design. Furthermore, IFT-SLIC can be computed in linear time with respect to the number of pixels N, and its time complexity is independent of the number of superpixels k.

A. Implementation issues

In order to reduce the computation time needed for our method, we use the following implementation strategy, using differential image foresting transforms [19].

Let $C_i^t = [l_i^t\ a_i^t\ b_i^t\ x_i^t\ y_i^t]^T$ be the i^{th} cluster center at iteration t. During the consecutive IFT computations, we only recompute the cluster center for C_i^{t+1} if:

$$\|[l_i^t\ a_i^t\ b_i^t] - [l_i^{t+1}\ a_i^{t+1}\ b_i^{t+1}]\| > \epsilon_c$$

or

$$\|[x_i^t\ y_i^t] - [x_i^{t+1}\ y_i^{t+1}]\| > \epsilon_s$$

The centers marked for recomputation have their trees removed by running the *DIFT-TREE REMOVAL* algorithm [19], and their new seed positions are added to the seed set, to compute new trees, which may invade the influence zones of other roots. When a tree is removed from the forest, its pixels become available for a new dispute among the remaining roots.

V. Experiments and Results

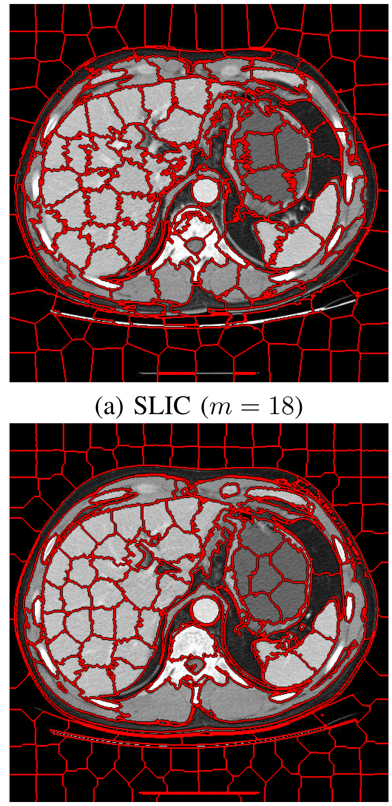
Instead of comparing the methods for a fixed configuration of their parameters, we show the accuracy values of SLIC [4] and IFT-SLIC for a wide range of their parameters. This type of approach provides a more impartial performance analysis as any bias towards poor selection of parameters is removed.

To measure the ability of the methods to adhere to image boundaries, we considered datasets with corresponding ground-truths. The superpixels by SLIC and IFT-SLIC are computed, and we assign to each superpixel the most frequent label of the ground truth occurring in its interior. The resulting segmentation is then compared to the ground-truth data using the Dice coefficient. Figure 1 illustrates this process step by step. We present the accuracy results, employing the mean performance curve involving three 2D datasets.

In the first experiment, we used the test set of 50 natural images of the public GrabCut dataset [20]. For the second dataset, we conducted quantitative experiments, using a total of 40 image slices of 10 thoracic CT studies to segment the liver (Figure 2). In the third experiment, we performed the segmentation of the talus bone, using 40 slices from MR images of the foot (Figure 3). In the case of medical images, the ground truth data was obtained from an expert of the radiology department at the University of Pennsylvania.

Figures 5, 6, and 7 show the mean accuracy curves for the three datasets for different superpixel sizes A, such that the input parameter k is set as $k = N/A$. For the talus bones we considered only superpixel sizes of 10×10 and 20×20 due to its limited size of the images (256×256 pixels).

[4]We used the source code of SLIC superpixels available at http://ivrg.epfl.ch/research/superpixels



(a) SLIC ($m = 18$)

(b) IFT-SLIC ($\alpha = 0.08$)

Fig. 2. A liver from a CT thoracic study. Superpixel results by: (a) SLIC , and (b) IFT-SLIC.

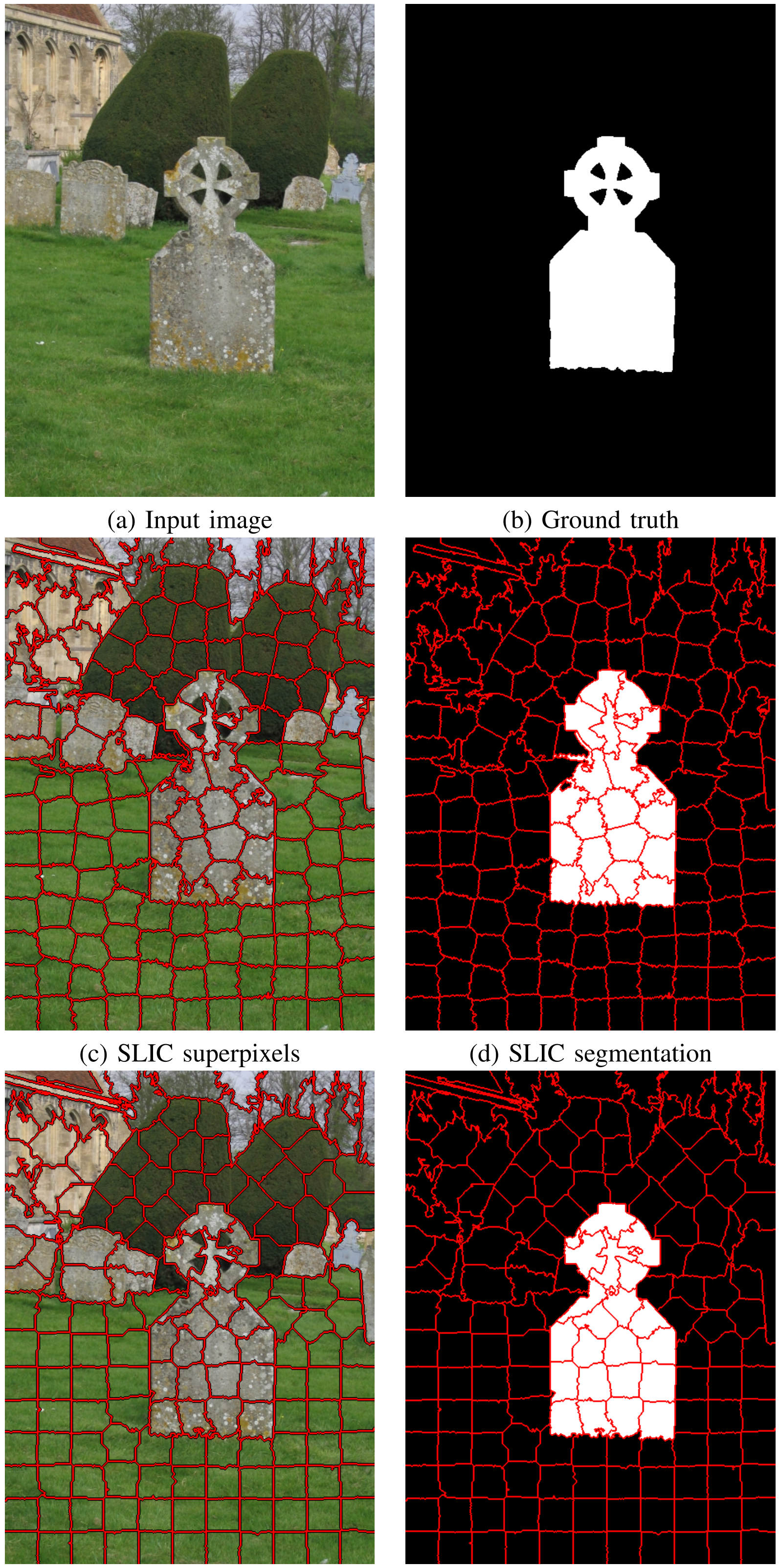
For SLIC we considered 40 samples of the parameter m, uniformly varying in the interval $[2, 80]$, which includes its recommended values [1], while for IFT-SLIC, we used 40 samples of α in $[0.005, 0.2]$ and $\beta = 12.0$ for obtaining good results. Figure 4 shows the effects on the superpixels for different values of α. We considered $\epsilon_c = 5$ and $\epsilon_s = 2$.

Clearly, the accuracy decreases as we increase the superpixel size for both methods, but IFT-SLIC presents a better performance compared to SLIC. In order to better elucidate the results, in Figures 8, 9 and 10 we plot the curves of SLIC and IFT-SLIC on a same graph, with 40 sample points ordered in increasing order of accuracy. It is clear that IFT-SLIC presents the highest accuracy values.

VI. Conclusion

In this paper, we developed an IFT based version of the SLIC algorithm, which exploits the connectivity information to improve the quality of the generated superpixels results. The results clearly showed the importance of non-smooth connectivity functions (f_D) under the framework of the *image foresting transform* (IFT) [2] for unsupervised segmentation.

As future work, we intend to test IFT-SLIC with other path-cost functions. We believe that even better results could be obtained by devising more specific path-cost functions to cope

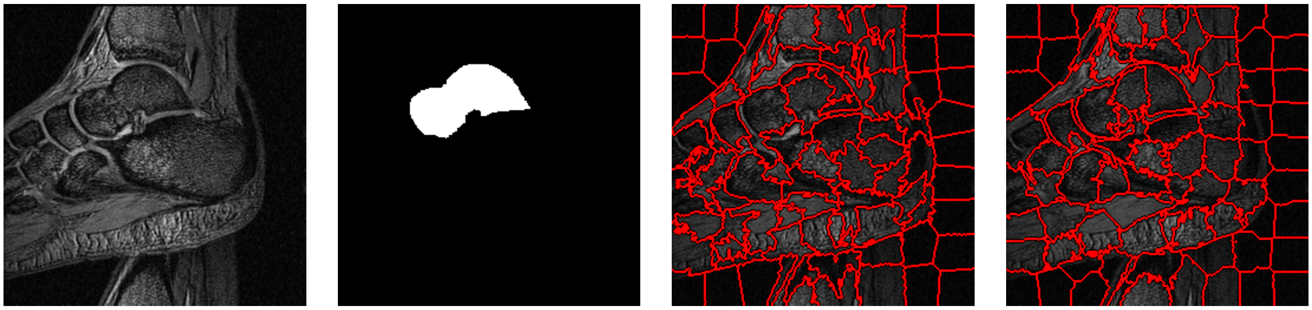


(a) Input image (b) Ground truth

(c) SLIC superpixels (d) SLIC segmentation

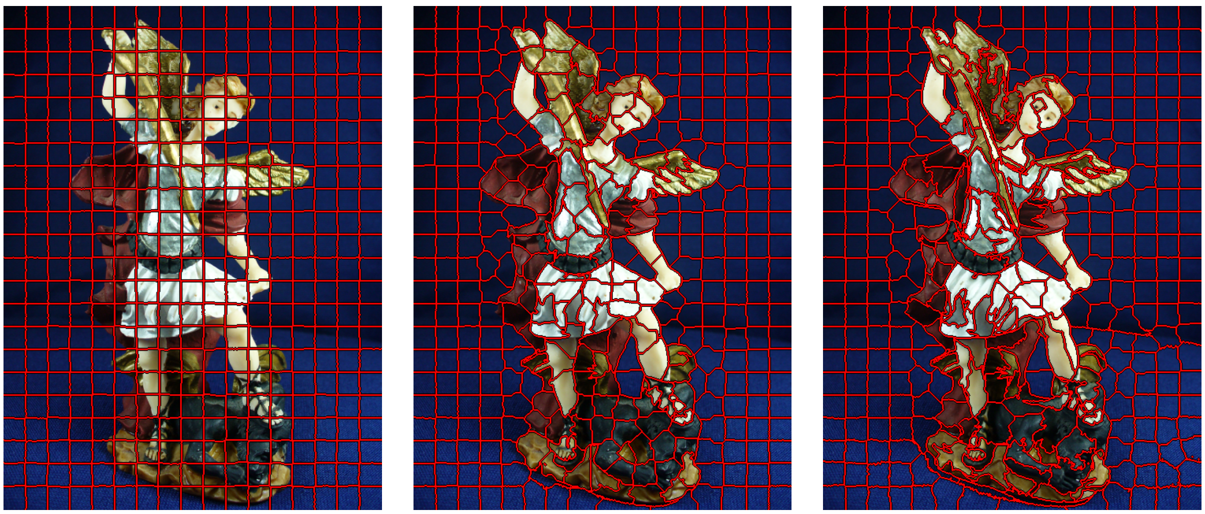
(e) IFT-SLIC superpixels (f) IFT-SLIC segmentation

Fig. 1. (a-b) Input image and its ground truth. The superpixels of 40×40 are computed by: (c) SLIC and (e) IFT-SLIC. We assign to each superpixel the most frequent label of the ground truth occurring in its interior: (d) The SLIC result has $Dice = 0.9659$, and (f) IFT-SLIC has $Dice = 0.9694$.



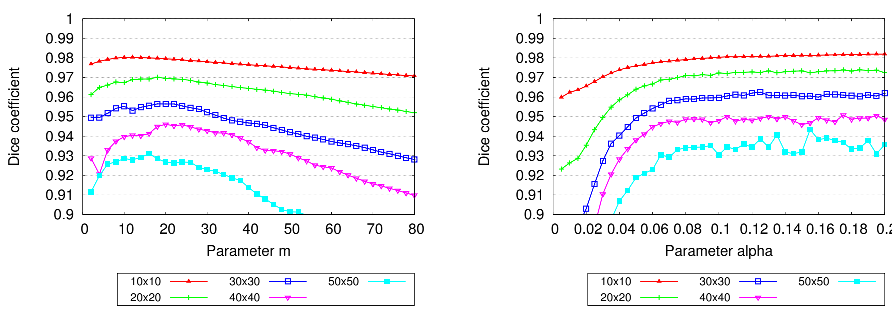
(a) Input image (b) Ground truth (c) SLIC ($m = 18$) (d) IFT-SLIC ($\alpha = 0.05$)

Fig. 3. (a-b) True segmentation of the talus in MRI slices of a foot. (c) Superpixels by SLIC. (d) Superpixels by IFT-SLIC.



(a) $\alpha = 0.01$ (b) $\alpha = 0.04$ (c) $\alpha = 0.08$

Fig. 4. The effects of different values of α on the superpixels by IFT-SLIC. For higher values of α, we have a better adhesion to the image boundaries.



(a) SLIC (b) IFT-SLIC

Fig. 5. The mean accuracy curves for segmenting the GrabCut dataset for different superpixel sizes.

with the particularities of a given application. In this sense, our framework of superpixels via IFT can open new perspectives in the research area of image processing using graphs.

Acknowledgment

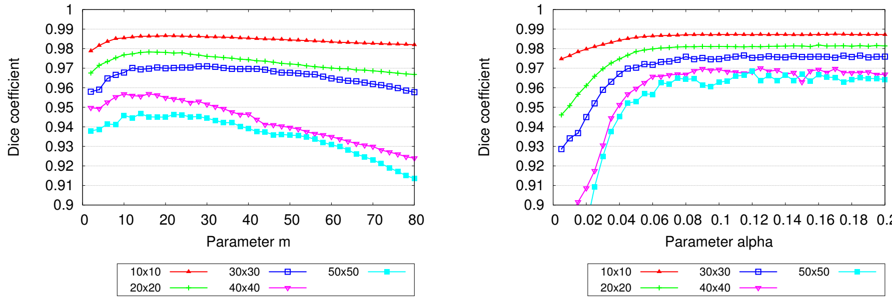
The authors thank CNPq (305381/2012-1, 486083/2013-6, 302970/2014-2, 479070/2013-0, FINEP 1266/13), FAPESP grant # 2011/50761-2, CNPq, CAPES, NAP eScience - PRP - USP, and Dr. J. K. Udupa (MIPG-UPENN) for the images.

References

[1] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Susstrunk, "Slic superpixels compared to state-of-the-art superpixel methods," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 11, pp. 2274–2282, 2012.

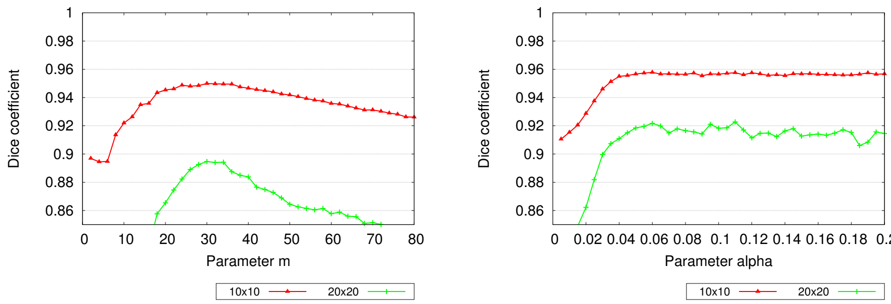
[2] A. Falcão, J. Stolfi, and R. Lotufo, "The image foresting transform: Theory, algorithms, and applications," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26, no. 1, pp. 19–29, 2004.

[3] L. Rocha, F. Cappabianco, and A. Falcão, "Data clustering as an optimum-path forest problem with applications in image analysis," *Intl. Jrnl. of Imaging Systems and Technology*, vol. 19, pp. 50–68, 2009.



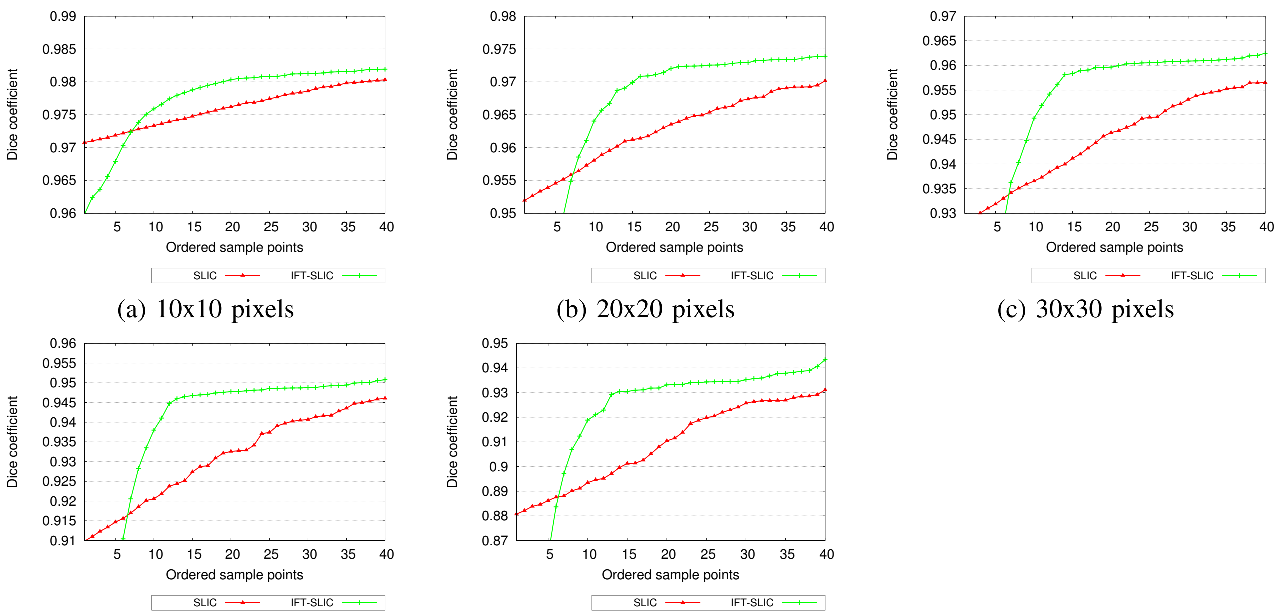
(a) SLIC (b) IFT-SLIC

Fig. 6. The mean accuracy curves for segmenting the liver dataset for different superpixel sizes.



(a) SLIC (b) IFT-SLIC

Fig. 7. The mean accuracy curves for segmenting the talus dataset for different superpixel sizes.



(a) 10x10 pixels (b) 20x20 pixels (c) 30x30 pixels

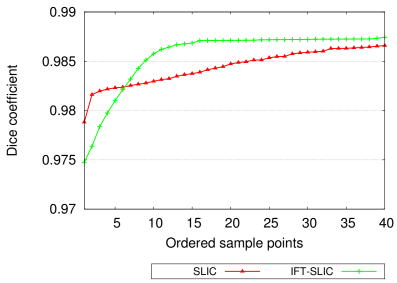
(d) 40x40 pixels (e) 50x50 pixels

Fig. 8. Curves obtained by ordering the sample points in increasing order of accuracy in the GrabCut dataset for the superpixel sizes: (a) 10x10 pixels, (b) 20x20 pixels, (c) 30x30 pixels, (d) 40x40 pixels, and (e) 50x50 pixels.

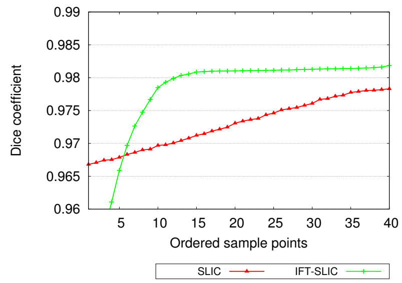
[4] L. Mansilla, F. Cappabianco, and P. Miranda, "Image segmentation by image foresting transform with non-smooth connectivity functions," in *XXVI SIBGRAPI - Conference on Graphics, Patterns and Images*, Arequipa, Peru, Aug 2013, pp. 147–154.

[5] D. Comaniciu and P. Meer, "Mean shift: a robust approach toward feature space analysis," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 24, no. 5, pp. 603–619, May 2002.

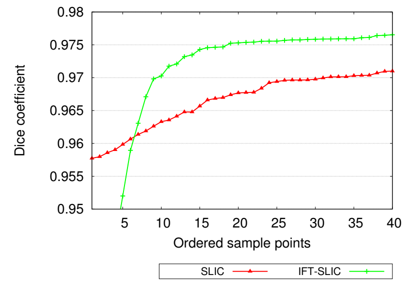
[6] B. Fulkerson, A. Vedaldi, and S. Soatto, "Class segmentation and object localization with superpixel neighborhoods," in *Computer Vision, 2009 IEEE 12th International Conference on*, Sept 2009, pp. 670–677.



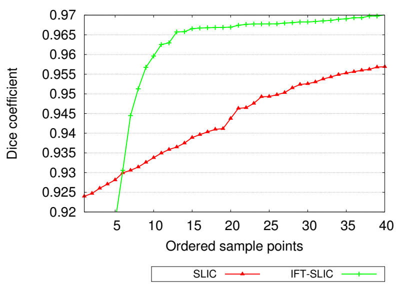
(a) 10x10 pixels



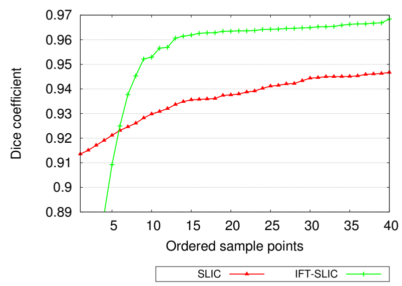
(b) 20x20 pixels



(c) 30x30 pixels

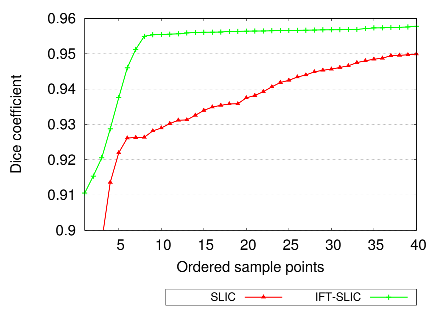


(d) 40x40 pixels

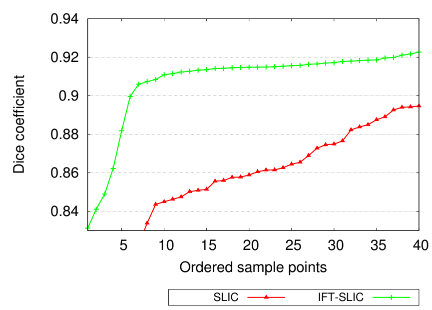


(e) 50x50 pixels

Fig. 9. Curves obtained by ordering the sample points in increasing order of accuracy in the liver dataset for the superpixel sizes: (a) 10x10 pixels, (b) 20x20 pixels, (c) 30x30 pixels, (d) 40x40 pixels, and (e) 50x50 pixels.



(a) 10x10 pixels



(b) 20x20 pixels

Fig. 10. Curves obtained by ordering the sample points in increasing order of accuracy in the talus dataset for the superpixel sizes: (a) 10x10 pixels, and (b) 20x20 pixels.

[7] P. Felzenszwalb and D. Huttenlocher, "Efficient graph-based image segmentation," *International Journal of Computer Vision*, vol. 59, no. 2, pp. 167–181, 2004.

[8] J. Shi and J. Malik, "Normalized cuts and image segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 8, pp. 888–905, 2000.

[9] S. Beucher and F. Meyer, "The morphological approach to segmentation: The watershed transformation," in *Mathematical Morphology in Image Processing*. Marcel Dekker, 1993, ch. 12, pp. 433–481.

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

[11] A. G. Silva and R. d. A. Lotufo, "Efficient computation of new extinction values from extended component tree," *Pattern Recogn. Lett.*, vol. 32, no. 1, pp. 79–90, Jan. 2011.

[12] A. Levinshtein, A. Stere, K. Kutulakos, D. Fleet, S. Dickinson, and K. Siddiqi, "Turbopixels: Fast superpixels using geometric flows," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 31, no. 12, pp. 2290–2297, Dec 2009.

[13] A. Moore, S. Prince, J. Warrell, U. Mohammed, and G. Jones, "Superpixel lattices," in *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, June 2008, pp. 1–8.

[14] ——, "Scene shape priors for superpixel segmentation," in *Computer Vision, IEEE 12th Intl. Conf. on*, Sept 2009, pp. 771–778.

[15] A. Moore, S. Prince, and J. Warrell, "Lattice cut - constructing superpixels using layer constraints," in *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, June 2010, pp. 2117–2124.

[16] P. Miranda and L. Mansilla, "Oriented image foresting transform segmentation by seed competition," *IEEE Transactions on Image Processing*, vol. 23, no. 1, pp. 389–398, Jan 2014.

[17] L. Mansilla and P. Miranda, "Image segmentation by oriented image foresting transform: Handling ties and colored images," in *18th Intl. Conf. on Digital Signal Processing*, Greece, Jul 2013, pp. 1–6.

[18] ——, "Image segmentation by oriented image foresting transform with geodesic star convexity," in *15th International Conference on Computer Analysis of Images and Patterns (CAIP)*, vol. 8047, York, UK, Aug 2013, pp. 572–579.

[19] A. Falcao 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.

[20] C. Rother, V. Kolmogorov, and A. Blake, ""grabcut": Interactive foreground extraction using iterated graph cuts," *ACM Transactions on Graphics*, vol. 23, no. 3, pp. 309–314, 2004.