Hierarchical Interactive Image Segmentation Using Irregular Pyramids*

Michael Gerstmayer, Yll Haxhimusa, and Walter G. Kropatsch

Pattern Recognition and Image Processing Group
Institute of Computer Graphics and Algorithms, Vienna University of Technology
{mig,yll,krw}@prip.tuwien.ac.at

Abstract. In this paper we describe modifications of irregular image segmentation pyramids based on user-interaction. We first build a hierarchy of segmentations by the minimum spanning tree based method, then regions from different (granularity) levels are combined to a final (better) segmentation with user-specified operations guiding the segmentation process. Based on these operations the users can produce a final image segmentation that best suits their applications. This work can be used for applications where we need accuracy in image segmentation, in annotating images or creating ground truth among others.

1 Introduction

Image segmentation cannot produce a perfect final segmentation, only by using low-level visual cues. The reason is the *intrinsic ambiguity* in the exact location of region boundaries in digital images. In general, homogeneity of low-level cues will not map to the semantics [12], and the degree of homogeneity of a region is in general quantified by threshold(s) for a given measure [6]. To avoid problems with the automatic segmentation methods one can use human help to guide segmentation methods, producing results acceptable by users/practitioners. Most interactive or semi-automatic segmentation algorithms make use of this external knowledge, some of them e.g. Snakes [11], Live Wire (or Intelligent Scissors) [17] and recent approaches based on the Graph Cuts formalism [1,18,16] are well known. They are often used in e.g. medical image segmentation, image or video object extraction and to refine or improve results from automatic methods.

But the notion of 'interactive' is ambiguous and not very well defined. Some of the methods are initialized (e.g. statistic shape models, rule sets, training sets) others use seed points or strokes for guiding and limiting a segmentation process [9]. Besides initialization, existing methods can also be categorized either as optimizing (manually guided, influenced) or post-processing methods (manually corrected, modified) [10]. The work presented in this paper is located between the last two categories. It uses user-interaction to guide the minimum spanning tree (MST) based pyramid segmentation [8]. The MST-based segmentation method produces a stack of (dual) graphs (a graph pyramid) [8] on each level of the

* This paper has been supported by the ASF under grant FWF-P20134-N13.

X. Jiang, M. Ferrer, and A. Torsello (Eds.): GbRPR 2011, LNCS 6658, pp. 245–254, 2011.

pyramid (Fig. 3). Each level of this hierarchy corresponds to one image segmentation. Regions from different levels of the segmentation pyramid are combined by user interaction resulting in a modified pyramid containing the 'final good' segmentation at top. In the regions where the user did not set any modification operation, the algorithm will be guided automatically by a pairwise comparison of region similarity [5].

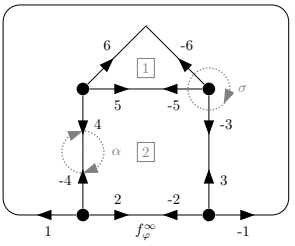
Meine et al. [15] use the topological GeoMap representation and user interaction to guide a segmentation method in the medical image analysis. The use of a topologically correct representation has a major impact on the processing and its results. This motivated us to use combinatorial maps, as a topological representation. The authors in [13] interactively modify a hierarchical watershed segmentation, which is similar to our user modification(s) of the MST based segmentation hierarchy. In our case we can, if needed, access each pixel, which is not the case in the work of [13].

The paper is structured as follows. We first give a short overview of the combinatorial maps and combinatorial pyramids (Section 2). After that the operations that a user can set are presented in Section 3.1, in Section 4 we show some results and conclude the paper.

2 Combinatorial Image Pyramids

In this section a short overview of the most important concepts of combinatorial image pyramids are given. Combinatorial maps and generalized combinatorial maps define a general framework which allows to encode any subdivision on nD topological spaces orientable or non-orientable with or without boundaries [2]. Images and its structure are represented in this work as weighted combinatorial maps. Using $2D$ images, combinatorial maps may be understood as a particular encoding of a planar graph, where each edge is split into two half-edges called darts. Since each edge connects two vertices, each dart belongs to only one vertex. A $2D$ combinatorial map is formally defined by the triplet $G = (\boldsymbol{\mathcal{D}}, \sigma, \alpha)$ [3] where $\boldsymbol{\mathcal{D}}$ represents the set of darts and $\sigma(d)$ is a permutation on $\boldsymbol{\mathcal{D}}$ encountered when turning clockwise around each vertex. Finally $\alpha(d)$ is an involution on $\boldsymbol{\mathcal{D}}$ which maps each of the two darts of one edge to the other one. Given a combinatorial map $G = (\boldsymbol{\mathcal{D}}, \sigma, \alpha)$, its dual is defined by $\overline{G} = (\boldsymbol{\mathcal{D}}, \varphi, \alpha)$, with $\varphi = \sigma \circ \alpha$. The cycles of permutation φ encode the faces/regions of the combinatorial map. In the following cycles of α, $\sigma(d)$ and φ containing a dart d will be respectively denoted by $\alpha^*(d)$, $\sigma^*(d)$ and $\varphi^*(d)$ (an example of a combinatorial map is shown in Fig. 1). Thus all graph definitions used in irregular pyramids are analogously defined. A *combinatorial pyramid* is a stack of combinatorial maps successively reduced by the set of contraction and removal operations, i.e. $(G_0, ..., G_k)$ where k represents the levels of the pyramid. Each map $k+1$ is built from the one below, k, by selecting a set of contraction kernels $K_{k,k+1}$ and applying it to a given combinatorial map G_k to get the reduced map $G_{k+1} = \mathcal{C}[G_k, K_{k,k+1}] = G_k \setminus K_{k,k+1}$. More on removal of the redundant edges can be found in [2].

Region adjacency graphs (RAG), dual graphs [8], combinatorial maps [7], and GeoMap [15] have been used before [2] to represent the partitioning of $2D$



$\mathcal{D}$ $(1,-1,2,-2,3,-3,4,-4,5,-5,6,-6)$:

$\alpha = (1,-1)(2,-2)(3,-3)(4,-4)(5,-5)(6,-6)$
$\sigma = (1,-4,2)(-2,3,-1)(-3,-5,-6)(4,6,5)$
$\varphi = (5,-6)(2,3,-5,4)(-1,-4,6,-3)(-2,1)$

Fig. 1. The house and its region relations are described with permutations on the dartset $\mathcal{D}$ of the combinatorial map. The infinite face f_φ^∞ is encoded with the orbit $\varphi^*(-2) = (-2,1)$.

space. From these structures, we use the combinatorial maps because *RAG*s cannot correctly encode multiple boundaries and inclusions. Dual graphs lack the explicit encoding of edge orientation around vertices, which is present in a combinatorial map [2](e.g. Fig. 1). Moreover with combinatorial maps, its dual must not be explicitly represented. One combinatorial map is enough to fully characterize the partition and to deduce the dual graph.

3 Interactive Operations on Pyramids

Usually, automatic segmentation methods will not be able to deliver a final segmentation that is acceptable by the users (see Fig. 2). Thus there is a need to perform a user interaction such that one can produce a better image segmentation. We have chosen a (hierarchical) pyramid based segmentation method where we define user operations which will guide the merging and division by using regions in different level of the pyramid to a final acceptable image segmentation. The irregular (combinatorial) pyramid [7] produces automatically a stack of segmented images (only some levels are shown in Fig. 2). The segmentation results are produced automatically by merging processes that take low-level cues (in this example RGB color values) into consideration [5].

In this work, the user can set focus on region(s) lying in different levels of the pyramid (having different granularities). One can apply the process of manually changing the image segmentation by merging and/or division operations in any level within the stack of segmentations. Note that in our pyramid all these manual operations defined on the regions will change the merging tree (Fig. 3a). Moreover they also guide the processes in changing it, resulting in a stack of segmented images where the final (wished) segmentation is at top of the pyramid. Because we keep the hierarchy it is always possible to decompose the object into its subparts or restart the process for further refinement. Instead of doing this with unpredictable result (e.g. effect of a stroke in Graph Cuts) we can explicitly address each region in the merging tree (until the pixel level if needed) while non restricting the flexibility of the algorithm in merging (other) non selected (or not in the focus) regions automatically. E.g. Fig. 3b) shows (two) final segmentations (level k and $k+1$). The user has decided first to put focus on parts of the face

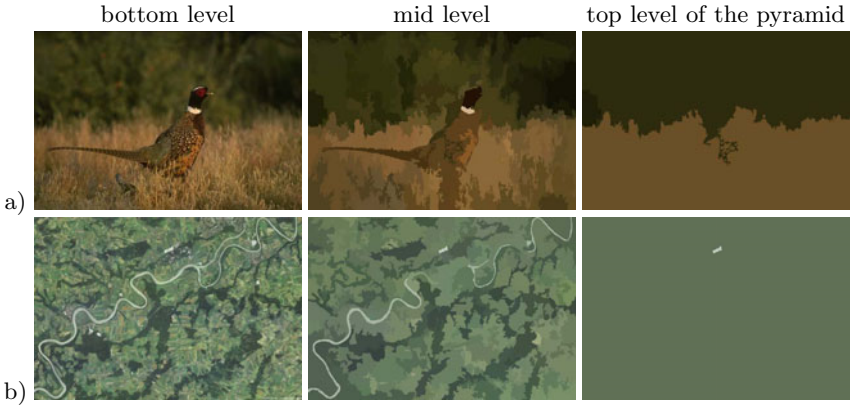


Fig. 2. Image segmentation with automatic segmentation method [8];
a) Due to lack of contrast bird is merged with meadow (Image 43074 from [14]);
b) Due to the thin structure the Danube river disappears.

like eyes, mouth, nose, ears etc. (lv. k), which then got manually merged with the face (lv. $k+1$). The inclusion relations of the eyes, mouth, nose and ears, within the face is correctly encoded in our merging tree. As it is shown, note that in our pyramid we can keep many 'final' segmentations.

3.1 Modifying Operations

The manual image segmentation process consists of two parts: (1) building the irregular pyramid based on the MST [8,7], and (2) a user interface where the user places its modification operations[1]. The framework uses as input the original image, its stack of automatically generated segmented images at different levels (e.g. images in Fig. 2a) and the hierarchical information of the merging tree.

Modifying relations between regions, requires the creation of a correspondence between the user-operations based on regions in the user interface (visual representation) and their combinatorial map correspondent in the same level of the pyramid. This information is given by the structural description of the image relations, inherently encoded in its primal and dual combinatorial maps. To get the inter-pixel boundary [4] in the primal or the edge representing the adjacency in its dual, a calculation is done through permutations on the darts in $\boldsymbol{\mathcal{D}}$ (e.g. boundary between the roof (region 1) and the wall (region 2) in Fig. 1). Each region is represented in the primal by a dart $\in \boldsymbol{\mathcal{D}}$ and its orbit φ^* describing the boundary. These darts (aligned around a vertex corresponding to this region) encode also the relations to the neighboring regions in the dual. Therefore the joining edge, for e.g. merging two regions, is calculated through iterating the φ^* permutations for each region until both darts belong to the same cycle of α and hence to the same edge. This transformation from regions to edges is necessary because the operations placed in the user interface implicitly describe what

[1] Can be found at `http://www.prip.tuwien.ac.at/`

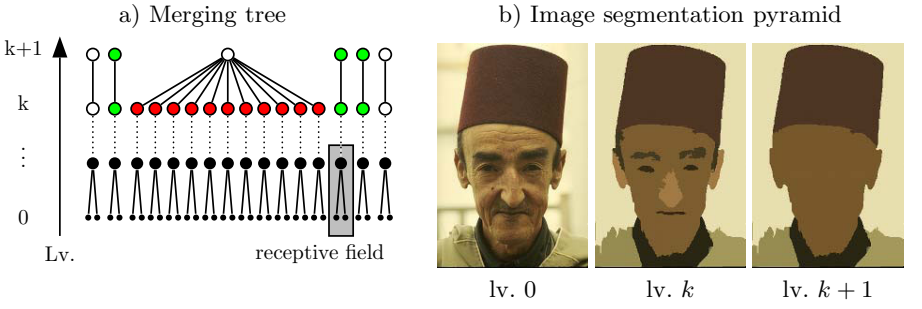


Fig. 3. a) Discrete levels with the merging tree. Interactive operations combine regions from different segmentations in different levels of the pyramid. Green vertices represent regions inhibited from merging in further processing, and red vertices are chosen to get merged. b) User segmentation of Image 189080 from [14].

to do and the representation in terms of combinatorial maps explicitly defines what to be modified. In the presented work we need only the concept of regions' boundary in order to define the user operations. Thus, the idea is general and can be implemented also on other topological representations like region adjacency graphs, dual graphs [8] etc. Therefore this work is not limited to the image representation with combinatorial maps even though it strongly benefits from them.

The operations for the purpose of modifying the relations between the regions are placed in the user interface either by separate selection or brushing over them (e.g. Fig. 4). The two fundamental operations that can be applied on adjacent regions $\mathtt{r}_1$ and $\mathtt{r}_2$ for guiding the segmentation process are:

- $\mathtt{mrg}(\mathtt{r}_1, \mathtt{r}_2)$: merging regions $\mathtt{r}_1$ and $\mathtt{r}_2$,
- $\mathtt{imrg}(\mathtt{r}_1)$: inhibition of merging $\mathtt{r}_1$ with other regions in the levels above

One can define other operations as a combination of these two basic modifying operations. These operations exert influence on the merging tree. But merging solely does not implicate that the resulting region will be inhibited automatically since the algorithm can decide to merge it with other regions in higher levels of the pyramid. Through nesting other combinations are possible, for e.g.:

- $\mathtt{imrg}(\mathtt{mrg}(\mathtt{r}_1, \mathtt{r}_2))$: merge and inhibit the resulting region,
- $\mathtt{imrg}(\mathtt{r}_1, \mathtt{r}_2)$: inhibit both from merging with other regions, etc.

Since we use a hierarchical representation, it is also possible to select regions at different granularity from multiple levels. The basic way of doing this is traversing through all the segmentation levels of the pyramid. The other way is the 'explore' mode, intended to traverse down (toward higher resolution) only within the receptive field of a single region (Fig. 4c). This can be used as a way of applying operations in higher levels on previously merged regions (at lower level of granularity). Its main purpose is dividing them again while e.g. inhibiting the

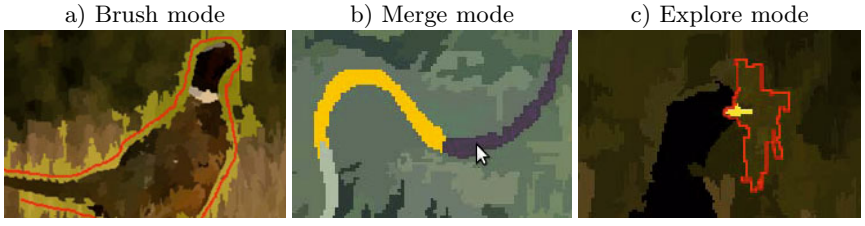


Fig. 4. User interactions. a) Using brush mode (red line) for effectively applying merge operations (encircle bird), b) Merging regions by selection (yellow - first selected region, violet - region to be selected), c) Dividing a region (surrounded red) into its components within its receptive field and restoring a single region from a lower level (indicated with yellow) e.g. the birds beak.

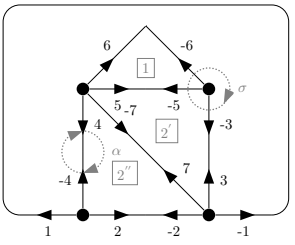
current state. It is also possible that some regions (or their receptive fields) overlap which can lead to conflicts in operations e.g. merging two regions and one of them is inhibited at the same time. We resolve these conflicts by analyzing the combinatorial map and the relations of region(s) under operation i.e. the affected edges (see Section 3.2).

The listing above of operations should be understood as an outline of the versatility of the framework, since many other operations are possible. Finally a set of operations is passed on to the framework, to perform the final segmentation. Each operation entry is of the form $\{\mathtt{lv.}, \mathtt{op.}, \mathtt{r}_1[, \mathtt{r}_2]\}$ where $\mathtt{lv.}$ is the level, $\mathtt{op.}$ the operation, $\mathtt{r}_1$ the first region and $\mathtt{r}_2$ the second one (optional).

3.2 Building Segmentation

In the bottom-up automatic building of the pyramid [8] candidate edges for contraction are chosen to be the smallest weighted edges. The decision whether two regions (vertices connected by edges) are merged is guided by comparison of region similarity [5]. The set of user operations are not immediately applied upwards in the merging tree. Since we allow operations to be selected on different levels of the pyramid we need to down-project these operations on a common starting level. A common starting level is the lowest level in which at least one operation is defined. In this common starting level the candidate-edges affected from operations (contraction or inhibition) are determined, allowing rebuilding the pyramid from this level upwards. More precisely, deleting all older levels above the common starting level and recalculating new levels, i.e. the merging tree is changed only from the common starting level above. That implies the recalculation of the operations to an equivalent instruction set, which is achieved with the hierarchical information, through permutations on $\boldsymbol{\mathcal{D}}$ with φ and set operations as shown in the following example.

Fig. 1 and Fig. 5 are two consecutive levels, $k+1$ and k in the pyramid. Let Fig. 5 be a common starting level. The boundary of the roof of the house (region 1) remains the same in both of the levels. However the wall of the house (region 2) in level k is divided into two subregions $2^{'}$ and $2^{''}$. When applying



$R_{k+1,1} \xrightarrow{div} R_{k,1}$

$R_{k+1,2} \xrightarrow{div} R_{k,2} = R_{k,2'} \wedge R_{k,2''}$

Darts for $R_{k,1} = \varphi(5, -6)$

Darts for $R_{k,2} = \varphi(3, -5, -7) \wedge \varphi(2, 7, 4)$

Fig. 5. The combinatorial map from a segmentation in a lower level (than Fig. 1) of the hierarchy

e.g. the operation $\texttt{imrg}(\texttt{mrg}(1, 2))$ in level $k + 1$, meaning merge region 1 (roof) and 2 (wall) in Fig. 1 and inhibit the resulting region from merging' we can see that this operation leads to a different non corresponding result $\texttt{imrg}(\texttt{mrg}(1, 2^{'}))$ when applied on the combinatorial map in the level below (shown in Fig. 5).

Beginning from the relation between the regions in the common starting level k and the level above $k + 1$, its φ permutation, the corresponding darts (or edges) can be reconstructed as shown in Fig. 5 (right). There are two different categories and collections of edges (inhibition and removal) necessary to process the operation correctly:

1. **Former Contractions:** in each set of $R_{k,1}$ and $R_{k,2}$ some darts belong to the same cycle of α. This edge indicates the adjacency/subdivision from region $R_{k+1,2}$ in level k and is marked for **removal** e.g. $(7, -7)$.
2. **User Operations:**
   - all edges of $R_{k,1}$ and $R_{k,2}$ without the marked edges for removal from former contractions represent the outer borders of regions $R_{k,1}$ and $R_{k,2}$. All these edges are marked for **inhibition**, e.g. darts $(3, -3)(5, -5)$, $(2, -2)(4, -4)$ and $(6, -6), (5, -5)$
   - the darts from the outer borders of $R_{k,1}$ and $R_{k,2}$ of each region are compared to find the edge in common. This edge is marked for **removal** e.g. $(5, -5)$.

Selection of edge $(5, -5)$ would lead to a conflict because it appears in both inhibition and removal edge sets. Because user intention is to merge, a rule decides that this edge has to be removed. Similar to above example we have correctly defined the assignment to the collections of user selected edges for processing (removal or inhibition) for all other user operations. Rebuilding the pyramid from the common starting level, is done as follows: first all edges in this collection of edges are processed first, then all other edges not in this collection are automatically processed by [8]. The processing is based on dual contraction [3] and guarantees a consistent state of the combinatorial map. Note that all edges marked for inhibition will not be deleted by the automated process and

survive until the top of the pyramid. The resulting segmentation pyramid considers the selected regions and operations containing a new final segmentation at top, where a final state is reached (e.g. Fig. 3, lv. k).

This framework can be used easily for creating a nested image annotation tree, e.g. by locking all levels below the top of the pyramid and iteratively merge the remaining regions (e.g. Fig. 3, lv. $k+1$) to build a new hierarchy. In contrast to other tools (e.g. labelme[19]) where each level of abstraction has to be created separately, this is an enormous simplification. Furthermore it could be easily used in creating ground truth in image and video databases.

4 Segmentation Results

We show by means of some images the applicability of the framework. The user has to define what is the region of focus and which objects are of interest. To produce the result in Fig. 6a three pyramid rebuilds are necessary. A detailed set of user operations applied are shown in Table 1. The brush was used to encircle the bird (Fig. 4a) in a level of the pyramid with a fine segmentation. This causes that a limiting boundary is created. In the second run errors in segmentation were removed. Finally in the third run the outside area is inhibited and all inner regions are automatically merged to produce a final segmentation (Fig. 6a).

In an aerial image of the Danube river (Fig. 2b) the user wants to segment the river properly. Some parts of the river were correctly segmented automatically in higher levels, thus these regions are inhibited in the first manual interaction. In the thinner branches correction at pixel level is necessary, thus 36 clicks are needed (Table 1, Fig. 6b). The last image segmented (Fig. 6c) by a user has multiple regions of interest at different granularities (the rock, the person on the rock and the lake/river). Within multiple levels and two rebuilds regions are merged together and inhibited from merging. The processing takes in general per rebuild couple of seconds on images with 500×500 on a PC (2 GHz processor with 1 GB RAM). Since each user segmentation is intended for different applications, a straightforward comparison with other frameworks is difficult.

a) 43074: Focus on the bird b) Focus on the river c) 14036: Focus on details

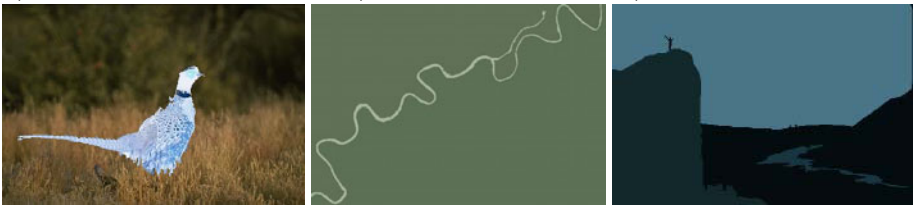


Fig. 6. Final segmentation after user interaction on the segmentation hierarchy

Table 1. Segmentation results in Fig. 6. # Lv.: number of different levels modified, # Op.: sum of operations, Time: *min*:sec, interaction time contains time for placing operations/int. time overall, computation time: time for result.

Image	Run	Input	#	Op.	# Lv.	Clicks	Interaction Time	Comp. Time
43074	1.	brush	104	`imrg(mrg)`	1	1	01:09/01:41	00:32
	2.	select	7	`imrg`	5	7	00:13/03:51	00:55
		select	3	`imrg(mrg)`		6		
	3.	select	1	`imrg(mrg)`	1	2	00:02/00:18	00:10
River	1.	select	14	`imrg(mrg)`	3	28	00:49/06:24	00:43
		select	5	`imrg`	5	5		
	2.	select	3	`imrg`	1	3	00:02/00:33	00:10
14036	1.	select	2	`imrg`	3	2	00:09/01:20	01:35
		select	2	`imrg(mrg)`		4		
	2.	select	3	`imrg`	1	3	00:14/01:00	00:15
		select	2	`imrg(mrg)`		4		

5 Discussion

Depending on the object(s)/region of interest where the user wants to set the focus on, the following ways can lead to the same final segmentation result: (1) select different regions from different levels, (2) encircle object of interest with the brush (or selection), (3) limit/fill-out object of interest using the brush (or selection), (4) start from a coarse segmentation of the object and then split up or (5) start from a fine segmentation of the object and merge. One can choose to combine one of the above to have a hybrid approach. Explicitly defining what to be done might cause that an object not denoted, but correctly segmented before will get lost. As shown in Figure 6, it is possible to force a segmentation, in the worst case by defining operations using segments at pixel level. Undoing operations relates to modify the instruction set, hence it is also possible to start from or correct an existing segmentation produced by other segmentation methods. A solution might be to reconstruct the hierarchical merging tree by using the segments of the output of the segmentation method. Out of the receptive fields of each segment, operations can be created to recalculate the intermediate levels.

The user interface interacts with the segmentation framework but is clearly separated and no knowledge about the underlying data-structure is necessary. We will further evaluate this framework in the terms of usability.

6 Conclusion

The approach of interactively modifying an irregular pyramid by guiding it with user-specified operations is introduced. In contrast to other methods, the presented framework delivers a general purpose but versatile method for creating user-guided segmentation hierarchies. The various strategies of interactions and the solutions developed for effective processing have been analyzed and discussed.

References

1. Arbelaez, P., Maire, M., Fowlkes, C., Malik, J.: Contour detection and hierarchical image segmentation. IEEE Transactions on Pattern Analysis and Machine Intelligence (99) (2010)
2. Brun, L., Kropatsch, W.: Contraction kernels and combinatorial maps. Pattern Recognition Letters 24(8), 1051–1057 (2003)
3. Brun, L., Kropatsch, W.G.: Dual contraction of combinatorial maps. Tech. Rep. PRIP-TR-54, Institute of Computer Graphics and Algorithms 186/3, Pattern Recognition and Image Processing Group, TU Wien, Austria (1999)
4. Brun, L., Vautrot, P., Meyer, F.: Hierarchical watersheds with inter-pixel boundaries. In: Campilho, A.C., Kamel, M.S. (eds.) ICIAR 2004. LNCS, vol. 3211, pp. 840–847. Springer, Heidelberg (2004)
5. Felzenszwalb, P.F., Huttenlocher, D.P.: Efficient graph-based image segmentation. International Journal of Computer Vision 59(2), 167–181 (2004)
6. Fuh, C.S., Cho, S.W., Essig, K.: Hierarchical color image region segmentation for content-based image retrieval system. IEEE Transactions on Image Processing, 9(1), 156–162 (2000)
7. Haxhimusa, Y., Ion, A., Kropatsch, W.G., Brun, L.: Hierarchical image partitioning using combinatorial maps. In: Chetverikov, D., Czuni, L., Vincze, M. (eds.) Proceeding of the Joint Hungarian-Austrian Conference on Image Processing and Pattern Recognition, Hungary, May 2005, pp. 179–186 (2005)
8. Haxhimusa, Y., Kropatsch, W.G.: Segmentation graph hierarchies. In: Fred, A., Caelli, T.M., Duin, R.P.W., Campilho, A.C., de Ridder, D. (eds.) SSPR&SPR 2004. LNCS, vol. 3138, pp. 343–351. Springer, Heidelberg (2004)
9. Heimann, T., et al.: Comparison and evaluation of methods for liver segmentation from ct datasets. IEEE Transactions on Medical Imaging 28(8), 1251–1265 (2009)
10. Hug, J.M.: Semi-Automatic Segmentation of Medical Imagery. Ph.D. thesis, Swiss Federal Institute of Technology Zürich (2000)
11. Kass, M., Witkin, A.P., Terzopoulos, D.: Snakes: Active contour models. International Journal of Computer Vision 1(4), 321–331 (1988)
12. Keselman, Y., Dickinson, S.: Generic model abstraction from examples. IEEE Transactions on PAMI 27(5), 1141–1156 (2005)
13. Klava, B., Sumiko, N., Hirata, T.: Interactive image segmentation with integrated use of the markers and the hierarchical watershed approaches. In: VISSAPP (1), pp. 186–193 (2009)
14. Martin, D., Fowlkes, C., Tal, D., Malik, J.: A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In: Proc. 8th Int'l Conf. Computer Vision, vol. 2, pp. 416–423 (2001)
15. Meine, H., Köthe, U., Stiehl, H.: Fast and accurate interactive image segmentation in the geomap framework. In: Tolxdorff, T. (ed.) Bildverarbeitung für die Medizin 2004, pp. 60–65. Springer, Heidelberg (2004)
16. Micusík, B., Hanbury, A.: Automatic image segmentation by positioning a seed. In: ECCV (2), pp. 468–480 (2006)
17. Mortensen, E.N., Barrett, W.A.: Interactive segmentation with intelligent scissors. Graphical Models and Image Processing 60(5), 349–384 (1998)
18. Rother, C., Kolmogorov, V., Blake, A.: "grabcut": interactive foreground extraction using iterated graph cuts. ACM Trans. Graph. 23(3), 309–314 (2004)
19. Torralba, A., Russell, B., Yuen, J.: Labelme: Online image annotation and applications. Proceedings of the IEEE 98(8), 1467–1484 (2010)