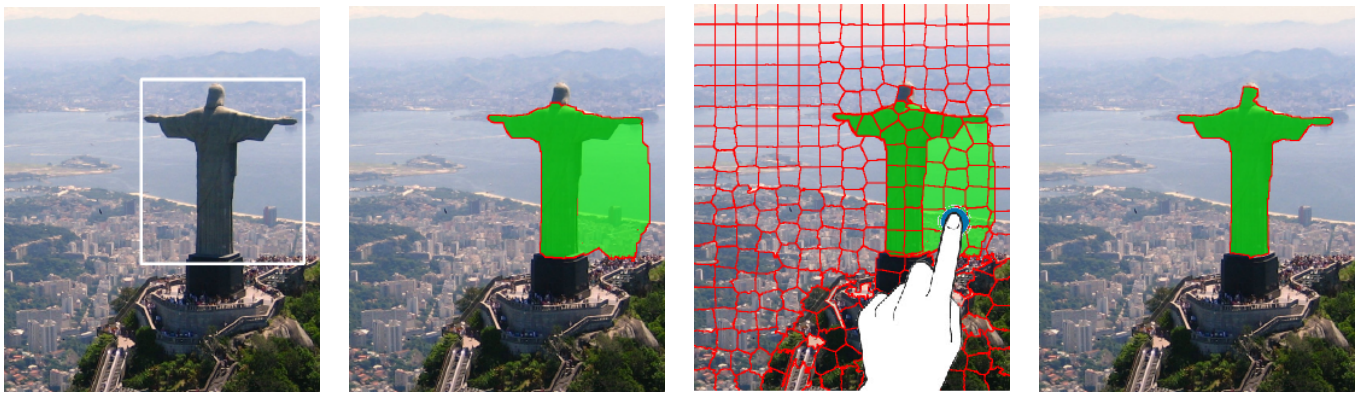
Interactive Object Class Segmentation for Mobile Devices

Ignazio Gallo, Alessandro Zamberletti, Lucia Noce
University of Insubria
Department of Theoretical and Applied Sciences, DiSTA, Varese, Italy



(a) Input image (b) GrabCut (c) SLIC (d) Final

Fig. 1. The proposed method enables the user to interactively correct the errors committed by an initial GrabCut segmentation phase by "tapping" the areas of the processed image that were wrongly labeled during the initial phase; the whole method is specifically designed to produce high quality segmentations on mobile devices.

***Abstract*—In this paper we propose an interactive approach for object class segmentation of natural images on touch-screen capable mobile devices. The key research question to which this paper tries to give an answer is: can we effectively correct the errors committed by an automatic or semi-automatic figure-ground segmentation algorithm while also providing real time feedback to the user on a low computational power mobile device? Many research works focused on improving automatic or semi-automatic figure-ground segmentation algorithms, but none tried to take advantage of the existing touch-screen technology integrated in most modern mobile devices to optimize the segmentation results of these algorithms. Our key idea is to use superpixels as interactive buttons that can be quickly tapped by the user to be added or removed from an initial low quality segmentation mask, with the aim of correcting the segmentation errors and produce a satisfying final result. We performed an extensive analysis of the proposed approach by implementing it both on a desktop computer and a mid-range Android device; even though our method is extremely simple, the results we obtained are comparable with those achieved by other state-of-the-art interactive segmentation algorithms. As such, we believe that the proposed approach can be exploited by most image editing mobile applications to provide a simple but highly effective method for interactive object class segmentation.**

***Keywords*-Interactive Image Segmentation; Object Class Segmentation; GrabCut Segmentation; Superpixel Segmentation.**

I. Introduction

Object class segmentation is defined as the task of identifying and extracting foreground objects from real world images [1]. While recent advancements in this field have pushed the accuracies of fully automatic techniques for solving this task much further than they were a few years ago [2], [3], in professional image editing it is still not possible to adopt fully automatic algorithms to select object boundaries as their current state-of-the-art results do not reach the minimum standard of accuracy demanded by professional image editors. For this reason, several interactive segmentation algorithms, mostly based on graph cut segmentation techniques, have been proposed over the last decade [4]–[16]. Among those, some can be guided by the users throughout the segmentation process in order to correct their errors, which often appear when the contrast between the object that needs to be selected and the background is extremely low or when the boundary of the object is highly jagged, with the aim of reaching final segmentation results that meet the desired degree of accuracy.

While the pipelines of some interactive segmentation methods integrated into professional softwares like Adobe Photoshop have not been published yet, there are many other published works that provide valid alternative solutions to those commercial techniques. It is particularly interesting to observe that, even though the first published interactive segmentation methods focused on providing the highest degree of accuracy while reducing as much as possible the amount of interaction required to obtain satisfying final segmentation results [15], more recent works focused on increasing the feedback with the final users by constantly showing in real time how user inputs affect the segmentation masks throughout the segmentation

process [12], [16]. In fact, real time feedback is extremely important in the image and video editing fields as the user can change the way he/she interacts with the segmentation algorithm to obtain the best results with the least effort. While most of the techniques employed in professional image and video editing are expensive in terms of required computational resources and can be effectively carried out only on powerful desktop machines equipped with high quality video resources, the proliferation of mobile devices integrating high quality cameras deeply increased the interest of the users to edit the acquired media contents directly on their mobile devices. This is proved by the constantly increasing number of image and video editing applications available in most of the mobile application stores. In the mobile environment, the expected minimum standard of accuracy for figure-ground segmentation is usually far below the requirements of professional editing; nonetheless, even the simpler and less effective currently available interactive segmentation algorithms cannot be efficiently run on mobile platforms as they usually require multiple executions of expensive energy based graph cut techniques throughout the segmentation process of a single image.

In this work we propose a novel method for interactive object class segmentation that is specifically designed to be efficiently executed on common mid-range mobile devices. We achieve this goal by performing the classic graph based energy minimization procedure only once during the whole segmentation of the given image. The low quality segmentation mask generated by this initial step can be subsequently manually refined using a set of superpixels automatically drawn over the image of interest. Those superpixels can be selected by the user to be added or removed from the current segmentation map in order to modify it by adding regions of the object of interest that were wrongly classified as background or viceversa. Since both the graph cut and the superpixel algorithm computed for the whole processed image are executed just once throughout the whole segmentation pipeline, the proposed method provides a useful user experience even when integrated into image editing mobile applications. Moreover, since the adopted superpixel algorithm can precisely follow object boundaries, the final degree of accuracy reached by the proposed method is comparable with those achieved by more sophisticated techniques. The choice of using the superpixel based refinement step is particularly effective on touch screen devices, where the user can simply "tap" the superpixels that he/she wants to add or remove from the current segmentation mask. An extensive analysis of the proposed method, including many examples of final segmentation results, are given throughout the manuscript. As in other interactive segmentation works, we evaluate the performance of our algorithm in terms of achieved overall accuracy over the number of interactions between the user and the algorithm itself. The source code for both the desktop and the mobile implementations will be available online (*http://artelab.dista.uninsubria.it/*).

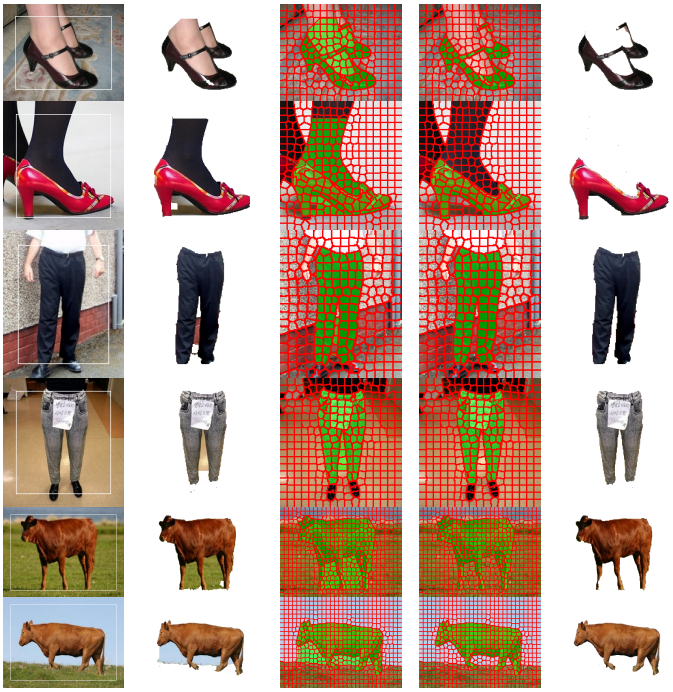


Fig. 2. Different examples of segmentation masks corrected with the use of the proposed method. For each row, from left to right: (a) the initial user drawn bounding box surrounding the object of interest within the processed image; (b) the resulting GrabCut segmentation mask; (c) the grid of superpixels and the initial GrabCut segmentation mask superimposed over the processed image; (d) "tapped" superpixels removed from the initial segmentation mask after the interactive refinement phase; (e) the final segmentation result.

A. *Related work*

According to the categorization of [12], works on interactive segmentation can be subdivided into: scribble-based, painting-based and boundary-based selection techniques.

In scribble-based selection approaches [4]–[9] the user is required to either draw a set of initial scribbles over foreground and background or to provide a bounding box that surrounds the object of interest within the processed image. These initial user-provided information enable the segmentation algorithm to infer some local color models or other characteristic features that are used to assign each pixel in the image to either foreground or background. Most of these scribble-based methods exploit the graph-cut algorithm [17] to cast the task of assigning a label to each pixel to the problem of finding a cut over a finite graph that minimizes a specific energy function. Due to its popularity, graph-cut has been improved over the years: i.e. GrabCut [15] is an iterative variant of graph-cut that obtains excellent segmentation results with a limited amount of interaction with the user; other works proposed different energy functions [1] or alternative and more effective ways of computing the weights of the graph built from the initial image [18]–[20].

Even though scribble-based techniques are widely used in the field of interactive segmentation, they are not popular in image editing; this is mainly due to the lack of feedback that

those techniques usually provide. In fact, taking a look at the "quick selection" tool integrated into Adobe Photoshop we observe that, while the user is painting the object of interest, the system interactively shows the evolution of the segmentation mask; this real time feedback greatly helps the user in obtaining better final results with the least effort. Different painting-based selection works have been published [12]–[14], among those, the recent method of [12] is able to obtain excellent results and provides the user almost instant feedbacks over the evolution of the segmentation process.

Finally, even though boundary-based selection techniques are worth mentioning [10], [11], they are no longer into use as they require the user to manually trace the boundary of the object of interest; this is a tedious task that is prone to errors when the object that needs to be segmented resides on a complex background, has a complex shape, multiple boundary components, an highly jagged boundary, etc.

The proposed work falls into the scribble-based selection techniques category as it uses GrabCut to provide the first raw segmentation of the object of interest. However, we augment the interaction with the user by superimposing over the processed image both the segmentation mask generated by GrabCut and the set of SLIC [21] superpixels computed over the image itself. The user can "tap" those superpixels to decide whether they should be added or removed from the initial segmentation mask; this is similar but opposite to [16], where the interactive segmentation process directly starts from superpixels.

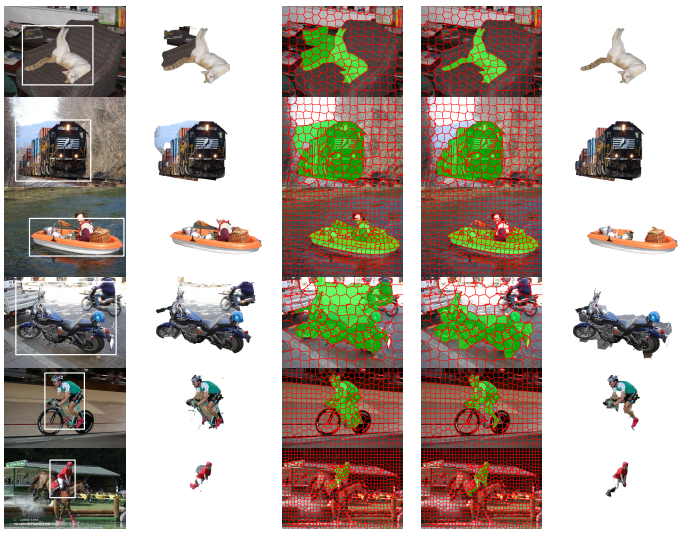


Fig. 3. Positive and negative examples of segmentation masks produced by the proposed method for some images from VOC2012 [22]. The width of each image is equal to 500 pixels. The choice of using $n = 8$, leads to nearly perfect results for the images in the first 3 rows but produces low quality segmentations for the remaining ones. The errors committed in these last examples can be corrected using the proposed zooming feature, as described in Sec. III.

II. Technical Background

In this section we briefly describe the two algorithms that represent the core of our object segmentation method: the GrabCut segmentation algorithm and the Simple Linear Iterative Clustering (SLIC) method.

A. GrabCut

GrabCut [15] is an efficient, interactive tool that uses graph cuts to perform foreground segmentation. Briefly, the algorithm works as follows: (i) initially, in order to create foreground and background regions, a user draws a rectangle (or lasso) surrounding the foreground, everything outside this rectangle will be consider as background, (ii) the algorithm does an initial labeling depending on this information, every pixel labeled in this phase is considered as hard-labeled, meaning that it cannot change its label during throughout the whole segmentation process.

A Gaussian Mixture Model (GMM) [23] is used to model the foreground and background regions; GMM learns and forms the new pixel distribution based on the initial labeling. In other words, all the pixels are labeled as probable background or probable foreground depending on their color proximity with the hard-labeled pixels.

To obtain the final segmentation mask, a graph is built from the pixel color distribution; nodes in the graph are associated to pixels in the image. In addition, two special nodes are added to the graph: the source and the sink; every pixel labeled as belonging to the foreground region is connected to the source node, while every background pixel is connected to the sink node. The color difference between neighbor pixels in the original image is used to determine the weights between the nodes representing those pixels in the graph; if there is a large difference in pixel color between two pixels, then a small weight is assigned. On the other hand, the weights between nodes and both the source and the sink are determined from the initial user drawn rectangle and denotes the probability of a pixel belonging to the foreground and background respectively.

A min-cut/max-flow [24] algorithm is used to segment the graph by determining the minimum cost cut of the graph that separates the source and sink nodes. The cost of a cut is defined as the sum of the weights of all the connections between the nodes that are separated in the cut operations. After the cut, two regions remain: the foreground region, that consists of all the pixels connected to the source node; and the background region, formed by all the nodes connected to the sink nodes.

B. Simple Linear Iterative Clustering

Simple Linear Iterative Clustering (SLIC) [21] is a simple and efficient algorithm to decompose an image into visually homogeneous regions called superpixels.

Briefly, SLIC exploits k-means [25] to generate superpixels using a combined 5-dimensional color and image plane space. Hence, each pixel is represented in the 5-dimensional $labxy$ space, taking into account both its color in the CIE LAB space $[l, a, b]$ and its position $[x, y]$ in the image. The $labxy$ space is

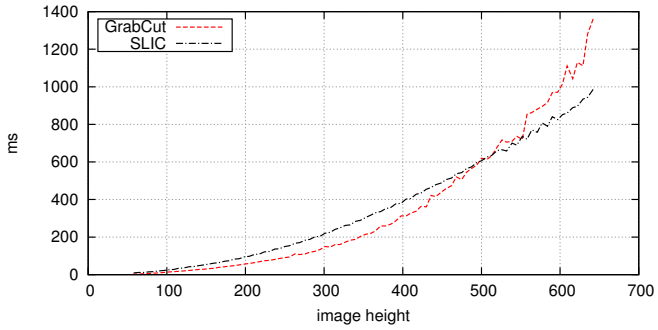


Fig. 4. Comparison between computational times required to complete the execution of GrabCut and SLIC while varying the size of the input image. The number of superpixels for the smallest dimension of the processed image was set to 10. In the mobile implementation, to provide real time feedback to the user, the image needs to be rescaled before being processed by GrabCut and SLIC.

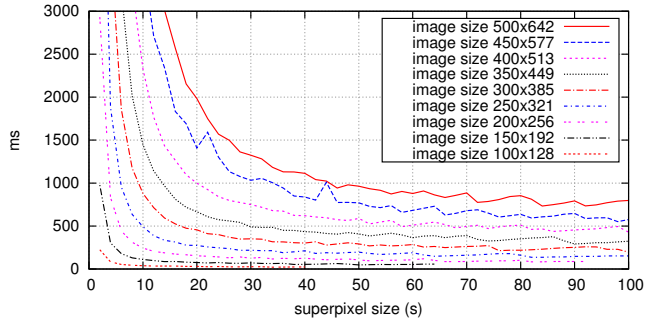


Fig. 5. Time required by SLIC to compute the superpixels for a given input image while varying S. Note that the algorithm becomes slower as the size of the superpixels increases; for this reason, in order for the proposed to provide a real time feedback on mobile devices, high values must be given to S. As explained in Sec. III, to obtain optimal segmentation results, the superpixel size may be decreased for highly textured regions of the object of interest.

finally exploited to generate superpixels by clustering pixels on the basis of their color similarities and proximity values within the plane itself.

In its default configuration, SLIC has one parameter, k, that represents the number of desired superpixels. At its first step, the clustering procedure defines k initial centroids as $C_i = [l_i, a_i, b_i, x_i, y_i],\ i \in [1, k]$, sampled on a regular grid. The distance between the centroids in the grid is defined as $S = \sqrt{\frac{N}{k}}$, where N is the number of pixels in the processed image. Defining S as such is mandatory in order to produce equally sized superpixels over the whole image. Since the Euclidean distance in $labxy$ space will cause inconsistencies in clustering, a distance measure D that considers superpixels size is introduced; it is defined as follows:

$$d_c = \sqrt{(l_j - l_i)^2 + (a_j - a_i)^2 + (b_j - b_i)^2} \tag{1}$$

$$d_s = \sqrt{(x_j - x_i)^2 + (y_j - y_i)^2} \tag{2}$$

$$D' = \sqrt{\left(\frac{d_c}{N_c}\right)^2 + \left(\frac{d_s}{N_s}\right)^2} \tag{3}$$

$$D = \sqrt{{d_c}^2 + \left(\frac{d_s}{S}\right)^2 \cdot m^2} \tag{4}$$

where Eq. 1 and 2 represent the distances in the color space d_c and in the image spatial space d_s respectively. The combination of the 2 distances into a single measure is reported in Eq. 3, where the 2 proximities are normalized by their maximum values: N_c for the color space and N_s for the spatial distance. The final formula for the distance D, shown in Eq. 4, is obtained by fixing the maximum spatial distance equals to the sampling interval $N_s = S = \sqrt{\frac{N}{k}}$ and by assigning a constant m to the maximum color distance N_c. In the CIE LAB color space, m can vary in the range $[1, 40]$. To reduce the chance of centering a superpixel on an edge, the k cluster centers are moved to the lowest gradient position in a 3×3 neighborhood.

For each centroid C_i, the clustering algorithm searches for similar pixels in a region $2S \times 2S$ around it. This is due to the fact that the spatial extend of any cluster is approximately $S \times S$. This approach is the key of the efficiency of SLIC, as it reduces superpixel search regions and makes SLIC's complexity independent from the number of superpixels. Each pixel in the image is associated to the nearest cluster center whose search area overlaps it. When all the pixels belongs to the nearest cluster, a new center is computed as the mean of the $[labxy]$ vector of all the pixels assigned to the cluster. A residual error E between the new cluster center position and the previous center position is computed using $L2$ norm. These steps are repeated until the error E is under a defined threshold that represents the error convergence.

III. Proposed Method

In the proposed method, as in most of the interactive image segmentation algorithms proposed in literature, a user first selects the object of interest to be extracted from the processed image either by drawing strokes on the foreground and/or background or by defining a rectangle that surrounds the object of interest itself within the given image. This user-provided information is then exploited by the segmentation algorithm to produce an initial segmentation mask for the processed image. In some interactive methods, the latest can be manually edited by the user to correct possible mistakes; in our work, this last phase is carried out using the superpixels generated by SLIC as buttons that can be interactively turned "on" or "off" in order to add or remove corresponding areas of the processed image that were wrongly added or omitted from the initial segmentation mask. To the best of our knowledge, there are no other works in literature that propose SLIC [21] as a tool to correct the initial segmentation errors; this seemingly simple approach becomes very useful on touch screen devices because, with just a few taps, the user can correct almost every initial error.

The interactive image segmentation algorithm proposed in this work is GrabCut [15]. To obtain an initial segmentation of the object of interest, users are required to draw a rectangle R using their fingers on the screen of its mobile device. The information provided by the rectangle is exploited by GrabCut as described in Sec. II to determine the color distributions of foreground and background regions to provide an initial "raw" segmentation of the object of interest. This step can be performed multiple times and the resulting segmentation is updated in real time on the device; this enables the user to obtain an initial mask that is fairly correct and avoids excessive strain during the subsequent refinement phase.

Once the initial GrabCut segmentation mask satisfies the user, he/she can correct its errors using SLIC superpixels as areas to be manually marked as either belonging to foreground or background. In this phase, the grid of superpixels is superimposed, together with the initial segmentation map, over the processed image; this enables the user to see in real time how his actions are affecting the final segmentation result. In Fig. 2, we provide some examples of the results obtained using GrabCut and the subsequent manual SLIC-based refinement phase; it is important to note that in all those examples we have always used the same size for SLIC's superpixels. Provided that the size of the images in the first 4 lines is equal to 200×160 and the size of the images in the last 2 lines is equal to 320×240, we observe that, in general, the size of the processed image deeply affects the maximum number of SLIC superpixels that can be defined over the image itself. When the number of superpixels is extremely high, the refinement phase generates high quality segmentation results; however, considering that our method is designed to be used on mobile devices equipped with small screens, having a high number of superpixels can sensibly increase the complexity of the refinement phase from a user point of view. To better highlight this concept, in Fig. 6 we provide 4 different examples of potential segmentation masks obtained using proposed method for an image from the Weizmann Horses dataset [26], while varying the size of SLIC's superpixels for the same initial segmentation mask generated by GrabCut for a given user drawn rectangle. In details, the first row shows the initial GrabCut segmentation phase, while the remaining ones show different results of the manual refinement phase; all the examples were refined by the same operator. It is clear that small superpixels lead to highly precise refinements of the initial segmentation masks but also increase the time required for the operator the fix the initial errors as there are large number of superpixels that need to be manually "tapped"; on the other hand, large superpixels lead to bad refinements that often worsen the initial results produced by GrabCut.

From these previous considerations, it becomes obvious that is mandatory to define a set of rules to automatically determine for every processed image the size of superpixels that will provide the best compromise between complexity and accuracy of the refinement phase. In the proposed method, the size S of SLIC superpixels is automatically computed on the basis of the size of the initial user drawn rectangle R. As reported in Sec. IV, we found that the optimal value for S can be computed as follows:

$$S = \frac{R_{min}}{n} \tag{5}$$

where R_{min} denotes the smallest dimension of R and n is the number of superpixels in which the dimension R_{min} of the processed image needs to be splitted.

In order to provide a method that is extremely interactive and computationally inexpensive, at each tap of the user on the touch screen only the pixels belonging to the tapped superpixel are updated and added or removed from the current segmentation mask. It is important to observe that, whenever the user is not satisfied by the value of S automatically computed by the algorithm or wants to obtain a more accurate segmentation for some parts of the object of interest, it is possible for him to zoom on a region of the object of interest to automatically trigger a new computation of the superpixels over that area using the same initial value of n; this means that the same number of superpixels defined over the initial processed image will be drawn over the zoomed region of interest. This enables the user to obtain high quality segmentations of those regions of the object of interest that reside on highly textured backgrounds, have complex boundary components, etc. Thanks to this feature the proposed method can obtain excellent results even when the value of n is fixed to 8.

TABLE I
COMPARISON BETWEEN OVERALL SEGMENTATION ACCURACIES ACHIEVED BY THE INITIAL GRABCUT SEGMENTATION AND THE SUPERPIXEL-BASED REFINEMENT PHASE. THE THIRD COLUMN DENOTES THE NUMBER OF SUPERPIXELS THAT HAD TO BE MANUALLY CHANGED BY THE OPERATOR DURING THE REFINEMENT PHASE.

Dataset	S	# of tapped superpixels	OA (%)	
			GrabCut	Proposed
Oxford Flower 17	50	9	96.24	99.78
Weizmann Horses	9	9	93.71	98.77
Drezzy	8	16	91.79	98.64

IV. EXPERIMENTS

The proposed algorithm was tested using 3 different datasets, each one contains images differing in size, color and texture profile. In particular, we use: Oxford Flower 17 [27], Weizmann Horses [26] and Drezzy [28]. All the previously mentioned datasets come with foreground/background segmentation masks associated with the object of interests appearing in the images. In the following paragraphs, we briefly motivate the reasons why we decide to employ those datasets in our experiments.

Oxford Flower 17: Composed by 17 flower classes with 80 images each. We only consider the subset of 848 images labeled for segmentation. The objects are always predominant in the scenes. Poses, illuminations and colors vary among all the images. More than one object may appear inside the same image. Since this dataset is composed of high-resolution

images, it becomes useful in identifying whether the computational complexity of the proposed method is sufficiently low to provide a good user experience even when processing images acquired using high quality phones cameras.

Weizmann Horses: This dataset contains 328 side-view color images of horses that were manually segmented. The images were randomly collected from the web. Each image contains at most one foreground object. Even though this dataset is mainly composed of low-resolution images, it is particularly relevant for our experiments as the object of interests appearing in its images are often placed on complex, highly textured, backgrounds; this helps us in determining whether the formula of Eq. 5 is appropriate. In fact, horses usually have thin legs compared to their bodies, as such, it becomes difficult to obtain good segmentation results when using an inappropriate size for the superpixels during the refinement phase.

Drezzy Dataset: This dataset is composed by 2068 images whose size varies from 200×200 to 100×100 pixels. From this dataset we used only the class "Man clothing" containing 150 images. The images were collected from the web and each image contains at most one foreground object. The characteristics of this dataset make problematic the segmentation with the GrabCut algorithm and then in many cases the segmentation for these images must be made integrally with the SLIC. As shown in the evaluation tables, this dataset is particularly interesting as the contrast between the object of interests and their respective backgrounds is extremely low; this causes the initial GrabCut segmentation phase to completely fail and therefore the final segmentation masks need to be completely manually defined during the refinement phase. This enables us to evaluate the performance of the SLIC-based refinement procedure independently from the initial segmentation step.

A. Evaluation Metric

We evaluated the effectiveness of the proposed method by assessing the consistency between the segmentation masks it generates and the ground truth segmentation masks from the different previously mentioned datasets using the Overall Accuracy OA [29] metric. The OA belongs to the category of pixel wise evaluation measures and describes the proportion of pixels of each class that are correctly classified out of the pixels belonging to that class; it is defined as follows:

$$OA = \frac{tp + tn}{tp + tn + fp + fn} \tag{6}$$

where the terms true positives (tp), true negatives (tn), false positives (fp), and false negatives (fn) denote whether the values assigned by the evaluated algorithm to the pixels in the output segmentation mask correspond to the ones provided in the ground truth annotations. The higher the value of OA, the better are the segmentation masks produced by the algorithm being evaluated.

B. Results and Discussion

In Table I we report the performance achieved by the proposed method for the 3 previously introduced datasets. The first column shows the average superpixel size S automatically selected by the proposed approach using Eq. 5, while the second column shows the average number of superpixels that needs to be "tapped" by the user to correct the errors committed by the initial segmentation phase based on GrabCut. The last two columns compare the OA values obtained by the initial GrabCut segmentation phase with the ones obtained after the superpixel-based manual refinement phase. From the obtained results, it is possible to observe that the proposed technique outperforms GrabCut on the evaluated datasets after a low number of manual refinement iterations; obviously, the number of superpixels that needs to be changed strictly depends on the magnitude of the errors committed by GrabCut: the greater the error of the initial phase, the greater the number of superpixels that needs to be "tapped" to obtain a satisfying final result. The effectiveness of the superpixel-based refinement phase is proved by the results achieved for the Drezzy dataset; in fact, even though the segmentation masks generated by GrabCut for that dataset are highly inaccurate, the subsequent manual refinement phase can produce excellent accuracies.

A further experiment was performed to evaluate the computational complexity of the proposed algorithm, all tests were conducted on a standard desktop computer (Intel Core i3, 4GB RAM). The computational complexity of the proposed algorithm strongly depends on the image size and the size selected for SLIC's superpixels. As shown in Fig. 4 and Fig. 5, the proposed method must be properly configured to prevent it from becoming too slow. In details, Fig. 4 highlights the quadratic complexity of the two algorithms we employ (GrabCut and SLIC) when fixing the number of superpixels for the shortest dimension of the processed image. Nonetheless, thanks to both the automatic selection of S and the possibility of zooming into particular regions of the processed image to obtain better segmentations of those areas, the proposed algorithm can be efficiently implemented and executed on modern mobile devices and provides a great user experience.

To obtain the curves of Fig. 5 we used a 500×642 image containing a centered foreground object surrounded by a 460×602 user drawn rectangle. In this configuration, we applied GrabCut before SLIC algorithm while varying the value for S in the following range $\{2, 4, \ldots, 100\}$. This experiment was repeated varying the scale of the original image by a factor of 0.1 at each run (from 460×602 to 100×128 pixels). Based on the results of this experiments, we decided to set $n = 8$ in order to keep the execution time of SLIC under acceptable values in every possible situation. As shown in Fig. 3 this empirical of n leads to optimal results even when processing extremely complex object of interests even when not using the zooming feature mentioned in Sec. III.

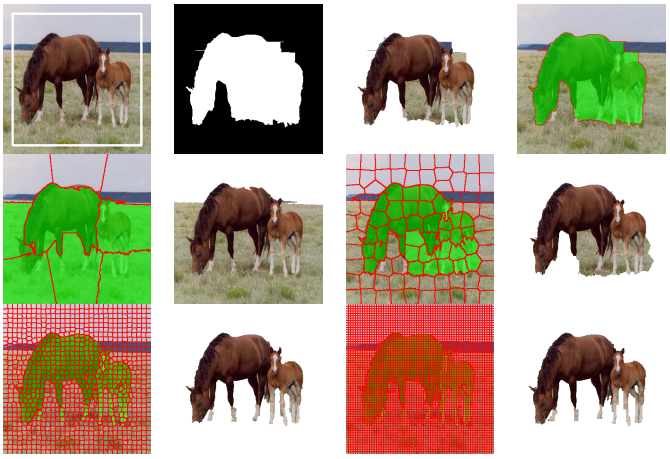


Fig. 6. Examples of segmentation results produced by the proposed method while varying the number of superpixels employed in the refinement phase. An increase in the number of superpixels leads to better results but also increases the number of elements that needs to be manually "tapped".

V. Conclusion

In this work, we have proposed a novel method for interactive segmentation that makes extensive use of the touch screen technology integrated in most modern mobile devices to provide a great user experience while still being able to produce final segmentation results that are comparable to those achieved by other more complex state-of-the-art algorithms proposed in the last decade. We paired an initial GrabCut-based segmentation phase that starts from the manual selection of a bounding box surrounding the object of interest within the processed image with a SLIC-based refinement phase that enables the user to manually "tap" on those superpixels that were wrongly added or omitted from the initial GrabCut segmentation map. We addressed the problem of automatically determining the size of superpixels that grants the best compromise between accuracy of the refinement phase and computational complexity of the proposed pipeline. To overcome the issues associated with the quadratic computational complexities of the algorithms employed in the proposed pipeline and obtain an interactive algorithm that provides real time feedback to the user, we decided to exploit the pinch-to-zoom functionality integrated in most modern devices to enable the user to focus his refinement efforts on specific portions of the object of interest. Even though the proposed method is very simple and can be easily implemented in just a few lines of code, it is extremely effective even when compared to other popular interactive segmentation algorithms. Finally, the manual superpixel-based refinement phase always improves the segmentation results of the initial phase, this is not an obvious consideration since, as proved in our experiments, a wrong computation of either the size of the superpixels or a suboptimal choice of the superpixel algorithm to be used during the refinement phase can worsen the result produced by GrabCut for difficult objects.

References

[1] V. Kolmogorov and R. Zabih, "What energy functions can be minimized via graph cuts?" *IEEE PAMI*, vol. 26, no. 2, pp. 147–159, 2004.

[2] D. Küttel and V. Ferrari, "Figure-ground segmentation by transferring window masks," in *CVPR*, 2012.

[3] J. Carreira and C. Sminchisescu, "Constrained parametric min-cuts for automatic object segmentation," *IEEE PAMI*, vol. 34, no. 7, pp. 1312–1328, 2012.

[4] X. Bai and G. Sapiro, "A geodesic framework for fast interactive image and video segmentation and matting," in *ICCV*, 2007.

[5] J. Wang and M. F. Cohen, "An iterative optimization approach for unified image segmentation and matting," in *ICCV*, 2005.

[6] L. Grady, "Random walks for image segmentation," *IEEE PAMI*, vol. 28, no. 11, p. 17681783, 2006.

[7] D. Lischinski, Z. Farbman, M. Uyttendaele, and R. Szeliski, "Interactive local adjustment of tonal values," *ACM Graphics*, vol. 25, no. 3, pp. 646–653, 2006.

[8] Y. Li, E. H. Adelson, and A. Agarwala, "Scribble-boost: Adding classification to edge-aware interpolation of local image and video adjustments," in *EGSR*, 2008.

[9] X. An and F. Pellacini, "AppProp: all-pairs appearance-space edit propagation," *ACM Graphics*, vol. 27, no. 3, pp. 1–9, 2008.

[10] E. N. Mortensen and W. A. Barrett, "Intelligent scissors for image composition," in *SIGGRAPH*, 1995.

[11] M. Kass, A. Witkin, and D. Terzopoulos, "Snakes: Active contour models," *IJCV*, vol. 1, no. 4, pp. 321–331, 1987.

[12] J. Liu, J. Sun, and H.-Y. Shum, "Paint selection," *ACM Graphics*, vol. 28, no. 3, pp. 691–697, 2009.

[13] J. Chen, S. Paris, and F. Durand, "Real-time edge-aware image processing with the bilateral grid," *ACM Graphics*, vol. 26, no. 3, p. 103, 2007.

[14] D. R. Olsen, Jr. and M. K. Harris, "Edge-respecting brushes," in *UIST*, 2008.

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

[16] O. Sener, K. Ugur, and A. A. Alatan, "Error-tolerant interactive image segmentation using dynamic and iterated graph-cuts," in *IMMPD*, 2012.

[17] Y. Boykov and M. P. Jolly, "Interactive graph cuts for optimal boundary & region segmentation of objects in n-dimages," in *ICCV*, 2002.

[18] F. Calderero and F. Marques, "Region merging techniques using information theory statistical measures," *IEEE IP*, vol. 19, no. 6, pp. 1567–1586, 2010.

[19] L. G. S. D. Liu, Y. Xiong and K. Pulli., "Robust interactive image segmentation with automatic boundary refinement," in *ICIP*, 2010.

[20] J. Ning, L. Zhang, D. Zhang, and C. Wu, "Interactive image segmentation by maximal similarity based region merging," *PR*, vol. 43, no. 2, pp. 445–456, 2010.

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

[22] M. Everingham, L. Gool, C. K. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes challenge," *IJCV*, vol. 88, no. 2, pp. 303–338, 2010.

[23] D. Reynolds, "Gaussian mixture models," 2009, pp. 659–663.

[24] M. Stoer and F. Wagner, "A simple min-cut algorithm," *JACM*, vol. 44, no. 4, pp. 585–591, 1997.

[25] J. B. MacQueen, "Some methods for classification and analysis of multivariate observations," in *BSMSP*, 1967.

[26] E. Borenstein, "Combining top-down and bottom-up segmentation," in *CVPR*, 2004.

[27] M.-E. Nilsback and A. Zisserman, "Delving deeper into the whorl of flower segmentation," *IVC*, vol. 28, no. 6, pp. 1049–1062, 2010.

[28] S. Albertini, I. Gallo, M. Vanetti, and A. Nodari, "Learning object segmentation using a multi network segment classification approach," in *VISAPP*, 2012.

[29] D. L. Olson and D. Delen, *Advanced Data Mining Techniques*, 2008.