Revisiting Component Tree Based Segmentation Using Meaningful Photometric Informations

Michał Kazimierz Kowalczyk [1], Bertrand Kerautret[1], Benoît Naegel[2], and Jonathan Weber[1]

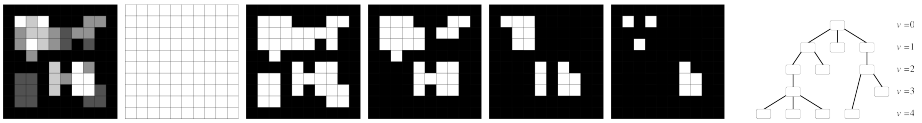
[1] Université de Lorraine, LORIA, UMR 7503, 54506, France
kerautre@loria.fr,mkk@ekk.pl,jonathan.weber@loria.fr
[2] LSIIT, Université de Strasbourg, LSIIT, UMR 7005 67412, France
b.naegel@unistra.fr

Abstract. This paper proposes to revisit a recent interactive segmentation algorithm based on an original image representation called the component-tree [1]. This method relies on an optimisation process allowing to choose a segmentation result fitting at best some image markers defined by the user. We propose different solutions to improve the efficiency of the method, in particular by including meaningful photometric informations and by assessing automatically the user parameter α.

1 Introduction

As segmentation is an ill-posed problem, interactive segmentation is considered as an efficient way for obtaining good results according to the user need. Moreover, recent devices with camera and tactile interface offer new perspectives and have shown their relevance for interactive segmentation [2]. Due to their restricted computing and memory capacities, segmentation algorithms implemented on tablets have to be computationnally and memory inexpensive. Besides, tactile interface do not provide precise markers and majority of interactive segmentation method based on markers are very sensitive to marker quality. In this context, interactive segmentation based on component tree seems particulary relevant [1] since it is a fast, efficient and robust to rough markers segmentation method. But this algorithm fails to segment some type of object and needs a parameter α which can be tedious to set. So, in this article we propose some image preprocessing to improve object segmentation and introduce an automatic way of setting α based on meaningful scales [3]. Furthermore, we have implemented the proposed method on tactile tablet.

In the next section, we recall the principles of the component-tree based segmentation. We then propose some ways of including image information in component-tree based segmentation and illustrate their relevancy by some experiments.



(a) (b) (c) (d) (e) (f) (g)

Fig. 1. (a) A grey-level image $I : [\![0, 9]\!]^2 \to [\![0, 4]\!]$ (from 0, in black, to 4, in white). (b–f) Threshold images $X_v(I)$ (white points) for v varying from 0 (b) to 4 (f). (g) The component-tree of I. Its levels correspond to increasing thresholding values v. The root (*i.e.*, the upper node located at the level $v = 0$) corresponds to the support $E = [\![0, 9]\!]^2$ of the image.

2 Component-tree based segmentation

In the following, we summarize the segmentation algorithm based on the component-tree structure described in [1].

2.1 Component-tree: definition

A discrete grey-level image can be defined as a function $I : E \to V$, with $E \subseteq \mathbb{Z}^n$ a finite connected set and $V = [V_{min}, V_{max}] \subseteq \mathbb{Z}$ the finite set of values of I.

Let $X \subseteq \mathbb{Z}^n$ be a non-empty set. The set of connected components of X is denoted by $\mathcal{C}[X]$. Let X_v the thresholding function defined for any $v \in V$ by: $X_v(I) = \{p \in E \mid I(p) \geq v\}$. Let $\mathcal{K} = \bigcup_{v \in V} \mathcal{C}[X_v(I)]$ be the set of all the connected components obtained from the different thresholdings of I at values $v \in V$. The *component-tree* of I is the rooted tree $T = (\mathcal{K}, L, R)$ such that:

(i) $\mathcal{K} = \bigcup_{v \in V} \mathcal{C}[X_v(I)]$,
(ii) $L = \{(X, Y) \in \mathcal{K}^2 \mid Y \subset X \wedge \forall Z \in \mathcal{K}, Y \subseteq Z \subset X \Rightarrow Y = Z\}$,
(iii) $R = \sup(\mathcal{K}, \subseteq) = X_{V_{min}}(I) = E$.

The elements of $\mathcal{K}$ (resp. of L) are the *nodes* (resp. the *edges*) of T. The element R is the *root* of T. An example of component-tree is illustrated in Fig. 1.

2.2 Segmentation based on component-tree

Based on this structure, a segmentation of I can be achieved by selecting a subset of nodes $\mathcal{K}' \subseteq \mathcal{K}$ and computing the associated binary image S defined as the set $S = \bigcup_{N \in \mathcal{K}'} N$ (see Fig. 2).

Let $G \subseteq E$ be a binary image: G can be, for example, a marker manually drawn on an original image I. The segmentation procedure consists in selecting automatically the subset $\mathcal{K}'$ of nodes in order to obtain the binary image S which is "closest" to G. More formally, given a similarity criterion d, the segmentation process is an optimisation problem consisting in determining the set:

$$\widehat{\mathcal{K}} = \arg \min_{\mathcal{K}' \subseteq \mathcal{K}} \Big\{ d\Big(\bigcup_{N \in \mathcal{K}'} N, G \Big) \Big\}.$$

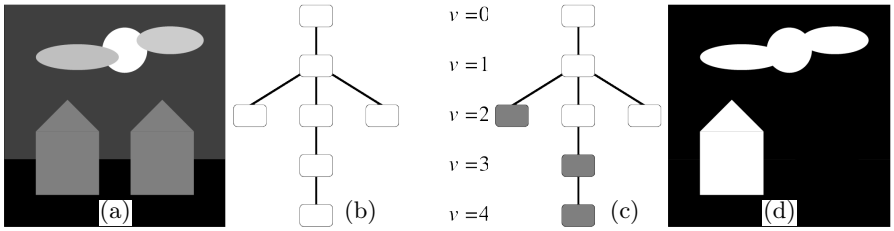


Fig. 2. (a) Example image and (b) its component-tree T. (c) In grey: subset $\mathcal{K}'$ of nodes selected from T. (d) The associated binary image S.

Given a parameter $\alpha \in [0, 1]$, the pseudo-distance d^α taking into account the amount of false-positives/negatives between the marker G and the binary segmentation $S = \bigcup_{N \in \mathcal{K}'} N$ is defined for any $X, Y \subseteq E$ by: $d^\alpha(X, Y) = \alpha.|X \setminus Y| + (1-\alpha).|Y \setminus X|$. The optimisation scheme is based on this pseudo-distance which constitutes an efficient similarity measure between two binary images. It can be efficiently solved by using dynamic programming (see [1] for more details).

The segmentation procedure consists in (i) the manual delineation of a rough marker inside the object to segment and (ii) the interactive setting of the α parameter in order to choose the "best" segmentation. Indeed, the quality of the result is highly dependent of the α parameter as illustrated in Fig. 3.

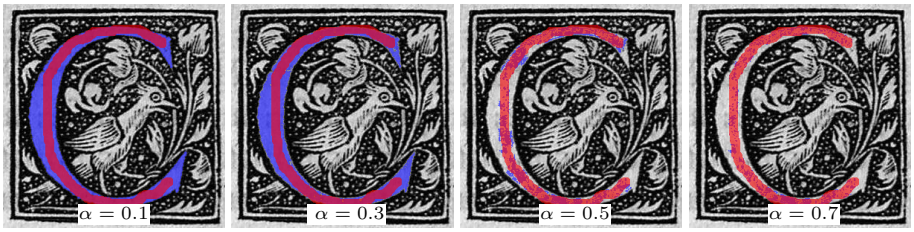


Fig. 3. Illustration of the influence of the α parameter on the segmentation quality (represented in blue). The marker G is superimposed in red.

3 Including Meaningful Image Information in Component Tree

The segmentation method previously described suffers of two main drawbacks which limits its efficiency in real-world image applications. First, the method is devoted to the extraction of objects which are represented by a node of the component-tree: this limits the extraction process to bright objects surrounded by dark background. A second problem is the manual setting of the parameter α which can be tedious and time consuming.

In this paper, we propose to address these drawbacks by: (i) exploiting the contrast information given by the marker in order to automatically compute the most relevant component-tree structure enabling to extract the object and (ii) determining automatically the most relevant α parameter based on gradient information and contour smoothness.

3.1 Exploiting photometric informations

We propose different strategies to exploit the photometric informations given by the marker in order to address the first problem.

Automatic component tree selection The component-tree based segmentation is dedicated to extract objects belonging to nodes, *i.e.* bright objects on dark background. In order to deal with the opposite case (*i.e.* dark objects on bright background) we need to compute the dual component-tree (the component-tree of the negative image). We propose an efficient strategy to determine automatically the nature of the tree.

Let $\mu_I(G)$ be the mean value of the points of the image I belonging to the marker G: $\mu_I(G) = \frac{1}{|G|} \sum_{p \in G} I(p)$.

Let N_G be a neighbourhood region associated to the marker G. Let $c_I(X, Y)$ be the contrast function defined by:

$$c_I(X, Y) = \mu_I(X) - \mu_I(Y),$$

with $X, Y \subseteq E$ and $X \cap Y = \emptyset$. If $c_I(G, N_G) < 0$, it means that the marked object is possibly darker than its neighbourhood: in this case the dual component-tree of source image I is computed. In this paper we consider the neighbourhood defined as the complement of G, *i.e.* $N_G = G^c = E \setminus G$.

Image values shifting In order to deal with objects which corresponds neither to component-tree nor to dual component-tree nodes, we propose to transform the source image. Let $s_o(I)$ the function defined for all $p \in E$ by:

$$s_o(I)(p) = V_{max} - |(I(p) - o)|\,,$$

where $o \in V$ is a reference colour defined by the marker G (*i.e.* $o = \mu_I(G)$). This function enables (as in [4]) to enhance pixels having a colour close to the marker, therefore favouring the presence of the marked object in a leaf of the associated component-tree.

Exploiting colour information Finally, we propose a way of working with colour images (*i.e.* images having their values in $V \subseteq \mathbb{Z}^k$). Indeed, the component-tree algorithm is devoted to grayscale images: its extension to colour images is not trivial and has been addressed in [4] and [5]. One possibility consists in transforming a colour image in a grayscale one in order to obtain a total ordering on the set of values.

For this purpose, we propose to transform the colour image by taking into account the most significants colour channels with respect to the local contrast between the marker and its background. More formally, let m be the ponderation function defined for each $p \in E$ by:

$$m(I)(p) = \frac{w_1 * I_1(p) + w_2 * I_2(p) + \ldots + w_n * I_k(p)}{w_1 + w_2 + ... + w_k},$$

where the $I^k_{i=0}$ represent the k colour channels of I and the $w^k_{i=0}$ are the weights associated to each channels.

Let r_i be the function giving, for each channel I_i, a contrast measure between the marker and its neighborhood: $r_i(G) = c_{I_i}(G, N_G)^2$.

We take $w_i = r_i(G)$ in order to favour, in the ponderation function, the channels in which the marker has a high contrast with respect to its neighborhood. Our approach allows to use RGB as well as CMYK and HSL/HSV (after translation hue to range $[0, 255]$) colour models. We can also remove channels that are not significant enough (by imposing for example a minimal threshold on the weights). The full preprocessing algorithm includes the three strategies described above and is illustrated in Fig. 4.

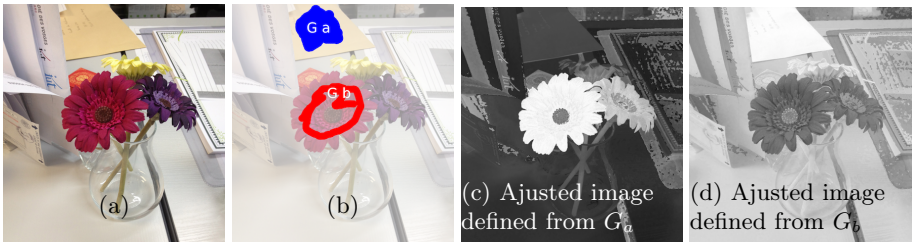


Fig. 4. Illustration of the image preprocessing based on image marker photometric informations. Two different markers G_a and G_b were used to experiment the image transformation from the source image (a). The resulting images are shown in (c,d).

3.2 Automatic setting of the alpha parameter

As illustrated in the previous section 2, the quality of the resulting segmentation is very sensitive to the choice of the parameter α . To remove this limitation, we propose to define a criterion allowing to assess automatically the best parameter α. In the same spirit as the deformable model based algorithms, we propose to take into account the image gradient intensity and the smoothness contour quality.

Meaningful Scale detection on discrete contour To evaluate the degree of contour quality we use the method of the meaningful scale detection which allows

to detect automatically what is the relevant scale of a given contour [3]. This method is based on the multiscale analysis of the length of the digital straight segments primitive. Note that this method can also be evaluated online [6]. Fig. 5 illustrates such meaningful scale detection obtained on synthetic and real image.

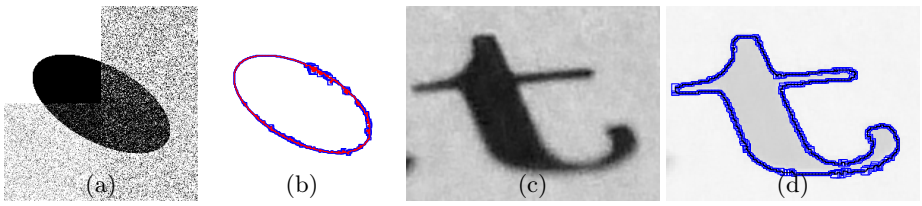


Fig. 5. Illustration of the meaningful scale detection on two images tests (a,c). The resulting meaningful scale is illustrated on images (b,d). For each contour point we display a blue box of size equals to the detected noise level.

Note that given a maximal scale S_{max} the meaningful scale detection can be considered as parameter free. In the following we will denote by $\eta(p)$ the amount of noise level given for a contour point p. This noise level values will be include in $[1, S_{max}]$.

Image gradient intensity We also propose to choose, among all the possible segmentations induced by the α parameter, the one whose contour coincides with regions of high gradient intensity.

Let $W \subseteq E$ a segmentation result, and $O \subseteq W$ the set of contour pixels of W. Let $Y = \{O_1, O_2, \ldots, O_n\}$ the set of all connected contours of the result and D the longest contour of Y (*i.e.* $|D|$ is maximal for all $O_i \in Y$). Let $j(O_i)$ be weight function defined for a contour as follows:

$$j(O_i) = \frac{|D|}{(|D| - |O_i| + 1) * N_{MP}(D)},$$

where $N_{MP}(D)$ is the number of meaningful pixels p of D (the number of pixels of D for which $\eta(p) = 1$). We have tested several criteria and obtained the best results by choosing the segmentation minimizing:

$$k = \frac{\sum_{O_i \in Y} |O_i|}{(\sum_{O_i \in Y} g_{max}(O_i) * j(O_i))}$$

where $g_{max}(O_i) = \sum_{p \in O_i} (\max\{g(p') | p \in O_i \ \wedge max(|x_p - x_{p'}|, |y_p - y_{p'}|) \leq 1\})$ and g is the image gradient. In this work, the gradient of I is approximated using the Sobel operator.

4 Experiments and comparisons

To evaluate the efficiency of the proposed solutions, we have first experimented the contrast feature improvements in comparison to the initial component tree

algorithm. Fig. 6 shows the results obtained on two real images. We can clearly see that the proposed contrast based method really improves the original approach.

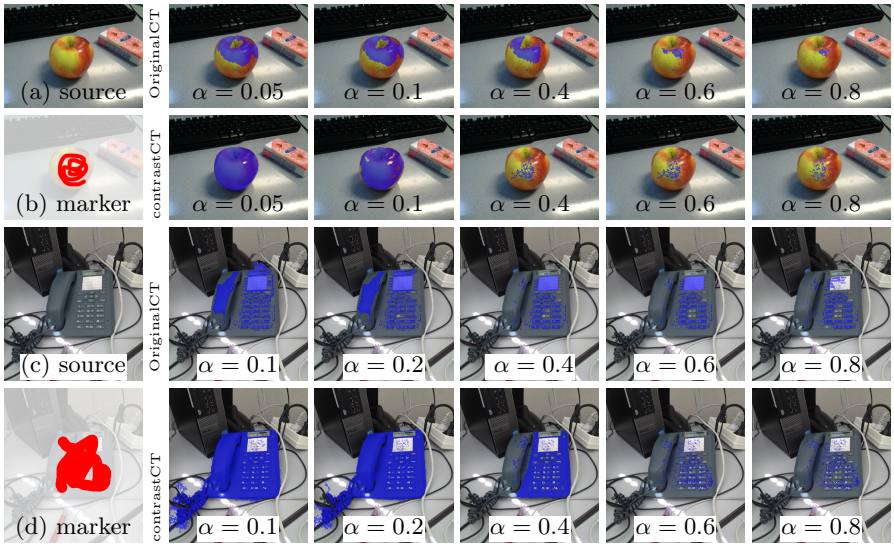


Fig. 6. Comparisons of the original Component Tree Algorithm (originalCT) and the improved algorithm exploiting contrast information (contrastCT) for different values of the parameter α.

The efficiency of the alpha automatic settings was evaluated on four various test images and compared with another segmentation method called *Morphological Snakes* which can also be applied in real time [7] (see Fig. 7). The improved component tree algorithm presents better quality result and has no parameter to tune contrary the morphological snake approach. Note that the source code of an online version working on *Android* and *PC* is available [8].

5 Conclusion

In this paper, we have addressed the problem of interactive segmentation. Several improvements to the component-tree based segmentation method were proposed. First, we apply some transformations to the image according to the marker defined in order to adapt the component-tree to the user need. Second, we introduce an automatic way of setting α parameter based on meaningful scales. The interest of the proposed approach was experimented by some experiments and comparisons.

In future works, we will focus on highly-textured objects, which are not currently segmentable with our approach. Moreover, we plan to study the ex-

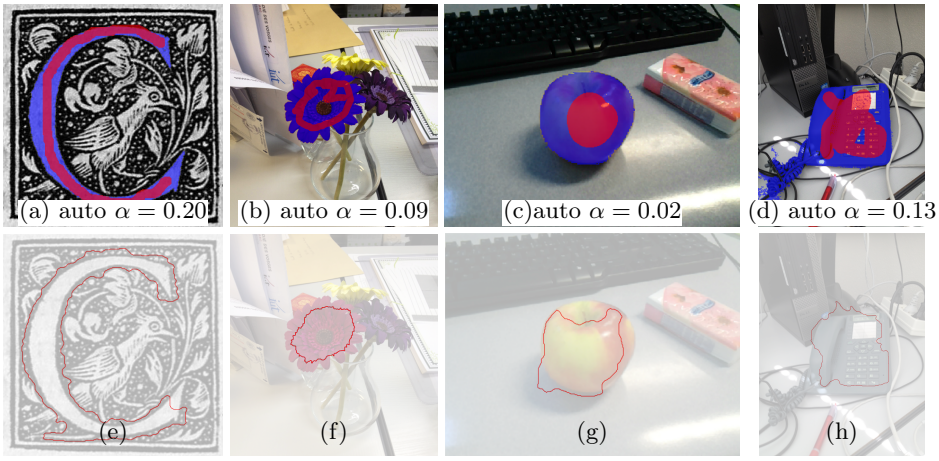


Fig. 7. Experiments of the automatic set of the alpha value (a-d) and comparisons with the *Morphological Snake* algorithm [7] (e-h) with the same markers than experiments (a-d) and with the expansion ballonforce (experimented from the author source code and displayed with the DGtal library [9]).

tension of component-tree to video data, one possible solution can be the use of spatio-temporal quasi-flat zones [10] as tree nodes instead of spatial connected components.

References

1. Passat, N., Naegel, B., Rousseau, F., Koob, M., Dietemann, J.L.: Interactive segmentation based on component-trees. Pattern Recognition **44** (2011) 2539–2554
2. Liu, D., Xiong, Y., Shapiro, L., Pulli, K.: Robust interactive image segmentation with automatic boundary refinement. In: Proc. of ICIP. (2010) 225 –228
3. Kerautret, B., Lachaud, J.O.: Meaningful scales detection along digital contours for unsupervised local noise estimation. IEEE Transactions on Pattern Analysis and Machine Intelligence (2012) In press: 10.1109/TPAMI.2012.38.
4. Naegel, B., Passat, N.: Component-trees and multivalued images: A comparative study. In: ISMM'09. Volume 5720 of LNCS., Springer (2009) 261–271
5. Passat, N., Naegel, B.: An extension of component-trees to partial orders. In: Proc. of ICIP. (2009) 3981–3984
6. Kerautret, B., Lachaud, J.O.: Meaningful scale detection: online demonstration http://kerrecherche.iutsd.uhp-nancy.fr/MeaningfulBoxes (2009)
7. Alvarez, L., Baumela, L., Henríquez, P., Márquez-Neila, P.: Morphological snakes. In: CVPR'10. (2010) 2197–2202
8. Kowalczyk, M. online (2012) https://github.com/michal-kowalczyk/RealTimeCTSegmentation.
9. DGtal team: DGtal: Digital geometry tools and algorithms library (2012) http://liris.cnrs.fr/dgtal.
10. Weber, J., Lefèvre, S., Gançarski, P.: Spatio-temporal quasi-flat zones for morphological video segmentation. In: ISMM'11. Volume 6671 of LNCS., Springer (2011) 178–189