Texture Segmentation by Contractive Decomposition and Planar Grouping

Dahl, Anders Bjorholm; Bogunovich, Peter; Shokoufandeh, Ali

Published in:
Lecture Notes in Computer Science

Link to article, DOI:
10.1007/978-3-642-02124-4_35

Publication date:
2009

Texture Segmentation by Contractive Decomposition and Planar Grouping

Anders Bjorholm Dahl[1], Peter Bogunovich[2], and Ali Shokoufandeh[2]

[1] Technical University of Denmark, Department of Informatics
Lyngby, Denmark
abd@imm.dtu.dk
[2] Drexel University, Department of Computer Science
Philadelphia, PA, USA
{pjb38,ashokouf}@drexel.edu

Abstract. Image segmentation has long been an important problem in the computer vision community. In our recent work we have addressed the problem of texture segmentation, where we combined top-down and bottom-up views of the image into a unified procedure. In this paper we extend our work by proposing a modified procedure which makes use of graphs of image regions. In the top-down procedure a quadtree of image region descriptors is obtained in which a novel affine contractive transformation based on neighboring regions is used to update descriptors and determine stable segments. In the bottom-up procedure we form a planar graph on the resulting stable segments, where edges are present between vertices representing neighboring image regions. We then use a vertex merging technique to obtain the final segmentation. We verify the effectiveness of this procedure by demonstrating results which compare well to other recent techniques.

1 Introduction

The problem of image segmentation, with the general goal of partitioning an image into non-overlapping regions such that points within a class are similar while points between classes are dissimilar [1], has long been studied in computer vision. It plays a major role in high level tasks like object recognition [2, 3], where it is used to find image parts corresponding to scene objects, and image retrieval [4], where the objective is to relate images from similar segments. Textured objects, in particular, pose a great challenge for segmentation since patterns and boundaries can be difficult to identify in the presence of changing scale and lighting conditions [5]. Often textures are characterized by repetitive patterns [6], and these are only characteristic from a certain scale. Below this scale these patterns will only be partly visible [7] which makes precise boundary detection in this case an additional challenge. The intensity variation of textures is often overlapping with the background, which may add further difficulty. Examples of proposed approaches to texture segmentation include active contours [8], templates [2], or region descriptors [9]. We recently introduced a new approach to texture segmentation [10], where the procedure is unsupervised in the sense that we assume no prior knowledge of the target classes, i.e. number of regions or known textures.

(a) (b) (c)

Fig. 1. Texture segmentation from contractive maps. In (a) a heterogeneous image is shown created by composing a Brodatz texture [11] with itself rotated 90^o in a masked out area obtained from the bird in (b). The resulting segmentation is shown in (c).

Our segmentation technique begins with a top-down quadtree decomposition procedure where nodes describe image regions such that the root describes the entire image; the next four children each describe quarter and so on. Each quadtree node contains a descriptor characterizing the texture of the associated region. This characterization is obtained as a distribution of a set of kernels that we introduced in [10]. At each level of the tree a novel contractive transformation is computed for each node and is applied to update the node. The decomposition is controlled by the stability of the resulting node descriptors relative to their neighbors, and a leaf is obtained either when a node is deemed stable or it covers a subpixel image region. Following this procedure we apply our graph-based merging technique. A planar graph is formed on the resulting leaves with edges connecting neighboring image regions whose weights are based on descriptor similarity. The final segmentation is obtained by iteratively merging nodes with highest similarity.

Figure 1 shows a result of our procedure. Figure 1(a) shows a heterogeneous image with itself rotated 90^o in a masked out area obtained from the bird in Figure 1(b). The resulting segmentation is shown in Figure 1(c). An overview of our procedure is shown in Figure 2. The remainder of the paper is summarized as follows: In section 2 we explain the entire procedure by first reviewing the kPIFS used to obtain a base description of the image, followed by a description of the top-down process where we introduce our novel contraction maps, and finally we describe the bottom-up process which includes the details of the planar graph merging technique. In section 3 we present some results and compare them to other methods. We provide a conclusion in section 4.

2 Method

In this section we present an overview of the general procedure for unsupervised texture segmentation. First we give a brief review of the process of obtaining base characterizations of small regions of the image which serve as a starting point for the segmentation. We then indicate our modifications to the decomposition transformation and the approach to merging leaves and generating the final segmentation.

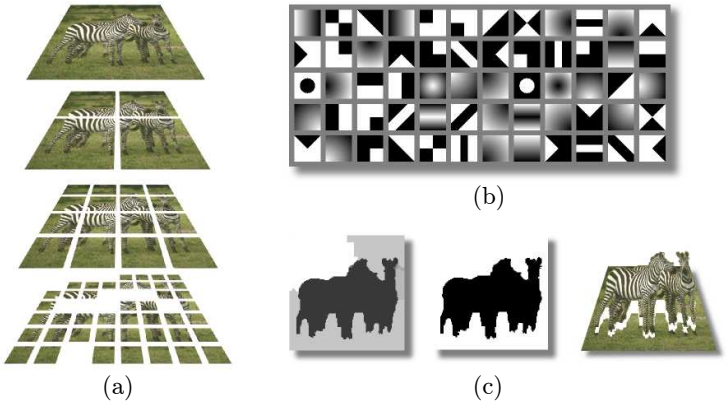

Fig. 2. The segmentation procedure. The top-down decomposition of the image is shown in (a). In (b) the feature kernel set is shown. The first image in (c) is the over-segmented image obtained from the decomposition. The segments are merged in the bottom-up procedure to obtain the final segment shown in the last two images.

2.1 *k*PIFS and the base descriptors

In [10] we introduced the concept of kernel partition iterated function systems (kPIFS) which proved to be a viable technique for obtaining a basic characterization of local image structure to serve as a starting point for segmentation. Since we are primarily focused on the top-down and bottom-up procedures in this paper we only provide a brief review of kPIFS descriptors and we refer the reader to our previous paper [10] for more details.

The kPIFS technique which we developed is inspired by and closely related to the partition iterated function systems (PIFS) introduced by Jacquin [12] for the purpose of lossy image compression [13]. We saw potential in PIFS to characterize local image structure based on evidence indicating that it can be used in tasks such as edge detection [14] and image retrieval [15].

The traditional PIFS image compression technique computes a set of self-mappings on the image. The process begins by partitioning an image into a set of domain blocks D_I, and again into smaller range blocks R_I, as illustrated by Figure 3(b). The image is encoded by matching an element $d_\ell \in D_I$ to each $r_k \in R_I$. In the course of matching, a transformation θ_k which is generally affine is calculated for the domain block d_ℓ that matches range block r_k and $\theta_k(d_\ell)$ is used to represent r_k. Once all of the maps are computed they can be applied to an arbitrary image and will result in an accurate reconstruction of the encoded image.

For our goal of characterizing local structure we designed kPIFS to avoid self-mappings between domain blocks and range blocks. Instead we chose to find mappings from an over-complete basis of texture kernels, D_K, to the range blocks of the image as illustrated by Figure 3(c). The kernels employed here are meant to represent local structural image patterns such as corners, edges of varying width and angle, blobs, and flat

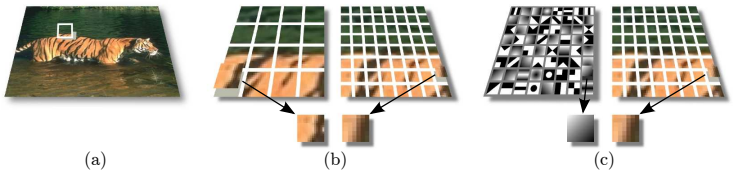

Fig. 3. Comparison of PIFS and kPIFS. Part (a) shows the original image with the highlighted area is focused on in (b) and (c). Part (b) is an example of PIFS where the best matching domain block is mapped to a range block. Part (c) shows the kPIFS where the domain blocks are replaced by domain kernels.

regions. In our procedure, each image range block will be characterized by distances of each of the domain kernels to the range block after a calibration transform is applied. Specifically, for a domain kernel $d_\ell \in D_K$ and a range block $r_k \in R_I$ the distance in kPIFS is given by

$$\delta_{k\mathrm{PIFS}}(r_k, d_\ell) = \left|\left| \frac{d_\ell - \mu_{d_\ell}}{\sigma_{d_\ell}} - \frac{r_k - \mu_{r_k}}{\sigma_{r_k}} \right|\right|, \tag{1}$$

where μ_x and σ_x are the mean and standard deviation respectively of block x. The calibrated blocks will be highly influenced by noise if σ_{r_k} is small and if it is zero we cannot estimate $\delta_{k\mathrm{PIFS}}$. Therefore, we use a measure of flatness of the range blocks, $b_f = \sigma_{r_k}/\sqrt{\mu_{r_k}}$. If $b_f < t_f$, where t_f is a threshold, we categorize the block as flat.

We then let each range block be described by its best mapped (least distant) domain kernels. The similarity for a kernel is weighted by the relative similarity of all of the kernels to the range block. Let Δ_{r_k} denote the mean distance from each kernel in D_K to the current range block obtained from (1) and let γ_{kernel} be a scalar constant controlling how many domain kernels are included in the descriptions. The kernel to range block similarity is given by $w_{[r_k,d_\ell]} = \max\{\gamma_{\mathrm{kernel}}\Delta_{r_k} - \delta_{k\mathrm{PIFS}}(r_k, d_\ell),\ 0\}$ for each $d_\ell \in D_K$ to form a vector of similarities which is normalized yielding a range block descriptor in the form of a distribution of domain kernels. Intuitively each $w_{[r_k,d_\ell]}$ describes the error in fitting kernel d_ℓ to block r_k.

2.2 Top-down decomposition

In the first step of the top-down procedure we begin the construction of the quadtree by decomposing the image to some start level l_{start}, where level 1 is the root covering the entire image, by splitting the region nodes at each level into 4 child subregion nodes. Once we are at level l_{start} we calculate a descriptor histogram for each of the $2^{2(l_{\mathrm{start}}-1)}$ region nodes by summing the kPIFS descriptors making up each region and normalizing. From this point onward iterative transformations for each node at the current level are constructed based on the local spatial neighborhoods and are applied to each of the nodes until an approximate convergence is reached. At this point stable regions are identified and the next level of the quadtree is constructed from the children of the nodes based on some stability (or discrepancy) measure.

In practice the choice of l_{start} in both the original and modified version is important in determining the resulting segments. If l_{start} is a small number then there is a risk that the region nodes identified as stable will still contain much heterogeneity while a larger l_{start} can result in an over-segmentation. We have experimentally found that $l_{\text{start}} = 6$ is a good choice as a start level, i.e. at 32×32 sub-image nodes.

The novel idea that we now introduce to this procedure addresses the iterative transformations that are applied to the nodes until convergence. The convergence of both the original transformation and the new one presented here rely on properties of contractive transformations in a metric space [16]. Here we briefly review the necessary concepts.

Definition 1 (Contractive Transformation). *Given a metric space (X, δ), a transformation $T : X \to X$ is called contractive or a contraction with contractivity factor s if there exists a positive constant $s < 1$ so that $\delta(T(x), T(y)) \leq s\delta(x, y)\ \forall x, y \in X$.*

Let us then denote $T^{\circ n}(x) = T \circ T \circ \cdots \circ T(x)$; that is, T composed with itself n times and applied to x. The property of contractive transformations that we are interested in is given in the following theorem which is proved in [16].

Theorem 1 (Contractive Mapping Fixed Point Theorem). *Let $(\mathbf{X}, \delta)$ be a complete metric space and let $T : \mathbf{X} \to \mathbf{X}$ be a contractive transformation, then there exists a unique point $x_f \in \mathbf{X}$ such that for all $x \in \mathbf{X}$ we have $x_f = T(x_f) = \lim_{n\to\infty} T^{\circ n}(x)$. The point x_f is called the* fixed point *of T.*

The importance of this theorem is that if we can show a transformation to be contractive in a defined metric space, then we are sure that some fixed point will be reached by applying the transformation iteratively. In both the original procedure and the updated version the metric space was defined as the set of image region descriptor histograms which can be thought of as lying in the space $\mathbb{R}^d$. It follows that any metric on $\mathbb{R}^d$ can be chosen, but in practice however we have just used the L_1 distance metric, denoted by δ_{L_1} and defined as $\delta_{L_1}(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^{d} |x_i - y_i|$.

In the original paper on the procedure [10] we proposed a transformation to perform an iterative weighted averaging of similar region descriptors within a local spatial neighborhood. Specifically, given some descriptor $\mathbf{w}_i$ at the current level of the quadtree, let $\mathcal{N}_i$ denote the set of $m \times m$ spatially local neighbor descriptors around $\mathbf{w}_i$ but not including $\mathbf{w}_i$, and let $\mu_{\mathcal{N}_i}$ be the average L_1 distance from $\mathbf{w}_i$ to all of the other descriptors in $\mathcal{N}_i$. We then denote a weighted average distance $t_{\mathcal{N}_i} = \psi\mu_{\mathcal{N}_i}$, where ψ is some weighting constant, and denote the set of close descriptors $\mathcal{N}_i^c = \{\mathbf{w}_j \in \mathcal{N}_i : d_{L_1}(\mathbf{w}_i, \mathbf{w}_j) \leq t_{\mathcal{N}_i}\}$. Then we define a transformation F_i for this descriptor to be the average of the descriptors $\mathcal{N}_i^c$ and $\mathbf{w}_i$. More explicitly:

$$F_i(\mathbf{w}) = \frac{1}{1 + |\mathcal{N}_i^c|} \left(\mathbf{w} + \sum_{\mathbf{w}_j \in \mathcal{N}_i^c} \mathbf{w}_j \right). \tag{2}$$

A transformation F_i was found for each $\mathbf{w}_i$ at the current level and it was applied iteratively to obtain updated descriptors, i.e. $\mathbf{w}_i^n = F_i^{\circ n}(\mathbf{w}_i)$, until $\delta_{L_1}(\mathbf{w}_i^n, \mathbf{w}_i^{n+1}) < \epsilon$ for some given error threshold ϵ. We claimed that each F_i was contractive and would thus yield a fixed point descriptor based on a result from Van der Vaart and Van Zanten

[17]. While this appears sufficient, the proof is complicated and indirect and F_i takes a somewhat inconvenient form. Here we propose a simpler affine transformation where contractivity can easily be observed.

Our new transformation is also defined for each region descriptor at each level of the quadtree. Let $\mathbf{w}_i$, $\mathcal{N}_i$ and $t_{\mathcal{N}_i}$ be defined as above and let $\mathcal{N}_i' = \{\mathbf{w}_i\} \cup \mathcal{N}_i$. We now define a set of scalar weights for every descriptor in $\mathcal{N}_i'$ such that $s_{(i,j)}$ represents a measure of similarity between $\mathbf{w}_i$ and $\mathbf{w}_j$ for $\mathbf{w}_j \in \mathcal{N}_i'$. The weights are defined as $s_{(i,j)} = \max\{(t_{\mathcal{N}_i} - d_{L_1}(\mathbf{w}_i, \mathbf{w}_j))/c_i, 0\}$, where c_i is a normalization constant so that $\sum_{j=1}^{|\mathcal{N}_i'|} s_{(i,j)} = 1$. In this way all $s_{(i,j)} \leq 1$, and each descriptor $\mathbf{w}_j \in \mathcal{N}_i$ has an associated similarity weight $s_{(i,j)}$ with the special scalar $s_{(i,i)}$ being the weight for the $\mathbf{w}_i$. Now define a new descriptor $\mathbf{v}_i$ to be a linear combination of the descriptors in $\mathcal{N}_i$ as $\mathbf{v}_i = \sum_{\mathbf{w}_j \in \mathcal{N}_i} s_{(i,j)}\mathbf{w}_j$, and our affine transformation G_i for descriptor $\mathbf{w}_i$ is given by

$$G_i(\mathbf{w}) = s_{(i,i)}\mathbf{w} + \mathbf{v}_i. \tag{3}$$

Again we iteratively apply G_i to $\mathbf{w}_i$ obtaining $\mathbf{w}_i^n = G_i^{\circ n}(w_i)$ until convergence, but here due to the simple affine form of G_i it is particularly easy to demonstrate the contractivity of the transformation. For arbitrary descriptors $\mathbf{x}, \mathbf{y} \in \mathbb{R}^d$ we have $\delta_{L_1}(G_i(\mathbf{x}), G_i(\mathbf{y})) = \sum_{j=1}^{d} |(s_{(i,i)}x_j + \mathbf{v}_{i_j}) - (s_{(i,i)}y_j + \mathbf{v}_{i_j})|$. Notice that the $\mathbf{v}_{i_j}$'s all cancel out and the $s_{(i,i)}$ can be factored out, simplifying to $\delta_{L_1}(G_i(\mathbf{x}), G_i(\mathbf{y})) = s_{(i,i)} \sum_{j=1}^{d} |x_j - y_j| = s_{(i,i)}\delta_{L_1}(\mathbf{x}, \mathbf{y})$ and since $s_{(i,i)} \leq 1$, we have that G_i is either contractive or it does not move $\mathbf{w}_i$ at all, either way we are guaranteed by theorem 1 to reach a fixed point descriptor which we can denote by $\overline{\mathbf{w}}_i$. In practice the convergence is quite fast and we generally need less than 10 iterations for $\epsilon = 0.01$.

When the fixed point descriptors $\overline{\mathbf{w}}_i$ are reached for all regions at the current level, we identify the stability of each region based on the discrepancy of its fixed point to the fixed point of its neighbors. Since both F_i and G_i average each $\mathbf{w}_i$ with its similar neighbors, there is a strong possibility that sub-images in the regions with high local discrepancy after the iterative procedure will cover different textures. To avoid misclassifications we split and repeat the contractive mappings on these regions at the next level of the quadtree, as illustrated in Figure 2(a). The discrepancy of a node is measured by comparing $\overline{\mathbf{w}}_i$ to the fixed points of its four spatially nearest neighbors which we denote by the set $\overline{\mathcal{N}}_i$. Let $\mu_{\overline{\mathcal{N}}_i}$ denote the average L_1 distance from $\overline{\mathbf{w}}_i$ to the descriptors in $\overline{\mathcal{N}}_i$ and let $m_{\overline{\mathcal{N}}_i}$ denote the maximum distance from $\overline{\mathbf{w}}_i$ to $\overline{\mathcal{N}}_i$, then the discrepancy measure of the region is defined as $\mathcal{D}_i = \mu_{\overline{\mathcal{N}}_i} + m_{\overline{\mathcal{N}}_i}$.

Though we are only concerned with splitting and reprocessing unstable regions, in practice all regions are split. From $\mathcal{D}_i$ we are able to calculate a border measure for each node as $\mathcal{B}_i = \mathcal{D}_i / \max\{\mathcal{D}_j : j \in \{1, \ldots, N_k\}\}$ where N_k is the total number of nodes at the current decomposition level. $\mathcal{B}_i$ determines how $\overline{\mathbf{w}}_i$'s children descriptors are calculated. Let $\{\mathbf{w}_{(i,j)} : j \in \{1, \ldots, 4\}\}$ denote the 4 initial descriptors of $\overline{\mathbf{w}}_i$'s children used in the next level of the quadtree. If $\mathcal{B}_i = 0$ then the region is stable and there is no chance of $\overline{\mathbf{w}}_i$ covering a boundary region and so we assign $\mathbf{w}_{(i,j)} = \overline{\mathbf{w}}_i$ for all children. When $\mathcal{B}_i > 0$ we let $\{\mathbf{v}_{(i,j)} : j \in \{1, \ldots, 4\}\}$ denote the descriptors of the child regions calculated as the normalized sum of kPIFS histograms in the same

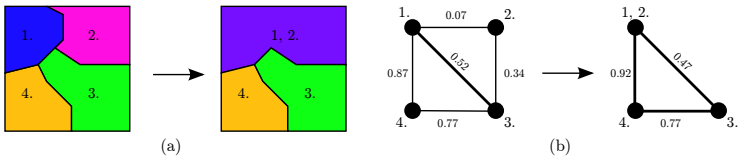

Fig. 4. Bottom-up merging of image regions. Part (a) shows the obtained segments and (b) show the corresponding graph. Edge weights are given similarity between the segments. In the right hand of (a) and (b) segments 1 and 2 of the left sides of (a) and (b) are merged.

manner as at the starting level l_{start}. Then we obtain the new descriptors as $\mathbf{w}_{(i,j)} = (1 - \mathcal{B}_i)\overline{\mathbf{w}}_i + \mathcal{B}_i\mathbf{v}_{(i,j)}$.

2.3 Bottom-up merging of regions

Upon the completion of the top-down procedure we obtain a quadtree decomposition of the image with leaves representing non-overlapping stable image regions. The goal of the bottom-up procedure is to merge these leaves into homogeneous clusters which form the final segmentation.

In our original approach we fit a mixture of Gaussians to the distribution of leaf nodes $\overline{\mathbf{w}}_f$ using the approach of Figueiredo [18] and the final segmentation was found by the Gaussian that gave the highest probability.

Our new approach begins by forming a planar graph G so that the vertices of G are the leaf nodes and an edge (i, j) is formed between vertices representing adjacent image regions with edge weight equal to $\delta_{L_1}(\overline{\mathbf{w}}_i, \overline{\mathbf{w}}_j)$, the distance between the associated fixed point descriptors. The bottom-up procedure then merges adjacent vertices of G based on edge weight. Let α_i denote the percentage of the total image covered by vertex i. Then α_i is considered in the merging, so the smallest regions will be forced to merge with the most similar neighboring region and when merging any two vertices i, j the ratio α_i/α_j is considered so that the merged vertex has a descriptor which is mostly influenced by the relatively larger region.

The merging of vertices is done in two steps. Initially we merge all vertex pairs i, j where the edge weight is close to 0, i.e. less than some small positive ϵ. These regions had nearly identical fixed points and the disparity is most likely only due to the fact that the fixed point is approximated. In the second step we let Δ_G denote the average weight in the current graph G which is updated after each merging is performed. We proceed in merging the vertices i, j with the smallest current edge weight until the relative weight $\delta_{L_1}(\overline{\mathbf{w}}_i, \overline{\mathbf{w}}_j)/\Delta_G$ is larger than some threshold $\gamma_{\text{merge}} \in [0, 1)$. Figure 4 gives an illustration of the process.

3 Experiments

In this section we show the experimental results of our procedure. The images used for testing our procedure are from the Berkley image database [19] and the Brodatz textures

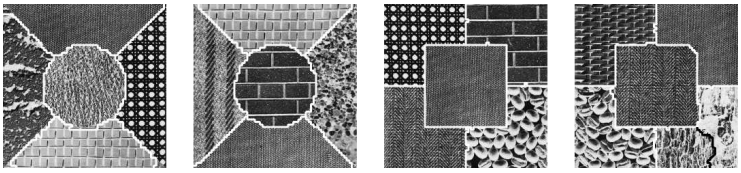

Fig. 5. Segmentation of the Brodatz textures [11]. The composition of the textures is inspired by the segmentation procedure of Fauzi and Lewis [3]. Segmentations borders are marked with white lines except (h) where a part in the lower right is marked in black to make it visible.

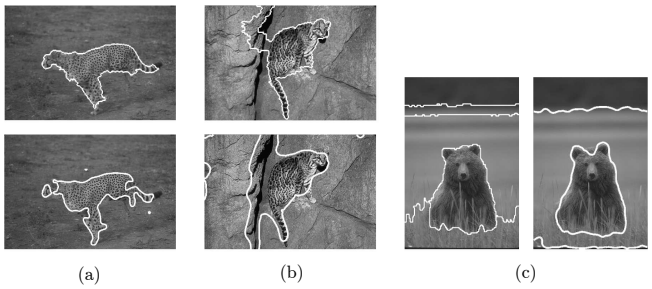

Fig. 6. Comparative results. This figure shows our results compared to that of Hong et al.[7]. Our results are on the top in (a) and (b) and right in (c).

[11]. Our procedure has shown to be very powerful for texture segmentation, which is demonstrated by comparing our results to state of the art methods of Fauzi and Lewis [3], Houhou et al.[8], and Hong et al.[7].

In Fauzi and Lewis [3] they perform unsupervised segmentation on a set of composed Brodatz textures [11]. We have compared the performance of our method to theirs by making a set of randomly composed images from the same set of Brodatz textures. These composed images are very well suited to our method because the descriptors precisely cover one texture, so to challenge our procedure we changed the composition. Some examples of the results are shown in Figure 5. We obtain very good segmentation for all images with only small errors along the texture boundaries. In 19 of 20 images we found the correct 5 textures and only the texture in the lower right hand corner of the last image was split into two. It should be noted that this texture contains two homogenous areas. In [3] only 7 of 9 composed images were accurately segmented. These results show that the texture characterization is quite good. But the challenge of textures in natural images is larger, as we will show next.

We have tested our procedure on the same set of images from the Berkley segmentation database [19] as was used in Hong et al.[7] and Houhou et al.[8]. The results are compared in Figures 6 and 7. Our method preforms well compared to that of Hong et al., especially in Figures 6(a) and (c). It should be noted that the focus of that paper was also on texture scale applied to segmentation. The results compared to the method of

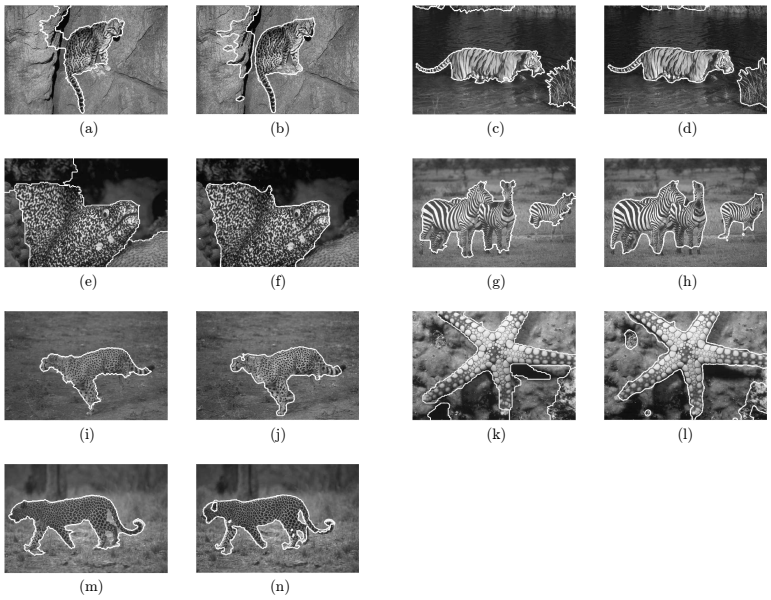

Fig. 7. Comparative results. This figure shows our results in columns one and three compared to the results from Houhou et al.[8] in columns two and four.

Houhou et al.are more alike and both methods find the interesting segments in all images. In Figures 7(e) and (f) our method finds some extra textures which are clearly distinct. In Figures 7(k) and (l) both methods find segments that are not part of the starfish, but are clearly distinct textures. There are slight differences in the two methods, e.g. in Figures 7(a) and (b) where the object is merged with a part of the background in our method, whereas it is found very nicely in the method of Houhou et al. [8]. An example in favor of our procedure is Figures 7(m) and (n) where part of the head and the tail is not found very well by their method, whereas it is found very well by our procedure.

4 Conclusion

Texture poses a great challenge to segmentation methods, because textural patterns can be hard to distinguish at a fine scale making precise boundary detection difficult. We have presented a novel, computationally efficient approach to segmentation of texture images. To characterize the local structure of the image, we begin by a top-down decomposition in the form of a hierarchical quadtree. At each level of this tree a contractive transformation is computed for each node and is iteratively applied to generate a novel encoding of the sub-images. The hierarchical decomposition is controlled by the stability of the encoding associated with nodes (sub-images). The leaves of this quadtree and their incidency structure with respect to the original image will form a planar graph in a natural way. The final segmentation will be obtained from a bottom-up merging process

applied to adjacent nodes in the planar graph. We evaluate the technique on artificially composed textures and natural images, and we observe that the approach compares favorably to several leading texture segmentation algorithms on these images.

References

1. Pal, N.R., Pal, S.K.: A review on image segmentation techniques. Pattern Recognition **26**(9) (January 1993) 1277–1294
2. Borenstein, E., Ullman, S.: Class-specific, top-down segmentation. In: Computer Vision - ECCV 2002 : 7th European Conference on Computer Vision, Copenhagen, Denmark, May 28-31, 2002. (2002) 639–641
3. Fauzi, M.F.A., Lewis, P.H.: Automatic texture segmentation for content-based image retrieval application. Pattern Anal. & App. **V9**(4) (November 2006) 307–323
4. Liu, Y., Zhou, X.: Automatic texture segmentation for texture-based image retrieval. MMM **0** (2004)
5. Malik, J., Belongie, S., Shi, J., Leung, T.: Textons, contours and regions: Cue integration in image segmentation. In: IEEE ICCV. (1999) 918–925
6. Zeng, G., Van Gool, L.: Multi-label image segmentation via point-wise repetition. Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on (June 2008) 1–8
7. Hong, B.H., Soatto, S., Ni, K., Chan, T.: The scale of a texture and its application to segmentation. 2008 IEEE Conference on Computer Vision and Pattern Recognition (2008) 1–8
8. Houhou, N., Thiran, J., Bresson, X.: Fast texture segmentation model based on the shape operator and active contour. In: CVPR. (2008) 1–8
9. Bagon, S., Boiman, O., Irani, M.: What is a good image segment? a unified approach to segment extraction. In D., F., Torr, P., A., Z., eds.: Computer Vision – ECCV 2008. Volume 5305 of LNCS., Springer (2008) 30–44
10. Dahl, A., Bogunovich, P., Shokoufandeh, A., Aanæs, H.: Texture segmentation from context and contractive maps. *Submitted to* Computer Vision and Pattern Recognition, 2009. IEEE Conference on (2009)
11. Brodatz, P.: Textures; a photographic album for artists and designers. (1966)
12. Jacquin, A.E.: Image coding based on a fractal theory of iterated contractive image transformations. IP **1**(1) (January 1992) 18–30
13. Fisher, Y.: Fractal Image Compression - Theory and Application. Springer-Verlag, New York (1994)
14. Alexander, S.: Multiscale Methods in Image Modelling and Image Processin. PhD thesis (2005)
15. Xu, Y., Wang, J.: Fractal coding based image retrieval with histogram of collage error. Proceedings of 2005 IEEE International Workshop on VLSI Design and Video Technology, 2005. (2005) 143–146
16. Rudin, W.: Principles of Mathematical Analysis. Third edn. McGraw-Hill (1976)
17. van der Vaart, A.W., van Zanten, J.H.: Rates of contraction of posterior distributions based on gaussian process priors. The Annals of Statistics **36**(3) (June 2008) 1435–1436
18. Figueiredo, M.A.T., Jain, A.K.: Unsupervised selection and estimation of finite mixture models. In: in Proc. Int. Conf. Pattern Recognition. (2000) 87–90
19. 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: ICCV. Volume 2. (July 2001) 416–423