Segmentation of Large Images with Complex Networks

Oscar Cuadros, Glenda Botelho, Francisco Rodrigues, João Batista Neto
Mathematics and Computer Science Institute
University of Sao Paulo (USP)
São Carlos, Brazil
Email: {ocuadros, glenda, francisco, jbatista}@icmc.usp.br

***Abstract*—Image segmentation is still a challenging issue in pattern recognition. Among the various segmentation approaches are those based on graph partitioning, which present some drawbacks, one being high processing times. With the recent developments on complex networks theory, pattern recognition techniques based on graphs have improved considerably. The identification of cluster of vertices can be considered a process of community identification according to complex networks theory. Since data clustering is related with image segmentation, image segmentation can also be approached via complex networks. However, image segmentation based on complex networks poses a fundamental limitation which is the excessive numbers of nodes in the network. This paper presents a complex network approach for large image segmentation that is both accurate and fast. To that, we incorporate the concept of super pixels, to reduce the number of nodes in the network. We evaluate our method for both synthetic and real images. Results show that our method can outperform other graph-based methods both in accuracy and processing times.**

***Keywords*-Image segmentation, complex networks and super pixels.**

I. Introduction

Image segmentation partitions the image into regions that represent similar features and constitutes an essential issue in pattern recognition due to its practical importance. In medical imaging, for example, image segmentation procedures can be used for diagnosis, allowing locating tumors and other pathologies [1]. In addition, image segmentation methods can be applied to traffic control systems, machine vision and localization of objects in satellite images [2]. Different algorithms have been proposed for image segmentation such as those based on image threshold (e.g. by means of histograms of gray levels [3]); region growing methods (e.g. [4]); and graph partitioning methods (e.g. [5]). Most of these methods present some drawbacks and do not provide accurate segmentation. Graph-based methods, for example, generally take into account spectral partitioning algorithms, which only divide a graph into two groups instead of an arbitrary number of clusters. Indeed, division into more than two groups can be attained by repeated bisection. Nevertheless, this approach does not lead to the best division into three groups [6].

With the development of complex networks theory, pattern recognition techniques based on graph have evolved considerably [7], [8], [9], [10], [11]. The identification of clusters of vertices can be performed by communities identification methods, provided by the complex networks theory [12]. Communities are defined as subsets of highly inter-connected nodes, relatively and sparsely connected to nodes in other communities [12]. These techniques provide more accurate partitions than the traditional ones based on graph, such as the spectral partitioning [13]. Since data clustering is strictly related to image segmentation, it is possible to consider those methods for identification of objects in an image. More specifically, an image is mapped onto a graph and community evaluation approaches can be considered to identify the objects, which correspond to communities in networks.

However, the image segmentation approach based on networks presents a fundamental limitation. Each image of size $M \times M$ is mapped onto a network composed by M^2 nodes, each one representing a pixel. According to this approach, only small images can be processed, since most community identification methods are computationally expensive. In order to overcome this limitation, we propose in this paper an image transformation, called super pixel, that allows a drastic reduction in the number of representative pixels in an image. Consequently, networks composed by $N << M^2$ nodes will be handled instead. Therefore, the processing times for image segmentation is reduced and large images can be processed.

Contributions: We can summarize the main contributions of this work as:

- The introduction of a complete image segmentation approach based on complex networks and evaluation in the synthetic and real large images.
- The introduction of super pixels to reduce the cardinality of the graph that models images. Therefore, the communities detection algorithms are applied to the super pixels, as opposed to pixels alone. This combination (super pixels and communities detection algorithms) is a new segmentation approach.
- Evidences that our method outperforms most traditional graph-based methods for images segmentation.

II. Related work

As mentioned above, the technique presented in this paper makes use of communities identification algorithms and super pixels. Therefore, we provide a brief overview of both fields in order to better contextualize our approach.

A. Super Pixels

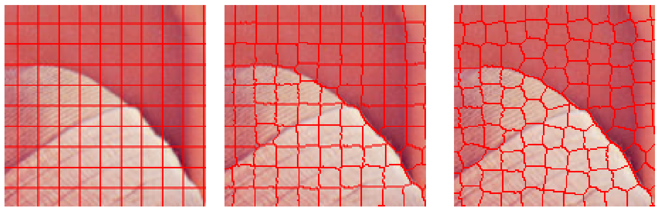
Super pixel is a region-based image segmentation approach and aims to represent images with a limited number of "super pixels" which are clusters of neighboring real pixels in an image. [14]. It is usually used as a pre-segmentation process to reduce the number of image pixels and, consequently, the computational cost of subsequent tasks. There are various studies on the extraction of super pixels [14], [15], [16]. A recent paper [17] proposes an efficient method that yields quasi-uniform super pixels with low computational cost. Results show that the method is efficient in terms of computational costs and compactness of segments and over-segmentation errors.

As proposed in [17], a connected K-means algorithm with convexity constraint is developed to allow the extraction of the super pixels. Initially, a regular grid divides the image into rectangular regions (segments) according to desired number of pixels (Fig. 1.a). Then, the pixels at the over segment boundaries are tested and assigned to the new segments by minimizing a cost function, given by:

$$C_{x,y}(i) = \lambda_1.|I(x,y) - I_i| + \lambda_2.|(x - C_x^i)^2 + (y - C_y^i)^2| \quad (1)$$

where I_i is the mean intensity of the i^{th} segment; x and y are the positions of the pixel tested among different segments; C_x^i and C_y^i are the center positions of the i^{th} segment and λ_1 and λ_2 correspond, respectively, to the weighting of intensity similarity and convexity constraints. The first term ensures that similarly colored pixels will be merged, whereas the second term provides super pixels to have more uniform and convex shapes by preventing distant pixels to be merged.

Once all the boundary pixels are tested, super pixel mean intensity and center positions are updated, yielding new super pixels. When the number of interchanged pixels is below a threshold, the iterations stop and the super pixel generation is finalized. At this stage, super pixels are said to have converged, that is, the initial regular grid cells become irregular, with their borders matching object boundaries of the image, as shown in Fig. 1.c.



(a) Initial grid. (b) Grid after some iterations. (c) Final grid.

Fig. 1. Super pixels computation.

B. Representing images as a Complex Network

Images can be mapped onto graphs, in which each node traditionally represents an image pixel and edge weights, which are computed according to a weight or similarity function. The function traduces a certain relationship between a pair of pixels. There are several weight functions which are based on Euclidian Distance, Manhattan, Gaussian and others. The computation of such function normally take into account two important parameters: threshold (t) and radius (R). The value t limits the number of connections created by checking whereas the edge weight is greater than the threshold. The radius R defines a circular region where the connections are established.

The threshold can vary according to the similarity of the pixel intensity and the radius also varies according to the image size and color regions proximity. In this paper, we defined two weight functions: one for grayscale images and the other for color images (RGB color model). As we are employing the super pixels strategy, then the gray level or RGB values of the pixels are, in fact, the mean value computed from all pixels contained in the super pixel.

C. Community identification in a Complex Network

Many real networks present community structures, that is, groups whose nodes are more densely interconnected to one another than with the rest of the network. The identification of communities is quite useful because nodes belonging to the same community are more likely to share properties. In addition, the number and characteristics of the existing communities provide subsidies for identifying the category of a network as well as understanding its dynamical evolution and organization [18].

A fundamental problem related to the community identification is how to define the best division of the network into their constituent communities, since in real networks usually no information on the number and size of existing communities is available. To solve this problem, Newman [19] proposed a measure called Modularity ($Q = \sum_i (e_{ii} - a_i^2)$), which indicates the quality of a particular division of the network. In this measure, e_{ii} is the fraction of network edges that are inserted within the community i, and a_i^2 is this same fraction, but considering that edges are inserted randomly.

Modularity values range from 0 to 1 and values close to 1 intensify the chances that communities do not exist by chance, that is, they are related to the network structure and semantics (usually values equal or greater than 0.3 are considered significant). Originally, the definition of Q was based on iterative divisions and search processes with high computational costs, due to the need of calculating Q for all possible partitions of communities in the network. In addition, it was proved that modularity optimization is an *NP-Complete* problem [20]. Therefore, several algorithms capable of finding good approximations of maximum modularity in reasonable computational time have been developed, for example, the Fast Greedy algorithm [21]. However, other algorithms which do not use modularity and have lower computational costs have also been developed. One such algorithm is the Label Propagation [22]. Details of both Fast Greedy and Label Propagation algorithms are given below.

- **Fast Greedy** [21] is a hierarchical agglomeration algorithm for community detection based on modularity measure. It is an improvement of the algorithm proposed by Newman [19] that uses a greedy optimization in which, starting from a state in which each node represents a community, communities are connected two by two, repeatedly, until the connection that results in the greatest value of modularity Q is selected. The algorithm continues until the entire network is considered a unique community.

The computational cost of the Newman's algorithm is $O((m + n)n)$ or $O(n^2)$ for sparse graphs, where n is the number of nodes and m is the number of edges of the network. Due to efficient data structures, Fast Greedy computational cost drops to $O(md(logn))$, where d corresponds to the depth of the dendrogram that describes the network. For sparse and hierarchical networks, $m \sim n$ and $d \sim logn$, thus the Fast Greedy runs in $O(nlog^2n)$. Fig. 2 presents a dendogram that describes a network. The circles correspond to the network nodes, which are connected two by two, repeatedly, until result in the single community. The dotted line corresponds to the highest value of modularity found in this dendogram.

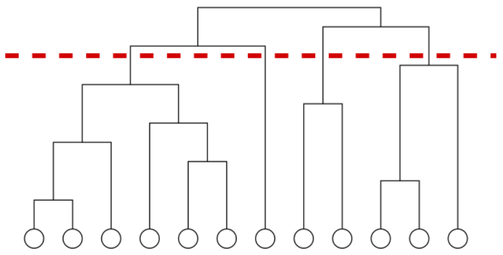


Fig. 2. Dendogram that describes a network.

- **Label Propagation** [22] is an algorithm for community detection based on the propagation of labels, and whose main advantage is its low computational cost. Initially, each node gets a different label. At each iteration, all nodes are scanned sequentially, and each node takes the label shared by the majority of its neighbors. If there is no unique majority, one of the majority labels is picked at random. Thus, labels propagate across the graph, causing the majority of the labels to disappear, while some will remain. The process reaches convergence when each vertex has the "majority label" of its neighbors. Fig. 3 shows a sequence of label propagation performed by the algorithm.

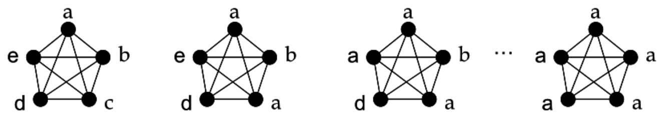


Fig. 3. Nodes are updated one by one as we move from left to right. Due to a high density of edges (highest possible in this case), all nodes acquire the same label.

When the neighbors of a node do not have an unique majority label, this algorithm does not result in an unique solution due to the random selection of labels. Thus, different partitions can be derived from the same initial condition. The time complexity of each iteration of the algorithm is $O(m)$ (m being the number of edges) and, the number of iterations needed for convergence seems to either be independent on the graph size or grow very slowly with it. That is the reason why this algorithm is really fast and can be used for analyzing large networks.

For the task of image segmentation, the Fast Greedy algorithm is more appropriate than the Label Propagation algorithm because the former results in the best division of the network in communities, that is, with the greatest value of modularity. However, the Label Propagation algorithm does not always produces the best division due to the random selection of labels. In addition, it results in an over-segmentation image, that is, a very large amount of communities.

III. Methodology

The methodology adopted in this paper can be best described by the diagram of Fig. 4. Given an image, we define a regular grid of size $n \times n$ which, after convergence, will yield a super pixel with irregular cells, whose borders match the boundary of objects in the image. From the super pixel, a Complex Network is created. The final stage consists of a community detector algorithm that produces a segmented image. These tasks are detailed below.

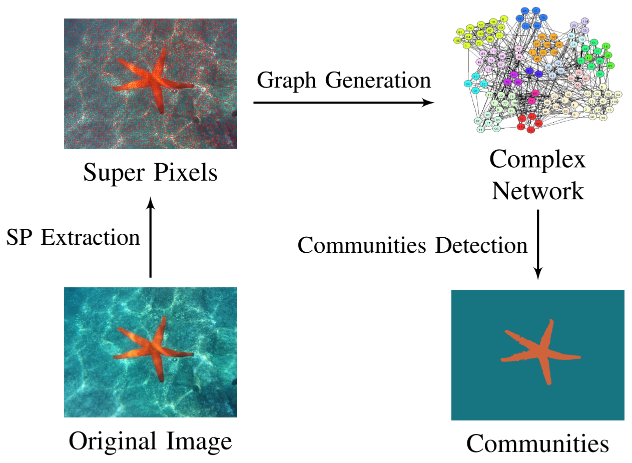


Fig. 4. Image segmentation approach based on complex network combined with super pixels. In the figure, SP = Super Pixel.

Super Pixels Computation

We implemented a super pixel extraction technique based in the approach developed by Cigla [17] (see Section II). We chose this technique because it is fast, produces compact super pixels and fewer over-segmentation errors. Although Cigla's paper does not explain the technique in details, we implemented a version of the super pixel approach using C++ language and efficient object-oriented data structures.

Four distinct parameters directly influence the outcome of the algorithm: (a) super pixel size s (initially taken as a square grid with side size is s); (b) number of iterations i; (c) weight of intensity similarity λ_1 and (d) weight of convexity constraints λ_2. Changes in these parameters can affect both speed of the algorithm and accuracy of the produced super pixels. For large values of s (super pixel size), more iterations are required in order for the algorithm to converge, therefore, increasing the processing time. If λ_2 is too high, super pixels may not converge at all. To achieve good segmentation results we run some experiments in a group of images and finally choose appropriate parameters values.

Graph Generation

From the converged super pixels, a graph is constructed. It is important to point out that nodes are no longer single pixels, but super pixels. Hence, to generate a graph we must consider the mean intensity of the entire super pixel. Then we devised two weight function to handle both gray level and colored images. For gray level images, nodes are linked to each other with a weight function based on the mean intensity within segments, given by:

$$W_{i,j} = 1 - \mid I_i - I_j \mid \geq t \tag{2}$$

where $W_{i,j}$ is the edge weight between i_{th} and j_{th} super pixels and is defined in the interval $[0, 1]$ and, I_i represents the mean intensity of gray level of segment i. Connections between nodes are considered only if the weight is greater than threshold t. The value t can vary according to the similarity of the pixel intensity. In addition, connections are defined only inside a circular super pixel neighborhood of radius R, which varies according to the image size, super pixel size and color regions proximity. Fig. 5 depicts a yellow region that encompasses all super pixels with a maximum distance of $R = 5$ from a given super pixel. The choice of a "good" values R is important to prevent a connection between two super pixels that are far from each other. The weight function for colored images is the normalized Euclidean distance of the mean of RGB values of super pixels, and is given by:

$$W_{i,j} = 1 - \frac{\sqrt{(R_i - R_j)^2 + (G_i - G_j)^2 + (B_i - B_j)^2}}{\sqrt{3}} \geq t \tag{3}$$

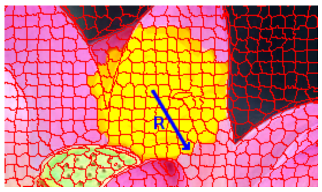


Fig. 5. Super pixel neighborhood (yellow region) of radius $R = 5$.

We emphasize that both weight functions for image segmentation were established by us. Also, the graph generation process was implemented with threads, which reduced the computational cost.

Image Segmentation using Complex Networks

After the construction of graphs, communities detection algorithms were used to segment images. We chose Fast Greedy [21] and Label Propagation [22] algorithms because they have low computational cost, compared to others algorithms. These algorithms are implemented in the Igraph Library [23] and were embedded in our implementation. The result of these algorithms is the division of graph nodes in different communities. Each community represents an image region.

The results presented in Section IV are based on the Fast Greedy algorithm only because it is more appropriate than Label Propagation algorithm to the task of image segmentation. As mentioned in section II, although faster, Label Propagation community detector results in an over-segmentation image, that is, produces a very large amount of communities.

The algorithm 1 presents the complete segmentation process, including the super pixel computation, graph generation and communities detection.

Algorithm 1: Algorithm of the complete segmentation process.

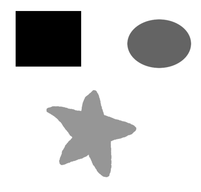
```
input : Image $I$, Intensity $\lambda_1$, Convexity $\lambda_2$,
        SP-iterations $n$, Radio $R$, Threshold $t$
output: Detected Communities
/* Super Pixel extraction      */ ;
Divide $I$ into rectangular segments;
foreach $n$ do
    foreach segment $i$ do
        foreach border pixel $p$ of segment $i$ do
            Compute $C_p(i)$ // cost function (1) ;
            Compute $C_p(j)$ // Segment neighbor $j$ ;
            if $C_p(i) > C_p(j)$ then
                Assign $p$ to the segment $j$;
        Update mean intensity and center position of
        segment $i$;
/* Graph generation          */ ;
Create a graph $G$;
foreach segment $i$ do
    foreach neighbor $j$ if the segment $i$ is inside a region
    of radius $R$ do
        Compute $W_{i,j}$  // weight function (2)
        or (3) ;
        if $W_{i,j} \geq t$ then
            Add to $G$ an edge between $i$ and $j$ with
            weight $W$;
/* Fast Greedy community detector  */ ;
Assign each vertex of $G$ as a community;
repeat
    while true do
        Join communities in pairs and compute $Q$ ;
        if $Q$ is the greatest increase then
            break ;
until the input of the network is an unique community;
```

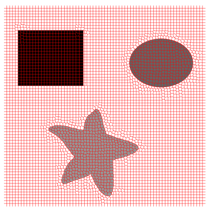
IV. Experimental Results and Comparisons

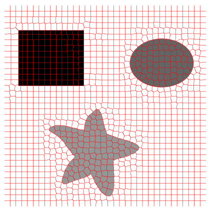
This section provides some experiments aiming to show how image segmentation of large images can be effectively attained by the technique proposed in this paper. We employ both synthetic and real images and compare our method with other two well known graph-based image segmentation techniques. We run all experiments with both Label Propagation and Fast Greedy community detectors. In all of them the latter produced more accurate segmentation than the former. Hence, the results reported here are related with the Fast Greedy algorithm.

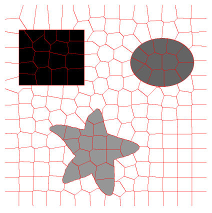
A. *Experiment 1 - Synthetic Image*

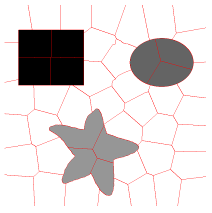
This first experiment, rather than focusing on image segmentation accuracy, describes the trade off between distinct resolutions of super pixel and computational time for the three stages of the image segmentation process, that is, super pixel convergence, graph assembling and community detection. To that, we employed a 700 $\times$ 700 synthetic image with three objects with distinct gray levels each (ellipsis, star, rectangle) and a white background. Fig. 6 shows the original image and the converged super pixel grids for five super pixel sizes: 10, 20, 50, 100 and 150.


(a) Original Image.


(b) Converged 10 $\times$ 10 super pixel.


(c) Converged 20 $\times$ 20 super pixel.


(d) Converged 50 $\times$ 50 super pixel.


(e) Converged 100 $\times$ 100 super pixel.

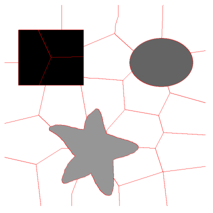

(f) Converged 150 $\times$ 150 super pixel.

Fig. 6. Varying the super pixel resolutions.

The segmented image for Fig. 6.a, for all super pixel size, is shown in Fig. 7. Each of the four detected components has been assigned a different color. The expected 100% accuracy in segmentation is due to a) correct convergence of the grid cells (super pixels) to the actual boundaries of the objects in the original image (as seen in Fig. 6.b to Fig. 6.f) and b) correct detection of the four communities (background + 3 objects) for all super pixel sizes. Recalling equation 2, the edge weight for this experiment is computed over the mean gray level of each super pixel.

Tables I e II and the plot of Fig. 8 give an insight into the influence of super pixel resolution on the computational times for all stages involved in the image segmentation process of the proposed method. In general, as the super pixel size increases, so does the overall processing time. Although fewer super pixels imply in shorter times for the Fast Greedy community detector algorithm (and also the graph assembling, if radius is kept unaltered), extra computing time is spent to make large super pixels converge to the actual object boundaries. However, a trade off between super pixel size 50 $\times$ 50 and segmentation time can be observed for this example. Notice that the shortest processing time is achieved for that size, outperforming shorter super pixel sizes (10, 15 and 20).

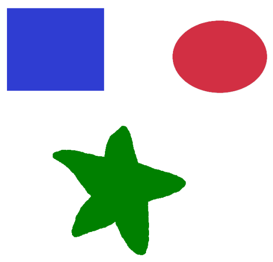


Fig. 7. Segmented image for all super pixel sizes.

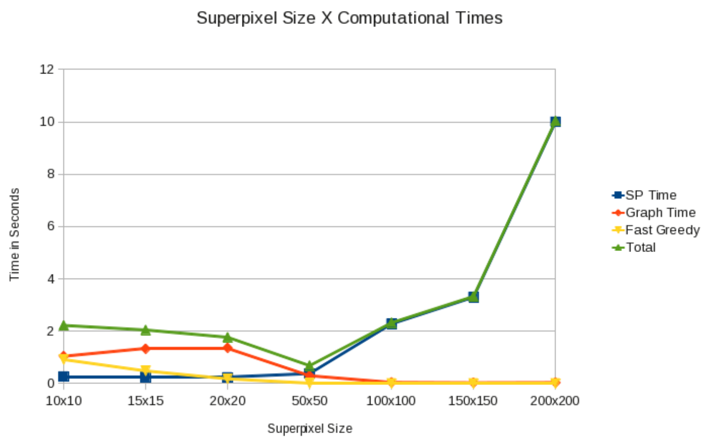


Fig. 8. Plot processing times versus Super pixel sizes.

TABLE I
PARAMETERS USED IN THE SEGMENTATION OF A SYNTHETIC IMAGE, FOR VARIOUS SUPER PIXEL SIZES. THE RADIUS IS GIVEN IN SUPER PIXELS.

	Super pixel Parameters				Network Parameters	
	Size(total)	Iter	λ_1	λ_2	Thresh	Radius
10×10	10(4900)	7	1	0.005	0.89	6
15×15	15(2209)	7	1	0,001	0.7	7
20×20	20(1225)	7	1	0.001	0.7	7
50×50	50(196)	12	1	0.001	0.7	4
100×100	100(49)	70	3	0.00009	0.7	2
150×150	150(25)	100	5.5	0.00009	0.7	2
200×200	200(16)	150	9	0.00009	0.72	2

B. *Experiment 2 - Flower*

In this experiment we show how changes in the parameters may affect segmentation of real images. We show how changes

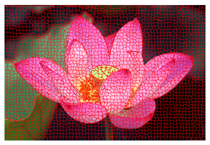
TABLE II
COMPUTATIONAL TIME OBTAINED IN THE SEGMENTATION OF A SYNTHETIC IMAGE, FOR VARIOUS SUPER PIXEL SIZES. THE TIME IS GIVEN IN SECONDS.

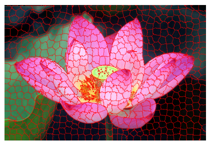
	Computational Times			
	Super Pixel	Graph	Fast Greedy	Total
10×10	0.2631	1.0427	0,9139	2,22
15×15	0.2407	1.3229	0,4815	2,04
20×20	0.2288	1.3606	0,1774	1,77
50×50	0.3785	0.2962	0,0069	0,69
100×100	2.2807	0.0419	0,0001	2,32
150×150	3.2996	0.0283	$6{,}51*10^{-5}$	3,33
200×200	5,28	0.0368	$6{,}91*10^{-5}$	5,32

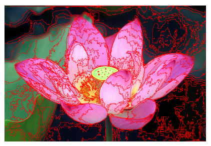
in the super pixel size may lead to incorrect segmentation and also how slight variations in the parameters of the complex network produce correct results, revealing different objects in the scene, for each distinct setup.

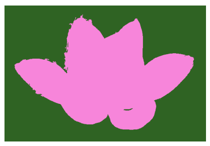
Fig. 9 illustrates an example in which changes in super pixel sizes may lead to incorrect segmentation. This is somehow expected for real images with more complex objects, with different shapes and areas smaller than the super pixel size. Fig. 9.a-f depict, respectively, the 639 × 430 original image, converged grid for super pixel size 10, size 20, size 50, segmented image for 10×10 and 20×20 super pixels and segmented image for 50×50.

(a) Original 639 × 430 Image.


(b) Converged 10 × 10 super pixel.


(c) Converged 20 × 20 super pixel.


(d) Converged 50 × 50 super pixel.


(e) Segmentation for 10 × 10 and 20 × 20 super pixels.

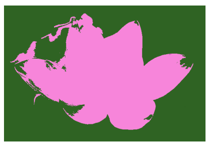

(f) Segmentation for 50 × 50 super pixel.

Fig. 9. Lotus flower segmentation. Background component in green, and flower component in pink.

Although two components have been identified for all three cases (flower and background), correct segmentation is only attained for 10 × 10 and 20 × 20 super pixels (Fig. 9.e), whereas for the the 50 × 50 case parts of the flower is grouped as background and vice versa (Fig. 9.f). The first three lines in Tables III e IV present the parameters and processing times all three setups. Notice the short processing times for all cases.

Fig. 10 illustrates the segmentation of the same lotus flower attained with parameters described in the fourth line of Tables III e IV. Notice that super pixel size remained equal to 10, but complex network parameters threshold and radius were set to 0.96 and 2, respectively. We ran the experiment with Fast Greedy, which returned 52 different communities. For

TABLE III
PARAMETERS FOR LOTUS FLOWER. THE RADIUS IS GIVEN IN SUPER PIXELS.

	Super pixel Parameters				Network Parameters	
	Size(total)	Iter	λ_1	λ_2	Thresh	Radius
10×10	10(2752)	6	1	0.0005	0.86	6
20×20	20(704)	15	2	0.0005	0.86	6
50×50	50(117)	30	1	0.000001	0.82	3
10×10	10(2752)	6	1	0.0005	0.91	2

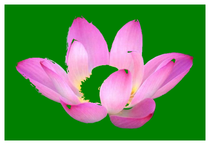
TABLE IV
COMPUTATIONAL TIMES FOR LOTUS FLOWER. THE TIME IS GIVEN IN SECONDS.

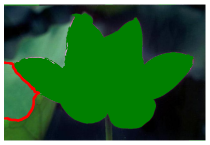
	Computational Times			
	Super Pixel	Graph	Fast Greedy	Total
10×10	0.16	0.94	0.42	1.52
20×20	0.34	0.61	0.02	0.97
50×50	0.74	0.14	0.01	0.89
10×10	0.27	0.11	0.039	0.42

clarity, communities are displayed separately, but only those with the four largest areas, which account for 99.1589% of the total image area. Fig. 10.a, Fig. 10.b, and Fig. 10.c depict, respectively, the petals, the background (accounted for the two largest communities found) and the sexual organs (yellow pistils and cup-like structure in the middle). This example shows that the method is capable of incorporating, by varying the segmentation parameters, the implicit subjectiveness of image segmentation process. Segmented images shown in Fig. 9.(e) and Fig. 10 are both correct and whether or not the core of the flower should be regarded as a legitimate object in the image depends on the observer.

C. Experiment 3 - Tucan

In this experiment we compare our method with two well know graph-based image segmentation approaches [24], [25], taking into account both processing times and segmentation accuracy. Timothee Cour et'al [25] employ the normalized cut graph partitioning framework combined with a decomposition of the image segmentation graph into different scales. Felzenswalb and Huttenlocher [24] define a predicate for measuring the evidence for a boundary between two regions (based on a greedy strategy) using a graph-based representation of the image to achieve segmentation. This method takes the parameters σ, k and min which dictates, respectively,


(a) Petals component.


(b) Background, accounting for two components.

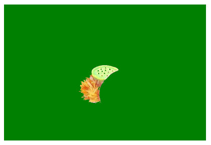
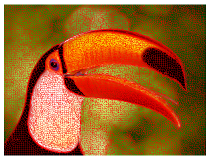
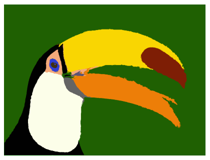

(c) Fourth community, corresponding to the sexual organs of the flower.

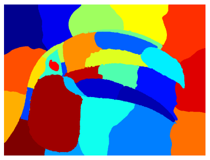
Fig. 10. Lotus flower segmentation for super pixel size 10 × 10, threshold = 0.91 and radius = 2.

a Gaussian smoothing of the image, a scale of observation (larger values of k, result in larger components) and the minimum component size. Both approaches are known to perform in nearly linear time and for that reason can also be used to segment large images.

(a) Original Image.


(b) Converged 10×10 super pixel.


(c) Our approach with the 9 largest components (super pixel=10, iterations=10, $\lambda_1 = 1$, $\lambda_2 = 0.0001$, threshold=0.95 and radius=3).


(d) Cour's method.

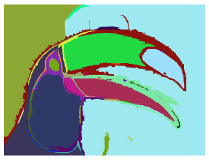

(e) Felzenswalb's method ($\sigma = 0.8$, $k = 500$ and $min = 100$).

Fig. 11. Segmentation for 1024×768 toucan image.

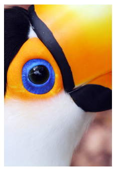
Segmentation was performed on a 1024×768 image of a toucan bird, for 10×10 super pixel size. Fig. 11 shows the original image, the converged super pixel grid and the segmented images for the three methods. A total of 133 communities have been identified by Fast Greedy, but only the nine largest communities (which account for 99.0714% of the total image area) are displayed. Notice that our methods produces crispier edges when compared with Felzenswalb's in Fig. 11.e. This is due to the correct convergence of the super pixel over the object boundaries. As for the Cour's method (Fig. 11.d), more accurate edges are produced, but uniform regions in the original (such as the background) tend to be identified as distinct communities, yielding an over segmented image. The overall processing times for our method, Felzenswalb's and Cour's are 0.17, 0.48 and 877.95 seconds, respectively. Notice that our method is almost 3 times faster than Felzenswalb's. The much higher processing time for Cour's is mainly caused by the time required to perform the multiscale normalized cut stage.

D. Experiment 4 - Extra Results

In this last experiment we present extra results obtained by our segmentation technique. As seen in the Fig. 12 the results are satisfactory, that is, the Fast Greedy community detector algorithm divides the appropriately images according to their colors.

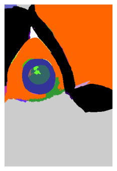
V. Conclusion and Future Work

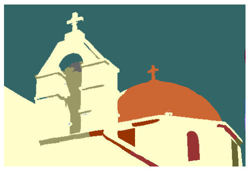
Is this paper we presented a feasible method based on complex networks and super pixels for the segmentation of


(a) Toucan 1.

(b) Church.

(c) Toucan 2.


(d) Segmented toucan 1.


(e) Segmented church.

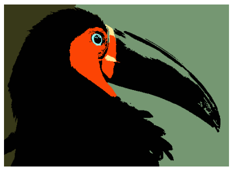

(f) Segmented toucan 2.

Fig. 12. Examples of the application of our segmentation technique.

large images. We showed that it provides accurate segmentation of large images within very low processing times. In our experiments we noticed how slight changes in parameters values (t, R, λ_1, λ_2) affect the accuracy and the quality of the segmented images. Hence, ways of estimating the "best" set of parameters values is an issue that deserves further investigation. A quantitative assessment of the segmentation produced by our method is another aspect we have to investigate more deeply.

The use of other color models, such as CIELAB, will be considered, as it is more appropriate for human eye perception and also produces precise measures for the Euclidian Distance. Therefore the CIELAB model could lead to more accurate segmentation of colored images when compared with the RGB model. Finally, we are considering the adaptation of the method for segmenting textural images, by extracting some texture signatures from super pixels.

Acknowledgment

The authors would like to thank FAPESP-Brazil and CNPQ-Brazil financing agencies.

References

[1] J. A. Noble and D. Boukerroui, "Ultrasound image segmentation: A survey," *IEEE Transactions on Medical Imaging*, vol. 25, no. 8, pp. 987–1010, 2006.

[2] H. Zhang, J. E. Fritts, and S. A. Goldman, "Image segmentation evaluation: A survey of unsupervised methods," *Computer Vision and Image Understanding, Elsevier*, vol. 110, no. 2, pp. 260–280, 2008.

[3] E. Navon, O. Miller, and A. Averbuch, "Color image segmentation based on adaptive local thresholds," *Image and Vision Computing*, vol. 23, no. 1, pp. 69–85, 2005.

[4] R. M. Haralick and L. G. Shapiro, "Image segmentation techniques," *Computer Vision, Graphics, and Image Processing*, vol. 29, no. 1, pp. 100–132, 1985.

[5] J. Shi and J. Malik, "Normalized cuts and image segmentation," *Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, pp. 888–905, 2000.

[6] S. Theodoridis and K. Koutroumbas, *Pattern recognition*, A. Press, Ed., 2003.

[7] F. A. Rodrigues, "Application of community identification methods for image segmentation," *Dynamics on Complex Networks and Applications*, 2006.

[8] L. Angelini, D. Marinazzo, M. Pellicoro, and S. Stramaglia, "Natural clustering: the modularity approach," *Journal of Statistical Mechanics: Theory and Experiment*, 2007.

[9] B. S. Oliveira, L. Zhao, K. Faceli, and A. Carvalho, "Data clustering based on complex network community detection," *IEEE Congress on Evolutionary Computation (CEC)*, pp. 2121–2126, 2008.

[10] C. Granell, S. Gómez, and A. Arenas, "Mesoscopic analysis of networks: Applications to exploratory analysis and data clustering," *Chaos*, vol. 21, no. 1, 2011.

[11] F. A. Rodrigues, G. F. de Arruda, and L. F. Costa, "A complex networks approach for data clustering," *Arxiv preprint arXiv:1101.5141*, 2011.

[12] S. Fortunato, "Community detection in graphs," *Physics Reports, Elsevier*, vol. 486, no. 3-5, pp. 75–174, 2010.

[13] M. E. J. Newman and M. Girvan, "Finding and evaluating community structure in networks," *Physical Review E*, vol. 69, no. 2, pp. 1550–2376, 2004.

[14] X. Ren and J. Malik, "Learning a classification model for segmentation," in *Proceedings on the Ninth IEEE International Conference on Computer Vision*, vol. 1, 2003, pp. 10–17.

[15] A. Levinshtein, A. Stere, K. N. Kutulakos, D. J. Fleet, S. J. Dickinson, and K. Siddigi, "Turbopixels: Fast superpixels using geometric flows," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, pp. 2290–2297, 2009.

[16] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Susstrunk, "Slic superpixels," EPFL Techical Report 149300, Tech. Rep., 2010.

[17] C. Cigla and A. A. Alatan, "Efficient graph-based image segmentation via speeded-up turbo pixels," in *17th International Conference on Image Processing*, 2010, pp. 26–29.

[18] L. F. Costa, F. A. Rodrigues, P. R. Villas Boas, and G. Travieso, "Characterization of complex networks: A survey of measurements," *Advances in Physics*, vol. 56, pp. 167–242, 2007.

[19] M. E. J. Newman, "Fast algorithm for detecting community structure in networks," *Physical Review E*, vol. 69, 2004.

[20] U. Brandes, D. Delling, M. Gaertler, R. Gorke, M. Hoefer, Z. Nikoloski, and D. Wagner, "On modularity - np-completeness and beyond," University Karlsruhe - Faculty of Informatics, Technical Report 2006-19, 2006, http://digbib.ubka.uni-karlsruhe.de/volltexte/documents/3255.

[21] A. Clauset, M. E. J. Newman, and C. Moore, "Finding community structure in very large networks," *Physical Review E*, vol. 70, no. 6, 2004.

[22] U. S. Raghavan, R. Albert, and S. Kumara, "Near linear time algorithm to detect community structures in large-scale networks," *Phisical Review E*, vol. 76, 2007.

[23] G. Csardi and T. Nepusz, "The igraph software package for complex network research," *Inter Journal, Complex Systems*, 2006, http://igraph.sourceforge.net/index.html.

[24] P. Felzenswalb and D. Huttenlocher, "Efficient graph based image segmentation," *International Journal on Computer Vision*, vol. 59, no. 2, pp. 75–174, 2010.

[25] T. Cour, F. Benezit, and J. Shi, "Spectral segmentation with multiscale graph decomposition," in *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2005.