Annotated Contraction Kernels for Interactive Image Segmentation

Hans Meine

Cognitive Systems Laboratory, University of Hamburg,
Vogt-Kölln-Str. 30, 22527 Hamburg, Germany
meine@informatik.uni-hamburg.de

Abstract. This article shows how the interactive segmentation tool termed "Active Paintbrush" and a fully automatic region merging can both be based on the theoretical framework of contraction kernels within irregular pyramids instead of their own, specialized data structures. We introduce "continous pyramids" in which we purposely drop the common requirement of a fixed reduction factor between successive levels, and we show how contraction kernels can be annotated for a fast navigation of such pyramids. Finally, we use these concepts for improving the integration of the automatic region merging and the interactive tool.

1 Introduction

One of the most valueable and most often employed tools for image segmentation is the watershed transform, which is based on a solid theory and extracts object contours even with low contrast. On the other hand, it is often criticized for delivering a strong oversegmentation, which is simply a consequence of the fact that the watershed transform has no built-in relevance filtering. Instead, it is often used as the basis for a hierarchical segmentation setting in which an initial oversegmentation is successively reduced, i.e. by merging adjacent regions that are rated similar by some appropriate cost measure (e.g. the difference of their average intensity) [1,2,3,4]. This bottom-up approach fits very well with the concept of irregular pyramids [5,6], and the main direction of this work is to show how the Active Paintbrush – an interactive segmentation tool developed for medical imaging [2] – and an automatic region merging [7,3,2] can be formulated based on the concepts of irregular pyramids and contraction kernels. This serves three goals: a) delivering a useful, practical application of contraction kernels, b) basing the description of segmentation methods on well-known concepts instead of their own, specialized representation, and c) demonstrating how a common representation facilitates the development of a more efficient integration of the above automatic and interactive methods.

The following sections are organized as follows: In section 2, we summarize previous work on the Active Paintbrush and automatic region merging (2.1) and on irregular pyramids and contraction kernels (2.2). Section 3 combines these concepts and introduces the ideas of continuous pyramids and annotated contraction kernels (3.1), before proposing methods that exploit this new foundation for a better integration of automatic and interactive tools (section 3.2).

2 Previous Work

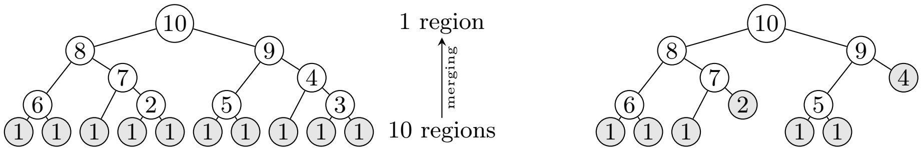
2.1 The Active Paintbrush Tool

The Active Paintbrush was introduced by Maes [2] as an efficient interactive segmentation tool for medical imaging. It is based on an initial oversegmentation produced using the watershed transform, and a subsequent merging of regions. The latter is performed in two steps:

1. First, an automatic region merging reduces the oversegmentation by merging adjacent regions based on some homogeneity measure (in [2], an MDL criterion is used, but there is a large choice of suitable measures [3,8]).
2. Subsequently, the Active Paintbrush allows the user to "paint" over region boundaries to quickly determine the set of regions belonging to the object to be delineated.

Since this is a pure bottom-up approach (i.e. the number of regions monotonically decreases, and no new boundaries are introduced), this approach relies on all important boundaries being already present in the initial oversegmentation. The user steers the amount of merging performed in the first step in order to remove as many boundaries as possible (to reduce the time spent in the second step) without losing relevant parts.

Merge Tree Representation For this work, it is important to highlight the internal representation built within the first step, in which the automatic region merging interactively merges the two regions rated most similar (an equivalent approach is used in [7,3,2,8]). This process is continued until the whole image is represented by one big region, and at the same time a hierarchical description of the image is built up: a tree of merged regions, the leaves of which are the primitive regions of the initial oversegmentation (illustrated in Fig. 1a). This tree can also be interpreted as encoding a stack of partitionings, each of which contains one region less than the one below.



(a) full merge tree (10 regions)

(b) pruned tree (7 regions)

Fig. 1: Hierarchical description of image as tree of merged primitive regions [2]

By labeling each merged node with the step in which the merge happened, it becomes very easy to prune the tree as the user adjusts the amount of merging interactively: for instance, the partitioning at level $l = 4$ within the above-mentioned stack can be retrieved by pruning all branches below nodes with a label$\leq l$ (cf. Fig. 1b).

Limitations While this approach already allows for a relatively efficient interactive segmentation, there is one limitation that we will remove in this article which increases the efficiency a lot: the two above-mentioned steps are strictly separated. This is unfortunate, since the automatic method used in the first step in general produces partitionings that suffer from oversegmentation in some parts, but already removed crucial edges elsewhere, e.g. at locations with very low contrast. Thus, the merge parameter has to be set low enough not to lose the part with the lowest contrast, and the interactive paintbrush needs to be used to remove all unwanted edges in all other areas, too. It would be helpful if it was possible to just make the needed manual changes and then go back to the automatic method to quickly finish the segmentation.

2.2 Contraction Kernels

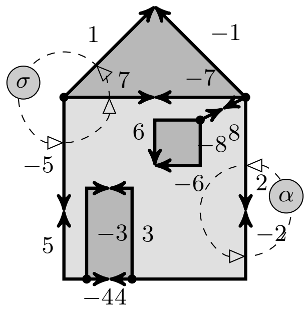
The concept of contraction kernels has been introduced in the context of irregular pyramids [9,10]. Like regular (Burt-style) pyramids, irregular pyramids define tapering stacks of images represented at increasingly coarser scales. However, irregular pyramids are based on graph-like s [5,6] to overcome the drawbacks of regular pyramids imposed by their rigid, regular structure. More recently, combinatorial maps have been widely adopted as the basis for representing irregular tessellations, hence irregular pyramids have been defined as stacks of such maps [11,4,8].

Contraction kernels are used to encode a reduction of one such graph-like structure into a simpler one, i.e. the difference between two levels in an irregular pyramid. In order to give a formal definition, we first need to recall the definitions of some underlying concepts, starting with combinatorial maps (see Fig. 2):

Definition 1 (combinatorial map). *A* combinatorial map *is a triple* $(\mathcal{D}, \sigma, \alpha)$ *where* $\mathcal{D}$ *is a set of* darts *(half-edges), and* σ, α *are permutations defined on* $\mathcal{D}$ *such that* α *is an involution (all orbits have length* 2*) and the map is* connected*, i.e. there exists a* σ*-*α*-path between any two darts.*

In order to represent a segmented image, each edge of the boundary graph is split into two opposite darts, and the permutation α is used to tie these pairs of darts together, i.e. each α-orbits represents an edge of the boundary graph. The permutation σ then encodes the counter-clockwise order of darts around a vertex, i.e. each σ-orbit corresponds to a vertex of the boundary graph. By convention, $\mathcal{D} \subset \mathbb{Z} \setminus \{0\}$ such that α can be efficiently encoded as $\alpha(d) := -d$. The dual permutation of σ is defined as $\varphi = \sigma \circ \alpha$ and thus encodes the order of darts encountered during a contour traversal of the face to the right, i.e. each φ-orbit represents a face of the tessellation.

In contrast to earlier representations using simple [5,6] or dual graphs [12], combinatorial maps explicitly encode the cyclic order of darts around a face,



$\mathcal{D} = \{1, -1, 2, -2, \ldots, 8, -8\}$

	1	−1	2	−2	3	−3	4	−4	...
α	−1	1	−2	2	−3	3	−4	4	...
σ	−5	−7	−1	3	4	5	−2	−3	...
φ	2	7	4	8	−4	−2	5	3	...

	...	5	−5	6	−6	7	−7	8	−8
α	...	−5	5	−6	6	−7	7	−8	8
σ	...	−4	7	−6	−8	1	8	2	6
φ	...	1	−3	6	−8	−1	−5	−6	−7

$\varphi := \sigma^{-1} \circ \alpha$ for contour traversal

Fig. 2: Example combinatorial map representing the contours of a house

which makes the computation of the dual graph so efficient that it does not need to be represented explicitly anymore.

Nevertheless, combinatorial maps also suffer from some limitations, most notably that they rely on “pseudo edges” or “fictive edges” [12,13] to connect otherwise separate boundary components. Topologically-wise, they are commonly called “bridges”, since every path between their end nodes must pass via this edge. These artificial connections have several drawbacks:

- In some situations, one may want to have bridges represent existing image features, for instance incomplete boundary information or skeleton parts. This would require algorithms to differentiate between fictive and real bridges.
- If we relate topological edges with their geometrical counterparts, we are faced with the problem that fictive edges do not correspond to any geometrical entity. Even topologically-wise, fictive edges “appear arbitrarily placed” [13].
- They lead to inefficient algorithms; e.g. contour traversals are needed to determine the number of holes or to find an enclosing parent region.

Because of the above limitations, combinatorial maps are often used in conjunction with an inclusion relation that replaces the fictive edges [14,15].

Using these topological formalisms, segmentation algorithms can rely on a sound topology that allows them to work with regions and boundaries as duals of each other. However, segmentation first and foremost relies on an encoding of the tessellation’s geometry, which is not represented by the above maps. Thus, they are typically used side-by-side with a label image or similar.

Therefore, we have introduced the GeoMap [16,17,8] which represents both topological and geometrical aspects of a segmentation, thus allowing algorithms no longer to deal with pixels directly, and ensuring consistency between geometry and topology. In particular, this makes algorithms independent of the embedding model and allows to use either inter-pixel boundaries [18], 8-connected pixel boundaries [16], or sub-pixel precise polygonal boundaries [17,8].

Reduction Operations In order to build irregular pyramids using any of the above maps, one needs some kind of reduction operation for building higher levels from the ones below, analogous to the operations used for regular pyramids. While in Gaussian pyramids, the reduction operation is parametrized by a Gaussian (smoothing) kernel, Kropatsch [9] has introduced contraction kernels for irregular pyramids (for brevity, we give the graph-based definition here, which is less involved than the analoguous definition on combinatorial maps [10]):

Definition 2 (contraction kernel). *Given a graph $G(V, E)$, a* contraction kernel *is a pair (S, N) of a set of* surviving vertices $S \subset V$ *and a set of* non-surviving *edges $N \subset E$ such that (V, N) is a spanning forest of G and S are the roots of the forest (V, N).*

A contraction kernel is applied to a graph whose vertices represent regions (cf. the dual map $(\mathcal{D}, \varphi)$) by contracting all edges in N, such that for each graph in the forest, all vertices connected by the graph are identified and represented by its root $s \in S$ (details on contractions within combinatorial maps may be found in [11]). In simple words, a contraction kernel is used to specify groups of adjacent regions within a segmentation that should be merged together.

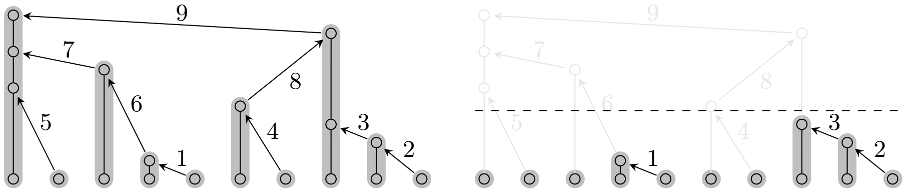
3 Contraction Kernels for Efficient Interactive Segmentation

3.1 Interactive Navigation of Continuous Pyramids

Contraction kernels as described in section 2.2 form a very general description of a graph decimation, i.e. much more general than previous approaches [5,6], which had strict requirements on the chosen survivors and contracted edges. For example, although it may be desirable for some approaches to have a logarithmic tapering graph pyramid for computational reasons [19], the above definition does not enforce this at all.

Continuous Pyramids In fact, we can build "continuous pyramids" in which only one region is merged in every step, as done by the stepwise optimization used for the Active Paintbrush preprocessing [7,2]. In our context, the reduction factor between successive levels can be declared irrelevant:

- In practice, it is unneeded to represent all levels at the same time; instead, we will show in the following how to efficiently encode only the bottom layer and an annotated contraction kernel that allows to directly recreate any level of the whole hierarchy from it. Thus, memory is no issue.
- The whole purpose of introducing irregular pyramids is to preserve fine details at higher levels, which should let further analysis steps work on single levels instead of the whole hierarchy at once.
- Given the right merge order, traditional irregular pyramids simply consist of a subset of the levels of our continuous pyramid, and even for good cost measures, it is unlikely that the implicit selection of the levels is optimal. Therefore, we propose to separate the computation of the pyramid and the subsequent level selection, and leave the latter up to the analysis algorithm.



(a) annotated contraction kernel (b) contraction kernel for the fourth level

Fig. 3: Annotated contraction kernels for a continuous pyramid (cf. Fig. 1)

Annotated Contraction Kernels We have already hinted at how our representation of this continuous pyramid looks like: We simply represent the pyramid's bottom by means of one GeoMap and the series of merges by an *annotated contraction kernel* that resembles the merge tree from section 2.1. Then, when retrieving a given pyramid level l, we take advantage of the concept of *equivalent contraction kernels* [9,11], which means that it is possible to combine the effect of a sequence of contraction kernels (here, merging only two regions each) into a single, equivalent kernel.

The contraction kernel illustrated in Fig. 3a reduces the bottom layer to a single surviving region (represented by the leftmost vertex), i.e. it contains a single, spanning tree. The key to its use is the annotation: while the automatic algorithm used in the preparation step of the Active Paintbrush merged all regions in order of increasing cost (i.e. increasing dissimilarity), we composed the corresponding contraction kernels, effectively building the depicted tree, and labeled each edge with the step in which the corresponding merge happened (analoguous to the node labels used in [2]).

Now when a given level l shall be retrieved (e.g. the user interactively changes the desired granularity of the segmentation), we do not have to explicitly perform the sequence of region merges that led from the initial oversegmentation to l, but we can apply the combined, equivalent contraction kernel at once, which can be implemented much more efficently (e.g. partially parallelized). The annotation allows us to derive this contraction kernel simply by removing all edges with labels $\geq l$. This is illustrated by the dashed cut in Fig. 3b, which shows the contraction kernel leading to the same segmentation as in the example from Fig. 1b. The same approach can be used to jump from any level l_1 to a level $l_2 \geq l_1$, where edges with labels $< l_1$ can be ignored (the reader may imagine a second cut from below).

Often, we are also interested in the values of the merge cost (i.e. dissimilarity) measure associated with each step; therefore, we do not only label each edge in our annotated contraction kernel with the step, but with a (step, cost) pair. This makes an efficient user interface possible that allows an operator to quickly choose any desired level of segmentation granularity. Some example levels generated from a CT image of human lungs using the region-intensity- and -size-based "face homogeneity" cost measure c_{fh} from [3] are depicted in Fig. 4; from left

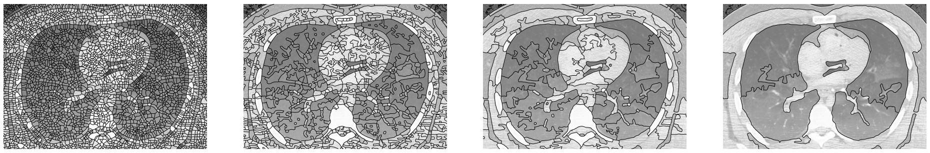
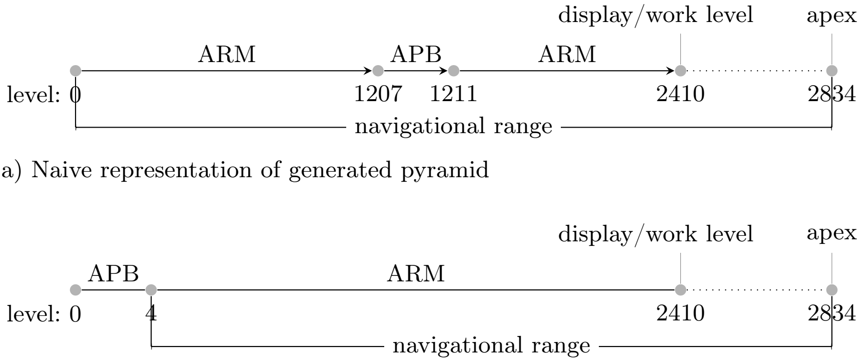


Fig. 4: Example pyramid levels generated by the automatic region merging

to right: level 0 with 9020 regions, level 7020 with 2000 regions ($c_{\mathrm{fh}} \approx 0.12$), level 8646 with 374 regions ($c_{\mathrm{fh}} = 0.5$), and level 9000 with 20 regions left ($c_{\mathrm{fh}} \approx 5.07$).

3.2 Efficient Integration of Manual and Automatic Segmentation

As described in section 2.1, the use of the Active Paintbrush [2] consists of two steps: after the oversegmentation and the hierarchical representation have been computed, the user first adjusts the level of automatic merging by choosing an appropriate level from the imaginary stack of tesselations. Afterwards, the operator uses the Active Paintbrush to "paint over" any remaining undesirable boundaries within the object of interest, which effectively creates new pyramid levels.



(a) Naive representation of generated pyramid

(b) Pyramid after reordering to protect manual changes from disappearing

Fig. 5: Alternating application of automatic and interactive methods

We can now implement the automatic and the interactive reduction methods based on the same internal, map-based representation and contraction kernels. This opens up new possibilities with respect to the combination of the tools, i.e. we can now use one after the other for reducing the oversegmentation and creating further pyramid levels up to the desired result. This is illustrated in Fig. 5a: the levels of our continuous pyramid are ordered from level 0 (initial oversegmentation) on the left to level 2834 (the apex, at which the whole image is represented as one single region) on the right. The current pyramid is the result of applying first the automatic region merging (ARM), then performing some manual actions with the Active Paintbrush (APB), then using the ARM again.

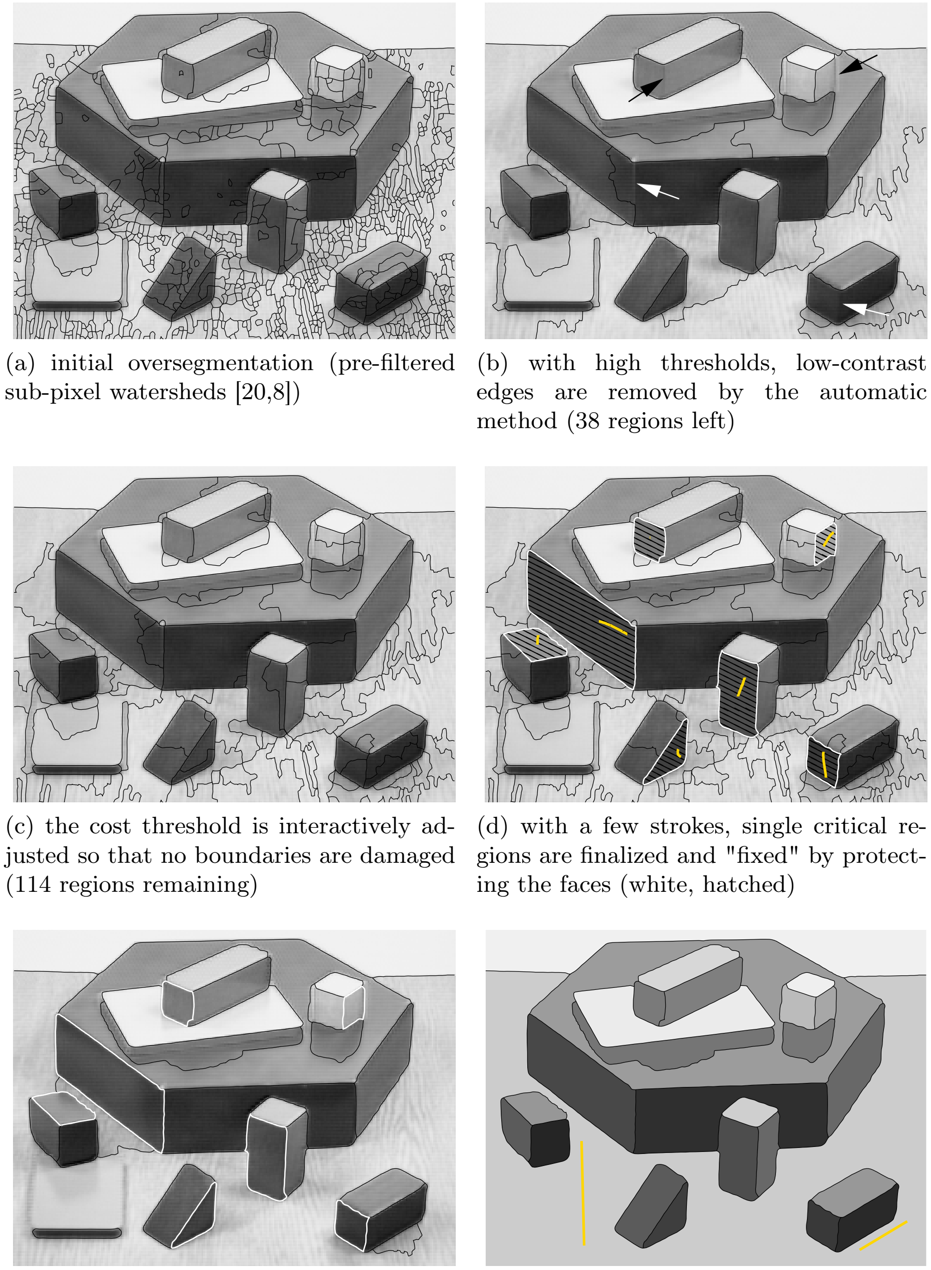
However, this architecture poses difficulties when the user is given the freedom to e.g. change the cost measure employed by the ARM or to navigate to lower pyramid levels than those generated manually: it is very unintuitive if the results of one's manual actions disappear from the working level, or if the pyramid is even recomputed such that they are lost completely. Again, the solution lies in the concept of equivalent contraction kernels, which make it possible to reorder merges: we represent the results of applying the Active Paintbrush in separate contraction kernels such that they always get applied first, see Fig. 5b. (This is equivalent to labeling the edges within our annotated contraction kernel with zero.) In effect, this makes it possible to locally finish the segmentation of an object at the desired pyramid level, but to go back to lower pyramid levels when one notices that important edges are missing in other parts of the image.

We also add the concept of *face protection* to improve the workflow in the opposite direction: often, the Active Paintbrush is used to remove all unwanted edges within the contours of an object of interest. Then, it should be possible to navigate to higher pyramid levels without losing it again, so we provide a means to protect a face, effectively finalizing all of its contours. An example segmentation session using these tools is illustrated in Fig. 6.

4 Conclusions

In this paper, we have shown how the theory of contraction kernels within irregular pyramids can be used as a solid foundation for the formulation of interactive segmentation methods. We have introduced annotated contraction kernels in order to be able to quickly retrieve a contraction kernel suitable for efficiently computing any desired level directly from the pyramid's bottom or from any of the levels in between. Furthermore, we have argued that logarithmic tapering with a fixed reduction factor is irrelevant for irregular pyramids in contexts like ours, and we have introduced the term continuous pyramids for the degenerate case in which each level has only one region less than the one below.

On the other hand, we proposed two extensions around the Active Paintbrush tool which make it even more effective. First, we have expressed both the automatic region merging and the interactive method as reduction operations within a common irregular pyramid representation. This allowed us to apply the theory of equivalent contraction kernels in order to separate the representation of manual actions from automatically generated pyramid levels and thus to enable the user to go back and forth between segmentation tools. Along these lines, we have also introduced the concept of face protection which complements the Active Paintbrush very well in a pyramidal context.



(a) initial oversegmentation (pre-filtered sub-pixel watersheds [20,8])

(b) with high thresholds, low-contrast edges are removed by the automatic method (38 regions left)

(c) the cost threshold is interactively adjusted so that no boundaries are damaged (114 regions remaining)

(d) with a few strokes, single critical regions are finalized and "fixed" by protecting the faces (white, hatched)

(e) now, automatic region merging can be applied again, without putting the protected regions at risk (30 regions left)

(f) with two quick final strokes, three remaining unwanted regions are removed to get this final result (27 regions)

Fig. 6: Example session demonstrating our new face protection concept; the captions explain the user actions for going from (a) to (f)

References

1. Najman, L., Schmitt, M.: Geodesic saliency of watershed contours and hierarchical segmentation. IEEE T-PAMI **18** (1996) 1163–1173
2. Maes, F.: Segmentation and Registration of Multimodal Images: From Theory, Implementation and Validation to a Useful Tool in Clinical Practice. PhD thesis, Katholieke Universiteit Leuven, Leuven, Belgium (1998)
3. Haris, K., Efstratiadis, S.N., Maglaveras, N., Katsaggelos, A.K.: Hybrid image segmentation using watersheds and fast region merging. IEEE Trans. on Image Processing **7** (1998) 1684–1699
4. Meine, H.: XPMap-based irregular pyramids for image segmentation. Diploma thesis, Dept. of Informatics, Univ. of Hamburg (2003)
5. Meer, P.: Stochastic image pyramids. Comput. Vision Graph. Image Process. **45** (1989) 269–294
6. Jolion, J.M., Montanvert, A.: The adaptive pyramid: A framework for 2D image analysis. CVGIP: Image Understanding **55** (1992) 339–348
7. Beaulieu, J.M., Goldberg, M.: Hierarchy in picture segmentation: A stepwise optimization approach. IEEE T-PAMI **11** (1989) 150–163
8. Meine, H.: The GeoMap Representation: On Topologically Correct Sub-pixel Image Analysis. PhD thesis, Dept. of Informatics, Univ. of Hamburg (2009) in press.
9. Kropatsch, W.G.: From equivalent weighting functions to equivalent contraction kernels. In: Digital Image Processing and Computer Graphics: Applications in Humanities and Natural Sciences. Volume 3346., SPIE (1998) 310–320
10. Brun, L., Kropatsch, W.G.: Contraction kernels and combinatorial maps. Pattern Recognition Letters **24** (2003) 1051–1057
11. Brun, L., Kropatsch, W.G.: Introduction to combinatorial pyramids. In: Digital and Image Geometry. Volume 2243 of LNCS. Springer (2001) 108–127
12. Kropatsch, W.G.: Building irregulars pyramids by dual graph contraction. IEEE-Proc. Vision, Image and Signal Processing **142** (1995) 366–374
13. Kropatsch, W.G., Haxhimusa, Y., Lienhardt, P.: Hierarchies relating topology and geometry. In: Cognitive Vision Systems. Springer (2004)
14. Brun, L., Domenger, J.P.: A new split and merge algorithm with topological maps and inter-pixel boundaries. In: The 5^{th} Intl. Conference in Central Europe on Computer Graphics and Visualization (WSCG'97). (1997)
15. Köthe, U.: XPMaps and topological segmentation - a unified approach to finite topologies in the plane. In Braquelaire, A.J.P., Lachaud, J.O., Vialard, A., eds.: Proc. DGCI '02. Volume 2301 of LNCS., Springer (2002) 22–33
16. Meine, H., Köthe, U.: The GeoMap: A unified representation for topology and geometry. In Brun, L., Vento, M., eds.: Proc. W. Graph-Based Representations in Pat. Rec. '05. Volume 3434 of LNCS., Springer (2005) 132–141
17. Meine, H., Köthe, U.: A new sub-pixel map for image analysis. In Reulke, R., Eckardt, U., Flach, B., Knauer, U., Polthier, K., eds.: Proc. 11^{th} Intl. Workshop on Combinatorial Image Analysis. Volume 4040 of LNCS., Springer (2006) 116–130
18. Braquelaire, J.P., Brun, L.: Image segmentation with topological maps and inter-pixel representation. J. Vis. Comm. and Image Representation **9** (1998) 62–79
19. Haxhimusa, Y., Glantz, R., Saib, M., Langs, G., Kropatsch, W.G.: Logarithmic tapering graph pyramid. In: Proc. DAGM '02. Volume 2449 of LNCS., Springer (2002) 117–124
20. Meine, H., Köthe, U.: Image segmentation with the exact watershed transform. In: Proc. Intl. Conf. Visualization, Imaging, and Image Processing. (2005) 400–405