Dictionary Based Image Segmentation

Anders Bjorholm Dahl(✉) and Vedrana Andersen Dahl

Department of Applied Mathematics and Computer Science,
Technical University of Denmark, Lyngby, Denmark
{abda,vand}@dtu.dk

Abstract. We propose a method for weakly supervised segmentation of natural images, which may contain both textured or non-textured regions. Our texture representation is based on a dictionary of image patches. To divide an image into separated regions with similar texture we use an implicit level sets representation of the curve, which makes our method topologically adaptive. In addition, we suggest a multi-label version of the method. Finally, we improve upon a similar texture representation, by formulating the computation of a texture probability in terms of a matrix multiplication. This results in an efficient implementation of our segmentation method. We experimentally validated our approach on a number of natural as well as composed images.

1 Introduction

Typically, image segmentation is based on obtaining regions with boundaries in between. This involves modeling of both the image data and the segment boundaries. Approaches ensuring boundary smoothness include modeling the connection between image elements using e.g. Markov random fields [18] which has been efficiently solved using *st*-cut [1,27]. Modeling the boundary using an explicitly represented deformable curve is another approach that originates from the snakes model [17].

Deformable curves may also be represented using level sets [5], with a number of methods suggesting to use a global information obtained from the curve to guide the segmentation, e.g. Chan and Vese [8] and Yezzi Jr et al. [34]. These methods do not rely on a well defined intensity gradient between regions, but still assume that regions are distinguished by their grey level.

In some cases, however, the average gray level is similar between regions and only textures differ. Textures must therefore be handled differently than intensity and color. The typical approach is to use a texture characterization, mapping the image to a texture descriptor space for segmentation in a similar manner to the intensity method. Here the assumption is that descriptors within textures are similar while they differ between textures.

Such an approach was suggested by Chan and Vese [8] using texture orientation, which has been extended in e.g. Rousson et al. [28] using the structure tensor. Here they estimate the probability of different regions based on estimates of the joint probability of the elements of the structure tensor to evolve the level

R.R. Paulsen and K.S. Pedersen (Eds.): SCIA 2015, LNCS 9127, pp. 26–37, 2015.
DOI: 10.1007/978-3-319-19665-7_3

set. The structure tensor is, however, estimated at a certain scale and in images with textures at different scale the segmentation might fail. To overcome this problem a local texture scale estimate based on total variation was proposed [3,4], where the scale estimate is used for obtaining an improved structure tensor characterization and hereby an improved segmentation. In Brox et al. [2] they use diffusion based on the structure tensor to obtain improved segmentation.

Many other texture descriptors characterizing the local image structure that allow for discriminating between textures have been suggested. These include local fractal features [32], gradient histograms [11,29], local binary patterns [25], textons [22], and more. Often images contain texture on different scale that can be deformed or rotated versions of the same texture. Typically, this is handled in by designing descriptors invariant to such properties.

A related approach for image segmentation is based on sparse dictionaries of image patches [12,19]. Coding the image using sparse dictionaries has shown impressive results for image processing problems like denoising or inpainting. The idea for segmentation is to utilize the strong reconstructive properties of sparse coding by building a dictionary for each texture class. High segmentation performance is obtained by utilizing that the texture class used for learning the dictionary can be reconstructed well whereas other texture classes cannot. Methods focusing on optimal reconstruction have been suggested [26,30], and improved performance has been obtained by also optimizing for discrimination [20,21]. Recently Gao et al. [13] suggested to use sparse dictionaries together with an active contour for segmentation. The algorithm learns the sparse dictionaries from rough user input in the image that must be segmented, and they show accurate segmentations of natural images.

Our texture representation is based on a dictionary of image patches similar to our earlier work [9], where the dictionary is obtained from the image that we want to segment. This approach does not assume any characteristics of the texture, instead we consider texture being information obtained from the image at a given scale. In this broad sense, a texture may also be intensity or color. As such, this dictionary based texture representation is different and in some ways simpler than both texture descriptors and sparse coding. The method [9] uses snakes to divide an image into a object and background. In contrast, in this work we evolve a curve using a level sets representation. By doing so, we obtain topological adaptivity, where disjoint regions with similar texture can be segmented to have the same label. In addition, our novel method can segment an image into an arbitrary number of textures, instead of the two in the formulation from [9]. Finally, we provide an improved texture encoding based on matrix multiplication, resulting in a efficient implementation of the method.

2 Method

Our algorithm is strongly motivated by a region based segmentation method, called active contours without edges, which was originally proposed by Chan and Vese [6] (see also [34] for a related approach) and has led to a number of extensions [7,33] and various numerical implementations [14,16,31]. Our segmentation

uses the same fundamental principle but we utilize a dictionary-based texture representation. In the description of our method, we will briefly go through the basic elements of active contours without edges in order to clarify the connection to our model. When explaining our texture representation, we start by considering two labels and then generalize to a multi-label segmentation.

Region Based Curve evolution. Early deformable models [5,17,23] are based on finding edges in the image, which provides only local support and is unsuitable for noisy or textured images. Region based approaches [6,8], on the other hand, utilize the global information obtained from the curve to guide the segmentation.

For example, consider an image $I : \Omega \subset \mathbb{R}^2 \rightarrow \mathbb{R}$ containing an object and a background characterized by two different intensities. A curve initialized in the image, and represented by a zero level set of a function $\phi : \Omega \rightarrow \mathbb{R}$, leads to labeling of pixels as either *inside* or *outside*. The mean label intensities m_{in} and m_{out} are calculated given this labeling, and here we assume that *inside* contains more of the object, while *outside* contains more of the background. Now the curve is evolved to segment the image. The curve shrinks where pixel intensities are close to m_{out}, while the curve expands where pixel intensities are close to m_{in}. The value $(m_{\text{in}} + m_{\text{out}})/2$ defines a threshold between shrinking and expanding. This two-step proces is repeated. As the curve evolves, mean label intensities are recalculated to more accurately discriminate the intensities of the object and the background, and the curve eventually segments out the object.

Such an evolution of a zero level set is given by

$$\frac{\partial \phi}{\partial t} = \delta_\epsilon(\phi) \left[(m_{\text{out}} - m_{\text{in}})(2I - m_{\text{out}} - m_{\text{in}}) + b\kappa \right] , \tag{1}$$

where $\kappa = \nabla \cdot \left(\frac{\nabla \phi}{|\nabla \phi|} \right)$ is a curvature of the level set curve and the term weighted with b is minimizing the length of the curve. To extend the evolution to all level sets, and depending on the implementation of the level sets, a regularization of the delta function $\delta_\epsilon(\phi)$, may be replaced with $|\nabla \phi|$, or left out [14].

Texture Dictionary. The central part of our method is a dictionary based texture representation, with overlapping image patches being assigned to dictionary elements just as in [9,10]. In this section we will show how this dictionary assignment defines a transformation from an arbitrary labeling of the image into a related probability image. If parts of the image are labeled as *inside* or *outside* this transformation will result in pixel-wise probability of belonging to *inside* or *outside*. The probabilities are computed based on the textures present in the two labels. Having a probability image, we can evolve the curve so that it shrinks where probability for inside is smaller and expands where probability for inside is larger. Just as with region based image segmentation, we can iterate this two-step process, by recomputing probabilities and evolving the curve.

To construct the dictionary we extract a certain number of $M \times M$ patches from an image, collect pixel intensities in patch-vectors of length $m = M^2$, and cluster the patches in n clusters using the k-means algorithm with Euclidian

distance. It is the cluster centers that define our dictionary. Figure 1 shows a small image and a small dictionary computed from the image. In this case, the first dictionary element represents the background, elements 2–8 represent the textured object, while elements 9-16 represent transitions from the object to the background. In general, to make sure that the nature of the textures present in an image is captured by the dictionary, we need a large number, typically a few hundred, of dictionary elements, which results in a significant redundancy. Having a texture dictionary we can assign overlapping image patches to the closest dictionary element, again using Euclidian distance. This assignment, also shown in Figure 1 is crucial for our method, because it defines a binary relation between image pixels and dictionary pixels.

Notice the following important property of this construction. Each image patch is assigned to a single dictionary element, but since image patches are overlapping, every image pixel relates to $m = M^2$ dictionary pixels. (Image pixels in a margin of width $M - 1$ relate to less than m dictionary pixels.) In other words, an assignment of a certain image patch to a certain dictionary element makes m pixels from the image patch relate to m pixels from the dictionary element. This binary relation between image pixels and dictionary pixels may be represented using a sparse binary matrix $\mathbf{B}$ with $|\Omega|$ rows and nm columns, where $|\Omega|$ is a total number of image pixels and nm is a total number of dictionary pixels. Note that the matrix $\mathbf{B}$ captures the texture information of the image by simultaneously encoding two things: a dictionary assignment of each image patch and a spatial relationship between the patches. Notice also that for calculating the matrix $\mathbf{B}$ we only need an assignment image.

In case of having an RGB image as an input, we would collect intensities from all three color channels when constructing the dictionary. Once the assignment of image patches to dictionary elements is completed, all computations are the same for any number of color channels.

Label to Probability Transformation. Having an assignment image (i.e. being able to calculate $\mathbf{B}$) we can define the transformation from a label image to a probability image. A label image is an arbitrary partitioning of Ω in discrete labels, see Figure 1. In case of *inside* or *outside* segmentation, it is sufficient to consider only one label, e.g. *inside*. For this one label, a label image is a binary map $L_{\text{in}} : \Omega \to \{0, 1\}$. Each patch from an image I has a corresponding (i.e. extracted from the same position in the image space) label patch from L_{in}. In turn, each dictionary element has a number of image patches assigned to it. This allows us to compute labels of dictionary elements. A label of a dictionary element is computed as a pixel-wise average of the label patches corresponding to image patches that are assigned to the dictionary element in question. Due to averaging, dictionary labels are no longer binary. Figure 1 shows how dictionary labels capture the texture information, e.g. the labels of the dictionary elements representing the transition from the texture to the background show this transition.

Dictionary labels can be computed efficiently by arranging the pixels of the label image in a binary vector $\mathbf{l}_{\text{in}}$ and multiplying with $\mathbf{B}$ which is normalized

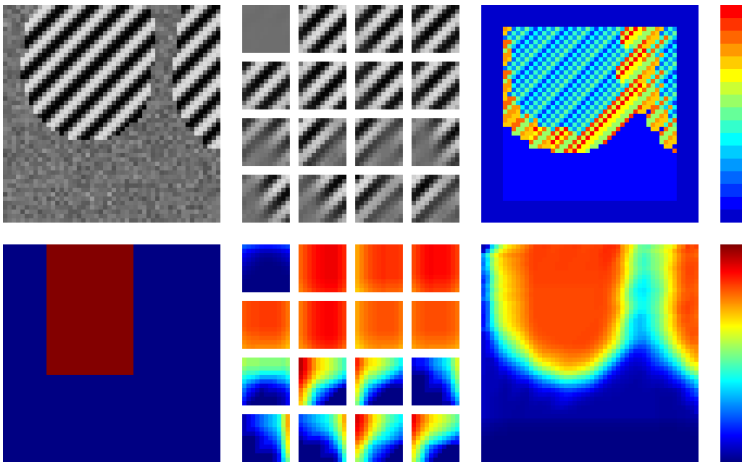

Fig. 1. Elements of the dictionary based texture representation. For a better illustration, both the image and the dictionary are small. Top left: input image. Top middle: dictionary consisting of 16 dictionary elements, ordered in rows. Top right: assignment of the image patches to the dictionary elements, shown as a color of the central patch pixel. The colorbar (far right) runs from darkest blue color representing 0 to red representing 16. Bottom left: An initial labeling of the image. Bottom middle: A dictionary labeling shown in the same order as dictionary elements. Bottom right: A probability image. The colorbar (far right) for the dictionary labeling and the probability image runs from 0 to 1.

so that all its rows sum to 1. The resulting vector

$$\mathbf{d}_{\text{in}} = \text{diag}(\mathbf{B1})^{-1}\mathbf{B}\,\mathbf{l}_{\text{in}} \tag{2}$$

contains a pixel-wise frequency of dictionary elements belonging to *inside*. Here, $\mathbf{1}$ is a column vector of ones. To obtain dictionary labels, elements of $\mathbf{d}_{\text{in}}$ need to be re-arranged according to the size of the dictionary.

Labels $\mathbf{d}_{\text{in}}$ and $\mathbf{d}_{\text{out}} = 1 - \mathbf{d}_{\text{in}}$ are biased due to the ratio of the area inside $|\Omega_{\text{in}}|$ and area outside $|\Omega_{\text{out}}| = |\Omega| - |\Omega_{\text{in}}|$. For example, had the initial labeling covered just a small part of the textured object, the frequency of *inside* label would be small also for the dictionary elements representing texture. To alleviate this we define a pixel-wise normalization function

$$\tilde{d}_{\text{in}} = \frac{1}{Z}\frac{d_{\text{in}}}{|\Omega_{\text{in}}|}, \quad Z = \frac{d_{\text{in}}}{|\Omega_{\text{in}}|} + \frac{d_{\text{out}}}{|\Omega_{\text{out}}|}, \tag{3}$$

which operates on each element of $\mathbf{d}_{\text{in}}$.

The next transformation involves computing pixel-wise image probabilities from the dictionary labels. This is again performed by averaging. Each dictionary label is placed in the image space at the positions of image patches that are assigned to the dictionary element in question. Due to the patch overlap, up to m values need to be averaged to compute a pixel probability.

The efficient computation is performed by multiplying

$$\mathbf{p}_{\text{in}} = \text{diag}(\mathbf{B}^{\text{T}}\mathbf{1})^{-1}\mathbf{B}^{\text{T}}\,\tilde{\mathbf{d}}_{\text{in}}\,. \tag{4}$$

Rearranging elements of $\mathbf{p}_{\text{in}}$ into an image grid results in the probability image $P_{\text{in}} : \Omega \to [0,1]$. Note that P_{in} is different from L_{in} because image patches from both inside and outside may be assigned to the same dictionary element. The binary values from L_{in} will therefore diffuse according to the texture information encoded in $\mathbf{B}$. We will utilize this diffusion to drive the curve evolution.

Our way of obtaining a probability image is closely related to the method described in [9]. However, our approach is more efficient. In [9] a sequences of patch averaging is performed every time a probability image needs to be computed. Here we notice that the relation between dictionary and image is unchanged, and that computing probability image includes two linear transformations. This allows us to precompute the matrix $\mathbf{B}$, significantly speeding up the computation of a probability image. Note that $\mathbf{B}$ is a binary and sparse matrix, so storing this large matrix may be done space efficiently.

Multiple Labels. To handle multiple labels, and not just *inside* or *outside*, we create a layered label image with layers L_1 to L_K. Each layer is a binary indicator of a label, and layers sum to one in each pixel. The transformation (2) is applied to each layer, resulting in dictionary labels $\mathbf{d}_1$ to $\mathbf{d}_K$. Area normalization is now performed pixel-wise for all layers

$$(\tilde{d}_1, \tilde{d}_2, \ldots, \tilde{d}_K) = \frac{1}{Z}\left(\frac{d_1}{|\Omega_1|}, \frac{d_2}{|\Omega_2|}, \ldots, \frac{d_K}{|\Omega_K|}\right), \quad Z = \sum_{k=1}^{K} \frac{d_k}{|\Omega_k|}\,. \tag{5}$$

After area normalization the transformation (4) is applied to each $\tilde{\mathbf{d}}_k$, resulting in K probability images, P_1 to P_K, which sum to one in each pixel, but are no longer binary.

Curve Evolution. We can now define a curve evolution for texture segmentation. Again we initially consider segmentation into *inside* and *outside*. A closed curve represented as a zero level set of a function $\phi : \Omega \to \mathbb{R}$ defines a label image L_{in} which attains value one where ϕ is negative and zero otherwise. The label image is transformed into probability image P_{in} as described above. Curve points at locations with large P_{in} should move outwards, curve points with large $P_{\text{out}} = 1 - P_{\text{in}}$ should move inwards, and the curve should converge in a band where $P_{\text{in}} = P_{\text{out}}$. To obtain the desired behaviour we define a curve evolution as

$$\frac{\partial \phi}{\partial t} = \frac{1}{2} - P_{\text{in}} + b\kappa|\nabla\phi|\,, \tag{6}$$

with a curve length minimization term as in (1). Notice that $0.5 - P_{\text{in}} = 0.5(P_{\text{out}} - P_{\text{in}})$.

To segment multiple labels we represent each of the K labels with a single level set function ϕ_k, $k = 1, \ldots, K$. Using such an approach a care has to be

taken to avoid vacuum and overlap [35]. Indeed, if we generalized to multiple labels by evolving each level set ϕ_k using (6) and a corresponding probability image P_k, a vacuum would occur. This is because probabilities sum to one in each pixels, and situation might occur where none of the label probabilities P_k are larger than 0.5 in some places, leading to all level set curves shrinking, and causing vacuum. Therefore we perform a following pixel-wise transformation of probabilities for all labels

$$(\tilde{p}_1, \tilde{p}_2, \ldots, \tilde{p}_K) = \left(\frac{p_1}{p_1 + \max_{j \neq 1}(p_j)}, \frac{p_2}{p_2 + \max_{j \neq 2}(p_j)}, \ldots, \frac{p_K}{p_K + \max_{j \neq K}(p_j)} \right) . \quad (7)$$

Basically, we normalize the probability p_k not by using the sum of all $p_j, j = 1, \ldots, K$, but only considering the most probable of other labels. Resulting probabilities $(\tilde{p}_1, \tilde{p}_2, \ldots, \tilde{p}_K)$ have the property that the two largest values sum to 1, therefore avoiding vacuum (as at least one value is larger or equal to 0.5) and avoiding overlap (as only one value may be larger than 0.5). Notice also that the transformation defined by (7) reduces to an identity for $K = 2$.

The resulting level set evolution for a multi-label segmentation is

$$\frac{\partial \phi_k}{\partial t} = \frac{1}{2} - \tilde{P}_k + b\kappa |\nabla \phi_k|, \quad k = 1, \ldots, K . \quad (8)$$

Our relative probabilities $\tilde{P}_k$ do not guarantee elimination of vacuum and overlap. However, based on experiments we concluded that the level set curves align well.

Algorithm. Our deformable model for the dictionary based image segmentation is initialized with an image I, an initial curve ϕ^0 (or, in the case of multiple labels curves ϕ_k^0, but we will from now leave out the subscript k) and a few parameters defining the dictionary: patch size m, dictionary size n and normalization flag. Normalization flag indicates whether the image patches have been normalized to unit length. In a preprocessing step, the dictionary is constructed, overlapping image patches are assigned to the dictionary patches, and a sparse binary matrix $\mathbf{B}$ is constructed.

After preprocessing, a curve is iteratively evolved. In each evolution step a (multi-layered) label image L is obtained by thresholding ϕ. The label image is transformed to a dictionary labels using (2), area normalization is performed as in (5), and result is transformed back to the image using (4). In the case of a multi-label segmentation the resulting probability images are transformed using (7). Finally, the curve is updated by (6) or (8) in a multi-label case, withe

$$\phi^{t+1} = \phi^t + \Delta t \frac{\partial \phi}{\partial t} \quad (9)$$

until convergence. This leaves us with the resulting segmentation $\hat{\phi}$ and the resulting probability images $\hat{P}$.

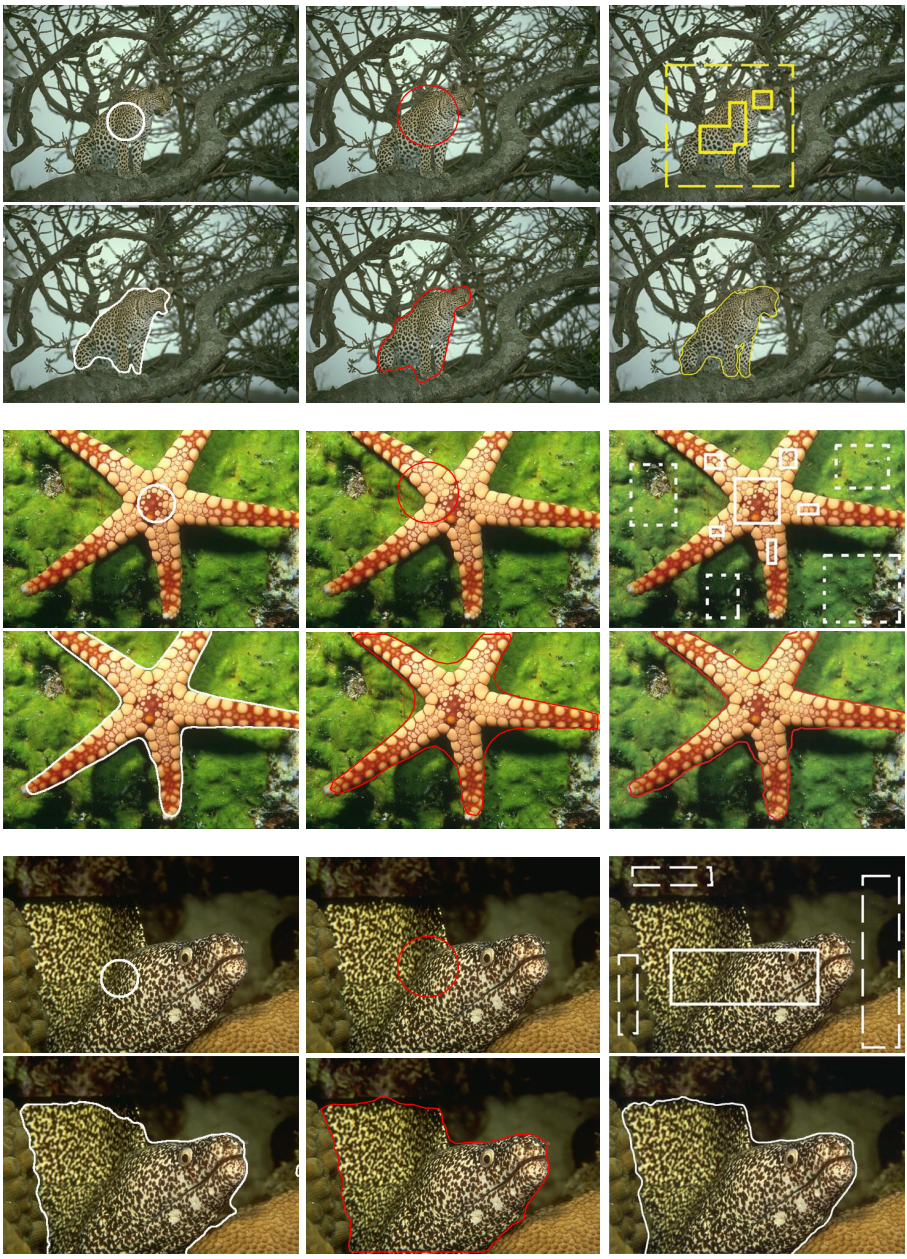

Fig. 2. Three leopard, star fish, and fish images. For each image we show the initialization at the top row and segmentation result at the bottom. Left – the proposed method. Middle – dictionary snakes [9]. Right – sparse texture active contours [13].

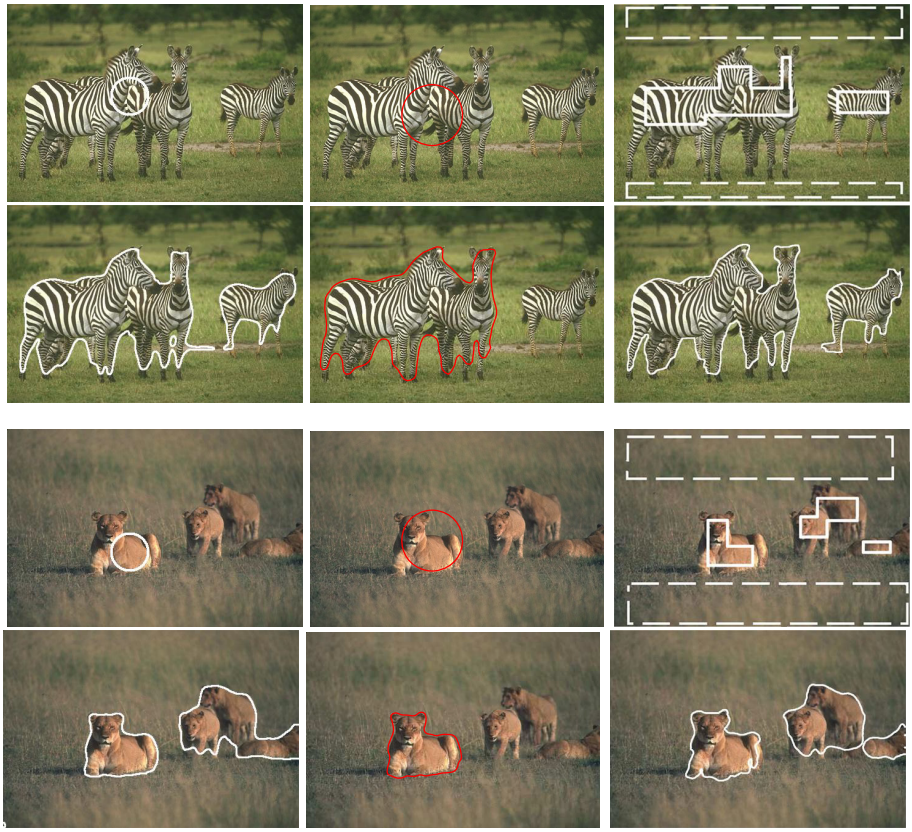

Fig. 3. Zebras and lions images. For each image we show the initialization at the top row and segmentation result at the bottom. Left – the proposed method. Middle – dictionary snakes [9]. Right – sparse texture active contours [13].

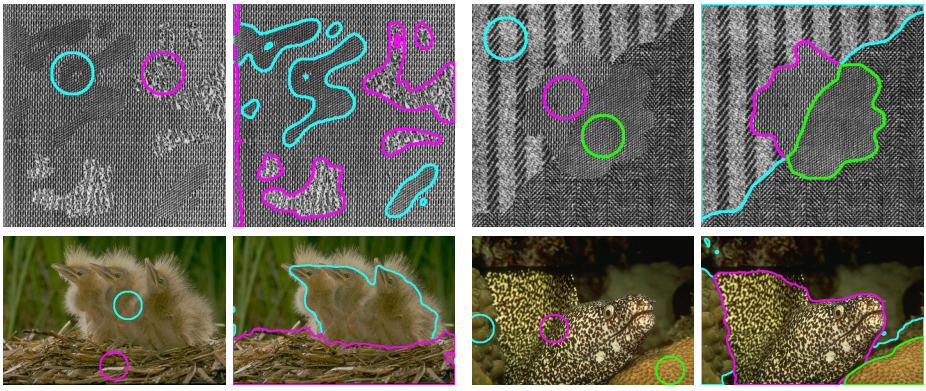

Fig. 4. Two composite and two natural images with three-label and four-label segmentation. For each image we show the initialization to the left and the segmentation to the right. One label (the background) is not shown by the curve, as it is the region not covered by other labels.

Implementation Details. Without the term minimizing the curve length, our iterative algorithm converges very fast, since the large time steps can be taken. On the other hand, the regularization by the length of the curve imposes a stringent time step restriction. This issue is addressed in [14–16] where fast approximations are obtained by replacing or supplementing the curve evolution with smoothing. Our current implementation is therefor as follows. When evolving the curve using a forward Euler step we use a large time step, ignoring the stability restrictions. This might introduce numerical errors, which we damp by smoothing the level set function with a Gaussian kernel.

3 Results

To demonstrate the strengths of our algorithm, we show results on the images from the Berkley segmentation dataset [24], such that we can make a direct comparison with related scientific work. We use the same parameters for all results shown in this section. Our dictionary is build from clustering 20000 randomly sampled image patches of size 3×3 pixels into 1000 clusters. The parameter b from (6) is set to 1.5 and the smoothing is performed by a Gaussian kernel with a standard deviation of 1.5.

Figures 2 and 3 show our inside-outside segmentation results compared to a number of images presented in [9] and [13]. The advantages of our method compared to [9] are topological adaptivity and a higher accuracy due to easier curve regularization. Compared to [13], our method accepts a much simpler initialization than the elaborated marking of both the object and the background.

Neither [9] or [13] support multiple labels, so for a multi-label case we bring only our results in Figure 4.

4 Discussion and Conclusion

Our texture representation with image patches clustered in a relatively large number of clusters might seem redundant. Indeed, some dictionary elements might be very similar. Therefore, a small variation in an image patch may result in a change in an assignment. This is, however, not an issue as long as similar dictionary elements have similar labels.

The experiments show a good discriminative properties of our texture representation, especially when we consider the simplicity and the general nature of the approach. Likewise, our segmentation results are encouraging compared to related work, especially considering the limited user input for initialization.

Acknowledgments. The authors acknowledge the financial support from CINEMA: the allianCe for ImagiNg of Energy MAterials (grant no. 1305-00032B) and NEXIM project (grant no. 11-116226), both funded by the Danish Council for Strategic Research.

References

1. Boykov, Y., Kolmogorov, V.: An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. TPAMI **26**(9), 1124–1137 (2004)
2. Brox, T., Rousson, M., Deriche, R., Weickert, J.: Colour, texture, and motion in level set based segmentation and tracking. Image and Vision Computing **28**(3), 376–390 (2010)
3. Brox, T., Weickert, J.: A tv flow based local scale measure for texture discrimination. In: Pajdla, T., Matas, J.G. (eds.) ECCV 2004. LNCS, vol. 3022, pp. 578–590. Springer, Heidelberg (2004)
4. Brox, T., Weickert, J.: A tv flow based local scale estimate and its application to texture discrimination. Journal of Visual Communication and Image Representation **17**(5), 1053–1073 (2006)
5. Caselles, V., Catté, F., Coll, T., Dibos, F.: A geometric model for active contours in image processing. Numerische mathematik **66**(1), 1–31 (1993)
6. Chan, T., Vese, L.A.: An active contour model without edges. In: Nielsen, M., Johansen, P., Fogh Olsen, O., Weickert, J. (eds.) Scale-Space 1999. LNCS, vol. 1682, pp. 141–151. Springer, Heidelberg (1999)
7. Chan, T.F., Sandberg, B.Y., Vese, L.A.: Active contours without edges for vector-valued images. Journal of Visual Communication and Image Representation **11**(2), 130–141 (2000)
8. Chan, T.F., Vese, L.A.: Active contours without edges. TIP **10**(2), 266–277 (2001)
9. Dahl, A.B., Dahl, V.A.: Dictionary snakes. In: ICPR (2014)
10. Dahl, A.L., Larsen, R.: Learning dictionaries of discriminative image patches. In: BMVC (2011)
11. Dalal, N., Triggs, B.: Histograms of oriented gradients for human detection. In: CVPR, vol. 1, pp. 886–893. IEEE (2005)
12. Elad, M.: Sparse and redundant representations: from theory to applications in signal and image processing. Springer (2010)
13. Gao, Y., Bouix, S., Shenton, M., Tannenbaum, A.: Sparse texture active contour. TIP (2013)
14. Gibou, F., Fedkiw, R.: A fast hybrid k-means level set algorithm for segmentation. In: 4th Annual Hawaii International Conference on Statistics and Mathematics, pp. 281–291. Hawaii, USA (2005)
15. Goldenberg, R., Kimmel, R., Rivlin, E., Rudzsky, M.: Fast geodesic active contours. TIP **10**(10), 1467–1475 (2001)
16. He, L., Osher, S.J.: Solving the chan-vese model by a multiphase level set algorithm based on the topological derivative. In: Sgallari, F., Murli, A., Paragios, N. (eds.) SSVM 2007. LNCS, vol. 4485, pp. 777–788. Springer, Heidelberg (2007)
17. Kass, M., Witkin, A., Terzopoulos, D.: Snakes: Active contour models. IJCV **1**(4), 321–331 (1988)
18. Li, S.Z.: Markov random field modeling in computer vision. Springer-Verlag New York, Inc. (1995)
19. Mairal, J., Bach, F., Ponce, J.: Task-driven dictionary learning. TPAMI **34**(4), 791–804 (2012)
20. Mairal, J., Bach, F., Ponce, J., Sapiro, G., Zisserman, A.: Discriminative learned dictionaries for local image analysis. In: CVPR, pp. 1–8. IEEE (2008a)
21. Mairal, J., Bach, F., Ponce, J., Sapiro, G., Zisserman, A.: Supervised dictionary learning (2008b). arXiv preprint arXiv:0809.3083

22. Malik, J., Belongie, S., Leung, T., Shi, J.: Contour and texture analysis for image segmentation. IJCV **43**(1), 7–27 (2001)
23. Malladi, R., Sethian, J.A., Vemuri, B.C.: Shape modeling with front propagation: A level set approach. TPAMI **17**(2), 158–175 (1995)
24. 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, vol. 2, pp. 416–423. IEEE (2001)
25. Ojala, T., Pietikainen, M., Maenpaa, T.: Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. TPAMI **24**(7), 971–987 (2002)
26. Peyré, G.: Sparse modeling of textures. Journal of Mathematical Imaging and Vision **34**(1), 17–31 (2009)
27. Rother, C., Kolmogorov, V., Blake, A.: "grabcut": Interactive foreground extraction using iterated graph cuts. ACM Trans. Graph. **23**(3), 309–314 (2004)
28. Rousson, M., Brox, T., Deriche, R.: Active unsupervised texture segmentation on a diffusion based feature space. In: CVPR, vol. 2, p. II-699. IEEE (2003)
29. Santner, J., Unger, M., Pock, T., Leistner, C., Saffari, A., Bischof, H.: Interactive texture segmentation using random forests and total variation. In: BMVC, pp. 1–12. Citeseer (2009)
30. Skretting, K., Husøy, J.H.: Texture classification using sparse frame-based representations. EURASIP journal on applied signal processing **2006**, 102 (2006)
31. Song, B., Chan, T.: A fast algorithm for level set based optimization. UCLA Cam Report **2**(68) (2002)
32. Varma, M., Garg, R.: Locally invariant fractal features for statistical texture classification. In: ICCV, pp. 1–8. IEEE (2007)
33. Vese, L.A., Chan, T.F.: A multiphase level set framework for image segmentation using the mumford and shah model. IJCV **50**(3), 271–293 (2002)
34. Yezzi Jr, A., Tsai, A., Willsky, A.: A statistical approach to snakes for bimodal and trimodal imagery. In: ICCV, vol. 2, pp. 898–903. IEEE (1999)
35. Zhao, H.K., Chan, T., Merriman, B., Osher, S.: A variational level set approach to multiphase motion. Journal of computational physics **127**(1), 179–195 (1996)