

Towards multi-stage texture-based active contour image segmentation

Daniel Reska¹ · Cezary Boldak¹ · Marek Kretowski¹

Received: 12 January 2016 / Revised: 2 October 2016 / Accepted: 17 November 2016 / Published online: 28 November 2016
© Springer-Verlag London 2016

Abstract In this paper, we present a three-stage approach to incorporation of texture analysis into a two-dimensional active contour segmentation framework. This approach allows to utilise texture information alongside other image features. The proposed method starts with an initial unsupervised feature computation and selection, then moves to a fast contour evolution process and ends with a final refinement stage. The algorithm is designed to be general in its nature and not restricted to any particular texture feature extraction method. In this paper, the initial stage generates a set of feature maps consisting of grey-level co-occurrence matrix and Gabor features. The implementation makes an extensive use of hardware acceleration for efficient calculation of a relatively large number of features. The performance of the method was tested on various synthetic and natural images and compared with results of other algorithms.

Keywords Image segmentation · Deformable models · Active contour · Texture analysis

1 Introduction

Segmentation is a common tasks in image processing and still an important topic of ongoing research. One of the most successful groups of segmentation algorithms are deformable models [16]—a class of methods based on a deforming shape that tries to adapt to a specific image region, extracting it from the background. Historically, the deformable models were best suited for regions with distinctive (but not necessary continuous) borders and fairly uniform interior intensity, as the influencing image forces used edge- and intensity-based features. This traditional approach, although highly effective in many applications, is often insufficient in case of regions with a distinctive texture. The mere task of texture recognition, easy for the human observer, is not trivial in the current state of the computer vision. As there is no universal model for the description of the image texture, the algorithms must rely on different feature extraction methods, like grey-level co-occurrence matrices [7], Gabor filters [11] or wavelet transforms [13]. Incorporation of the texture features into a deformable model-based method can increase its effectiveness and application possibilities, but it also has to deal with the various characteristics of the features and also keep in mind the performance issues.

In this paper, we propose a deformable model-based method for segmentation of images with various texture patterns. The method uses a multi-stage active contour driven by the textural features, alongside other image information. The segmentation process is divided into three stages: features calculation and selection, initial contour evolution and a refinement stage that fine-tunes the final result. The goal of the algorithm is to create a framework that employs different types of texture features. Currently, our implementation uses a set of grey-level co-occurrence matrices (GLCM) and

✉ Daniel Reska
d.reska@pb.edu.pl
Cezary Boldak
c.boldak@pb.edu.pl
Marek Kretowski
m.kretowski@pb.edu.pl

¹ Faculty of Computer Science, Bialystok University of Technology, Wiejska 45A, 15-351 Bialystok, Poland

Gabor features, generated with the help of graphics processing unit (GPU) acceleration.

2 Background and related work

Deformable model is an active shape (e.g. a 2D contour or a 3D surface) that tries to adapt to a specific image region. This adaptation process is usually influenced by external and internal forces that deform the shape towards the boundaries of the segmented region. The external forces attract the model to desired image features, while the internal forces control its smoothness and continuity. This formulation allows the models to overcome many problems, like image noise and boundary irregularities.

Deformable models were popularised with a seminal “snake” algorithm [12], which was a parametric active contour with an image force and a set of internal energies that controlled bending and stretching of the curve. Since then, the classical model has been heavily modified and extended, e.g. with addition of expansion forces [4], edge-based vector field energies [31], adaptive topology mechanisms [15] and region-based image energies [23]. Furthermore, different other models, like level set-based geodesic active contours [2] and region-based active contours without edges (ACWE) [3], were also created.

External forces can come from a variety of image features. Many segmentation methods commonly use edge [12,31] and region intensity statistics [23,32] to drive the segmentation process. However, larger patterns with high-contrast pose a greater challenge to traditional methods. In order to address this problem, methods combining deformable models with texture extraction and classification [19] were also introduced.

Texture-based methods generally fall into two categories: supervised and unsupervised. The supervised methods use information obtained from prior (and often controlled) texture analysis stage. Paragios and Deriche [17] proposed an extension of the geodesic active contour for texture segmentation. Their solution requires a given set of texture patterns that are present in the segmented image. An offline feature extraction and analysis are then performed on the pattern set. The computed information is used to generate region and boundary forces that influence the contour evolution. Pujol and Radeva [18] used a set of GLCM measures to generate the feature space, which was later reduced using linear discriminant analysis and was utilised to create a likelihood map that influenced the model deformation. Again, the method requires samples of the image texture patterns for the training stage.

Unsupervised methods usually utilise the features of the segmented region without earlier image analysis or pattern classification. Shen et al. [26] proposed a method for creat-

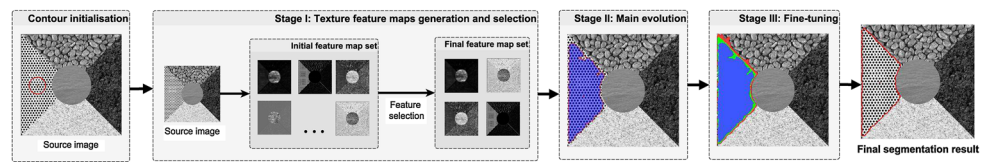
ing a likelihood deformation map from the intensity statistics of the initial region in addition to the boundary information. This solution was applied to 2D and 3D deformable models. A similar approach was used by Huang et al. [10], where the intensity characteristics were used to create the likelihood map for the textures with small texons, and a small bank of Gabor filters was utilised in the case of textures with larger patterns. Rousson et al. [24] proposed an unsupervised level set-based method that uses a variational framework based on texture features extracted with a structure tensor and nonlinear diffusion scheme. Tatu and Bansal [29] proposed a geodesic contour that uses intensity covariance matrices as the texture features. Savig et al. [25] created a texture edge detector from a Gabor feature space using the Beltrami framework [28] and integrated it with the geodesic and Chan-Vese models. The Beltrami framework was also used by Houhou et al. [9]. In this case, the proposed feature described not only the edge, but the properties of the entire texture region. The utilised active contour model was based on Kullback–Leibler distance and used a fast numerical segmentation scheme. Awate et al. [1] proposed a fast multi-stage level set method based on threshold dynamics [6] that uses general nonparametric statistical image neighbourhoods model instead of specific texture features. It provides automatic adaptation to image features and requires very few user-provided parameters, including the number of texture classes in the image and their scale. Recently, Wu et al. [30] presented a method based on ACWE with a global minimisation scheme, where GLCM and Gabor features are fused together to create final feature space.

3 Multistage texture-based active contour overview

The presented segmentation method is based on an active contour model equipped with a multistage evolution process. The goal of the evolution is to expand the contour shape according to the textural properties of its initial region. The evolution procedure consists of three main stages (see Fig. 1). Firstly, the algorithm generates a set of texture feature maps and selects the features that characterise the segmented region in the most meaningful way and which will serve as the influence of the deformation process. Next, the fast expansion stage moves the contour towards the edge of the segmented region using a strict textural stopping energy: it assumes that a sudden change in one of the selected features indicates a change in the texture pattern. Finally, the information collected in the previous stage is used to refine the contour shape.

The utilised parametric active contour is designed to segment a single continuous region, starting from a manual initialisation. In contrast to more global region-based methods, the contour provide a more local approach to the segmented region and its border, which results in a fast evolu-

Fig. 1 Diagram of the proposed method with the three stages



tion process that does not require any additional information about other texture classes in the image. The contour is also more sensitive to small local changes that could be lost in global region statistics.

3.1 Stage I: Texture feature extraction and selection

The first stage of the proposed method is responsible for the generation and selection of the texture features that will be used in the contour evolution process. The algorithm does not specify the type of the features. In fact, any method that can generate a feature map of the segmented image can be used. As there is no supervised texture classification step [17, 27] and the number of different textures in the image is not known, the selection condition picks the texture features that will distinguish only the segmented area from regions with different patterns regardless of their type or scale.

The generation stage begins with calculation of the texture features within the bounding box of the initial contour. The contour area should be large enough to sufficiently cover the texture pattern, preferably by including several texture pattern tiles. The features are calculated independently for each image pixel in the bounding box which creates a set of initial feature maps, denoted as T_{init} . Next, the selection step picks the maps with the most uniform feature values inside the contour. For the pixels covered by the initial snake in each of the feature maps t_i , the mean of the feature values \bar{x}_i , standard deviation σ_i and relative standard deviation $\%RSD(t_i) = \frac{\sigma_i}{\bar{x}_i} \times 100$ are calculated. The texture feature used in the next stages must have the $\%RSD$ lower than a user-specified threshold, as defined:

$$T_{\text{best}} = \{t_i \in T_{\text{init}} : \%RSD(t_i) < r\}, \quad (1)$$

where T_{best} is the set of selected features and r is the threshold (equal to 65% by default). Additionally to this uniformity detection, a set of extraction method-specific conditions can be employed to reduce the number of features in T_{best} for performance reasons. For example, generations of features in larger scales can be omitted in case of fine-grained patterns, as well as extraction of orientation-sensitive features in isotropic textures. On the other hand, generation of features in multiple scales is often desired to mitigate the influence of image resolution on the patterns, which allows the method to adapt to textures of different scale.

With the features selected, their maps are then calculated for the entire image. The final set T_{best} contains the features that will drive the evolution of the contour in next stages.

3.2 Stage II: Main contour evolution

The second stage performs a fast evolution process that expands the contour towards the edge of the segmented region. This stage is expected to find only a rough outline of the region, as its final refinement will be performed in the last stage.

In its present form, the method uses a discrete parametric contour [22]. The shape of the snake is composed of a set of points (snaxels) interconnected with line segments. The movement of each snaxel can be steered by a set of energies that describe the snaxel state in the given image position. The snake evolution process aims to minimise the energies of the snaxels by moving them to the positions of the lowest energy, i.e. a snaxel can be moved only if its energy in the potential destination will be lower than the energy in the current position. The calculations are performed iteratively, and the contour energy E_{snake} is minimised. E_{snake} is defined as:

$$E_{\text{snake}} = \sum_{s_i \in \text{snake}} E_{\text{int}}(s_i) + E_{\text{img}}(s_i) + E_{\text{con}}(s_i) \quad (2)$$

where s_i is a single snaxel, E_{int} is the internal energy (controlling the form of the contour), E_{img} represents the energy based on the image features and E_{con} is the energy of other constraints. The iterative evolution stops when there is no more movement of the snaxels or the number of displaced points is lower than a specified threshold.

The proposed method utilised a texture-based image energy, as well as a dynamic topology reformulation method [15] that allows the contour to refine its geometry in each iteration and reduces the necessity of strong internal forces. In each iteration, the position of every snaxel is updated by examining its neighbourhood and choosing a new position with lower textural energy. This energy E_{tex2}^s , proposed in [20], takes into consideration all the texture feature maps from T_{best} , selected in the first stage: the current snaxel s can be moved into the new position p only if a similarity condition is fulfilled for all the texture feature maps from T_{best} , as defined in:

$$E_{\text{tex}2}^s(p) = \begin{cases} \frac{1}{d(s,p)+1} & \text{if } \forall t \in T_{\text{best}} : \\ & |\text{val}_t(p) - \bar{x}_t| \leq \theta_2 \times \sigma_t \\ 1 & \text{otherwise,} \end{cases} \quad (3)$$

where $d(s, p)$ is the distance between the location on snaxel s and point p , t is a texture feature map in the T_{best} set, \bar{x}_t and σ_t are the feature mean and standard deviation of the snake initial region, $\text{val}_t(p)$ is the value of the texture feature in the point p , and θ_2 is a user-defined constant. As the energy is inversely proportional to the distance between the current snaxel and its potential position, it prefers more distant points, enabling faster expansion of the contour. This energy works under two assumptions: (1) features with a low dispersion in the initial snake region have a potential to discriminate it from other patterns and (2) a significant value change in one of the maps discourage the current snaxel from further movement. The method gathers such points in the next stage of the algorithm. It should be noted that a snaxel stoppage caused by the energy is not necessary permanent, as the topology reformulation can deal with outlier points that have a high value of $E_{\text{tex}2}^s$.

3.3 Stage III: Final contour refinement

The third stage of the algorithm results in a final form of the contour. As the borders of texture regions are especially difficult to extract, this stage performs a more advanced fine-tuning of the snake, assuming that the contour after the second stage is close to the actual boundaries of the region and the necessary computations will not affect the overall performance. This stage is also based on minimisation of the contour energy, but it uses a different version of the texture-based condition. Instead of focusing on a single unfitting texture feature, the energy accumulates the dissimilarities of the many features and discourages the snaxel from further movement when a given error threshold is reached. Furthermore, the calculations are performed for a range of the threshold values, which results in multiple final segmentations, from which the best one is finally selected.

The third stage uses a modified version of the texture energy described in Eq. (3). The energy uses the information about the features that were responsible for stopping the contour at its final position in the previous stage. With these data available, a local energy of a snaxel s is based on a new set of features T_{local} that contains only the features that overpassed the threshold [see Eq. (3)] in its local neighbourhood (10×10 pixels window by default). This operation eliminates the features that were selected in the first stage but were not useful in the second. Moreover, every feature map in T_{local} is assigned a weigh w_{t_i} , which is calculated by dividing the number of stoppages for the map in the snaxel neighbourhood by the total number of the stoppages from all feature maps in

the analysed window. This parameter gives more influence to the features that were responsible for more stoppages in the snaxel neighbourhood at the border of the contour.

The texture energy $E_{\text{tex}3}^s$ in this stage is defined as:

$$E_{\text{tex}3}^s(p) = \begin{cases} \frac{1}{d(s,p)+1} & \text{if } \left(\sum_{t_i \in T_{\text{local}}} \frac{|\text{val}_{t_i}(p) - \bar{x}_{t_i}|}{\bar{x}_{t_i}} \right. \\ & \left. \times 100 \times w_{t_i} \right) \leq \theta_3 \\ 1 & \text{otherwise,} \end{cases} \quad (4)$$

where t_i is a texture feature map in the selected set T_{local} , \bar{x}_{t_i} is the feature mean in the initial snake region, $\text{val}_{t_i}(p)$ is the value of the texture feature in the point p , and θ_3 is a user-defined constant. In this version of the energy, the per cent errors between $\text{val}_{t_i}(p)$ and \bar{x}_{t_i} for every feature map in T_{local} are multiplied by the weight w_{t_i} and added up. The energy encourages the expansion of the contour only when the sum of the errors does not exceed a given threshold θ_3 .

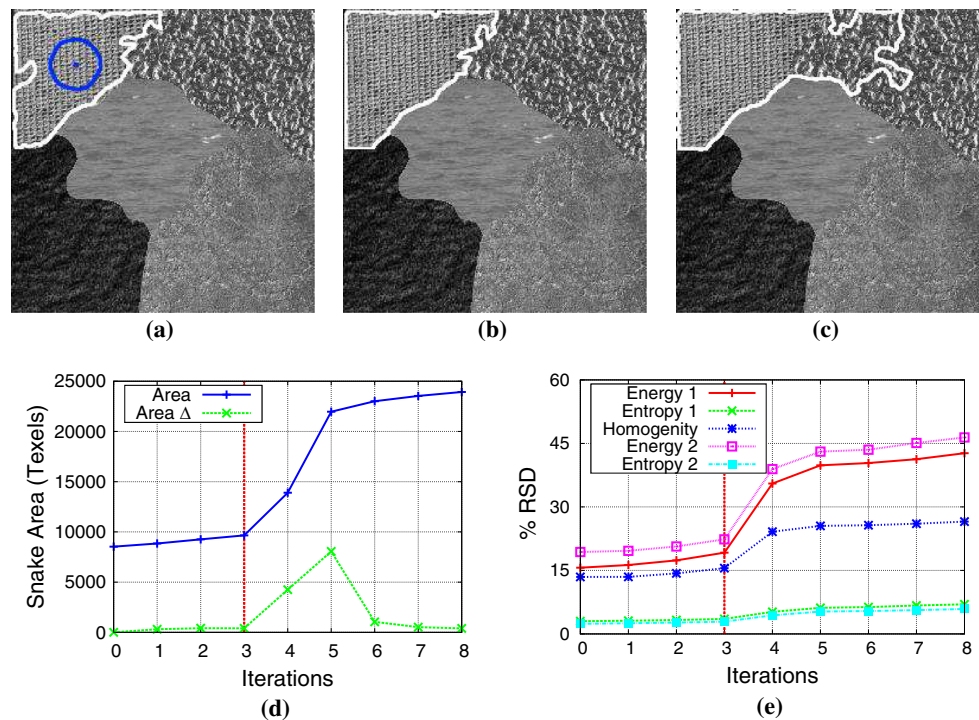
With the final form of the energy defined, the method performs a series of contour evolutions by incrementing the value of θ_3 by 50% in each iteration, creating a set of possible segmentation results (see Fig. 2). The final result can be currently selected in two ways: manually by the operator, or using the algorithm suggestion based on two conditions: (1) tracking of changes of the snake area and (2) analysis of values of the texture features in consecutive iterations. The changes are detected by calculating a slope $m = \frac{|\Delta v|}{\Delta i}$ of a characteristic v in the iteration i . The slope is calculated only between two consecutive iterations, therefore $\Delta i = 1$ and the slope $m = |\Delta v| = |v(i_n) - v(i_{n-1})|$. A slope greater than a specified threshold indicates a sudden change in the characteristics.

The first selection condition tracks the snake area. As the third stage is not expected to cause a big increase in the shape of the snake, a sharp change of the area will suggest a local “leakage” of the contour due to oversegmentation [see Fig. 2(c)]. Similarly, the second condition detects rapid changes in the standard deviations of the image texture features in the area covered by the snake in each iterations [see Fig. 2(e)]. This stage tracks only the features responsible for 80% of the stoppages after the second stage. The final result is suggested by selecting the iteration before the sudden change (in the tracked value) occurred. The final result is obtained by selecting a median value from the iterations suggested by the area and all tested texture features.

4 Method realisation

This section details an example realisation of the proposed approach. Currently, the utilised texture feature set consists of GLCM and Gabor features.

Fig. 2 Example of the Stage III result selection: **a** initial circular contour and result of Stage II, **b** final result from iteration 3, **c** rejected leaked contour from iter. 4, **d** plot of the change in contour area during Stage III and **e** plot of the change in texture features in the consecutive iterations (iteration 3 marked with a vertical line.)



4.1 GLCM features generation

Currently, the method uses five GLCM features: entropy, correlation, homogeneity, contrast and energy. This selection represents only a subset of available GLCM features, but it was deemed to be sufficient for this example implementation. With these five measures, a feature space/set is generated using a combination of specific parameters: window size, displacement and orientation. The calculated feature space should be large enough to be useful for characterisation of various texture patterns in different scales and resolutions. Generation of a large feature space, however, can hinder the overall performance of the method, even with the GPU-accelerated algorithm being used. In order to address this concern, the method start with a fixed set of GLCM properties and, if necessary, performs additional computations.

The generation algorithm uses a few pre-selection conditions to reduce the feature space in the case of isotropic and fine-grained textures. The window size is initially calculated for 3×3 , 5×5 and 7×7 square pixel neighbourhoods. This parameter set can be too limited to capture the properties of larger texture pattern; therefore, an additional step checks the possibility for an expansion of the window. In the calculated homogeneity maps, the initial region of the contour is partitioned into square regions with the size of the two biggest windows (5×5 and 7×7). For a given window size s , this partitioning creates a matrix of regions W_s with a single region denoted as $w_{i,j}$. A set of arithmetic means $q_{i,j}$ of the

feature values inside of each of the region $w_{i,j}$ is calculated, followed by relative standard deviation of the feature means from all the regions, defined as

$$\%RSD(W_s) = \frac{\sigma_q}{\bar{q}} \times 100, \quad (5)$$

where σ_q and \bar{q} are the average and standard deviation of feature means of all regions. A high value of this measure for the tested windows sizes ($\%RSD(W_5)$ and $\%RSD(W_7)$ greater than 20% by default) signalise a high dispersion of the average feature values inside the sub-regions, which can be an indicator of a larger texture pattern. In that case, the method to calculate the GLCM maps for bigger windows (9×9 , 11×11 , and so on) until the dispersion is below the acceptable threshold or the window area exceeds 25% of the initial region.

By default, the algorithm generates a map for the orientations of 0° , 45° , 90° , 135° , and one complete map that combines all these angles. Then, it selects only the directional maps which have their average feature value (inside the initial region) sufficiently different (at least 50% by default) from the average value in the complete map. This process can exclude maps where the orientation angle has no meaningful effect. Furthermore, in case of a clearly isotropic texture pattern, generation of the maps with different orientations can be manually turned off, leaving only the map with all angles.

Fig. 3 Segmentation of synthetic images: **a, c, e** initialisation of the contour with the final result and **(b, d, f)** next result after the selected segmentation

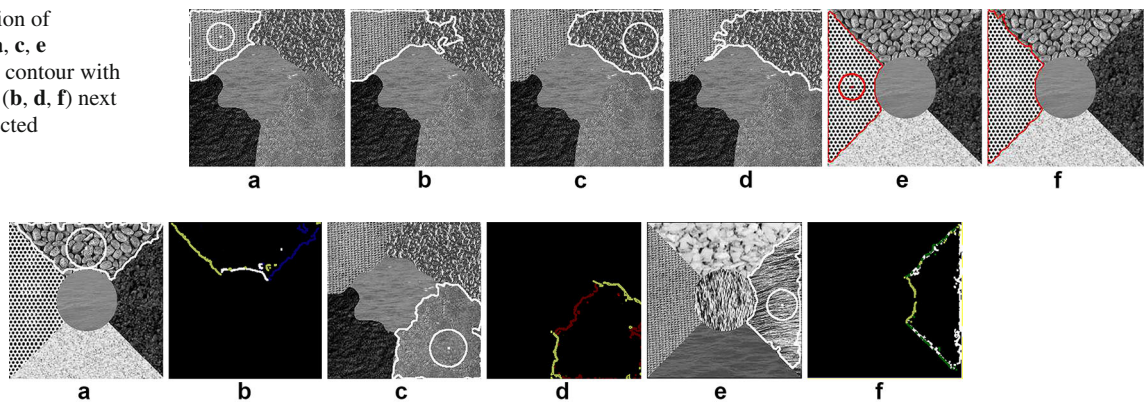
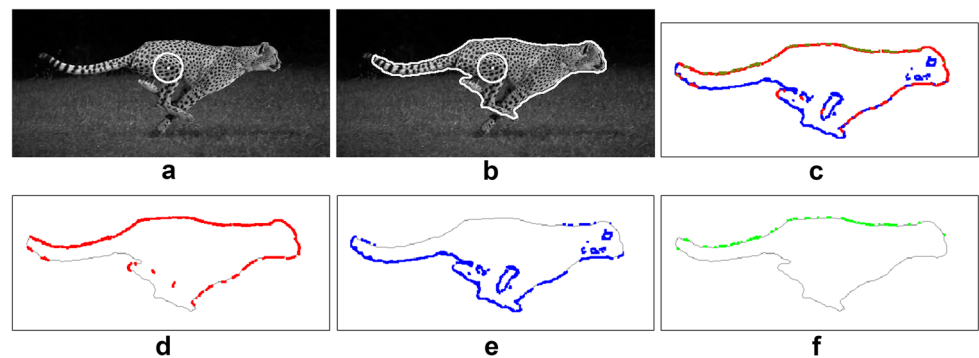


Fig. 4 Segmentation of different texture patterns: **a, c, e** initialisation with the final result and maps showing a visualisation of the texture features that were the most responsible for stopping the contour on the final

border: **b** contrast, energy and correlation, **d** contrast and homogeneity and **f** contrast, entropy and energy

Fig. 5 Natural image segmentation: **a** initialisation; **b** final result; **c** combined map of the most influential GLCM texture features; and **d, e, f** maps of the influence of single features (respectively): entropy, correlation and homogeneity



4.2 Gabor features generation

In addition to the GLCM features, our method also uses a bank of Gabor filters to extend the initial feature set. Gabor filters are linear filters that are supposed to model the behaviour of certain cells in the mammalian visual cortex [5] and have an ability to localise the information in both spatial and frequency domain. This properties make them useful in the image feature extraction process, particularly in the texture analysis field [11, 19]. In our method, we use a bank of 24 filters with four orientations (0° , 45° , 90° and 135°), three wavelengths (2, 4 and 8 pixels) and two Gaussian envelope scales. The filtering performance benefits greatly from the GPU-accelerated convolution algorithm. The generation process also employs an angle detection scheme, similar as in the case of GLCM features.

5 Experimental evaluation

The proposed method was tested on 256×256 synthetic images created using the Brodatz texture database, as well as on a set of natural images. The initial contours were man-

ually placed inside the desired regions and scaled to the preferred size. During the experiments, only the sensitivity parameter θ_2 was modified (between 3 and 5), while the other parameters were left constant on default values ($r = 65\%$, $\theta_3 = 100$ with 50% increase). The first stage of the algorithm selects usually from 20 to 40 feature maps to be used in the next stages. The number of the influential maps in the third stage depends on the pattern characteristics, but for the presented examples was usually lower than 10. The experiments were performed on a machine with AMD FX 8150 Eight-Core processor, 16 GB RAM, Nvidia GeForce GTX 660 graphics card (with 960 CUDA cores). The total segmentation time was between 5 and 10 s for each of the presented examples. The algorithm uses OpenCL and was implemented using the MESA system [21]—a platform for designing and evaluation of the deformable model-based segmentation methods.

The first example (see Fig. 3) presents segmentation results of synthetically composed images. In the first two images, the area condition in the third stage was successful in detection of the proper result—the area threshold was exceeded right after the contour “leaked out” of the segmented region. The result of the second stage was also close to the final form of the contour.

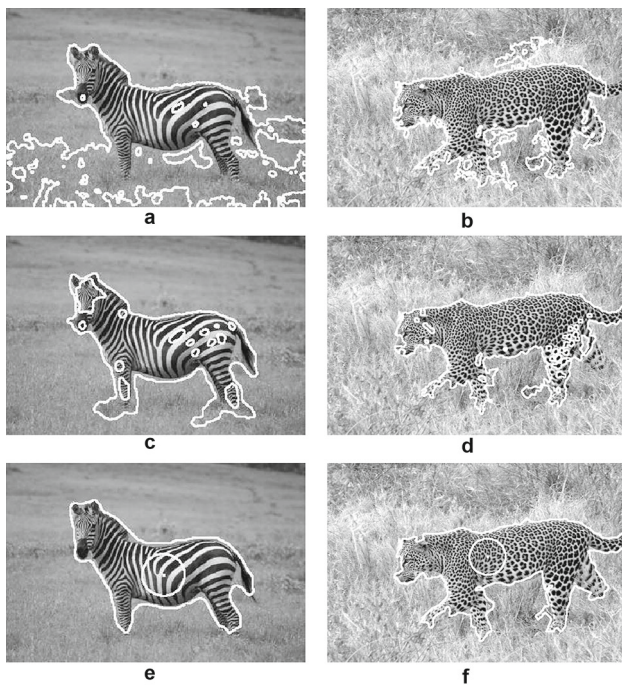


Fig. 6 Natural images segmentation: **a** ACWE with GLCM entropy map, **b** ACWE with GLCM correlation, **(c,d)** ACWE and level set with GLCM entropy and **(e, f)** results of the proposed method

Table 1 Segmentation quality for natural images: results of the ACWE and ILS in comparison with the results from the proposed method

Image	Method	VOE	RVD	ASSD
Leopard	Tex. ACWE	17.2	13.0	6.6
	Tex. ILS	14.1	−9.7	5.5
Zebra	Tex. ACWE	56.4	124.0	204.0
	Tex. ILS	24.8	3.5	8.5

Another example (see Fig. 4) shows the validity of the multi-feature approach. During the segmentation of the regions, different texture features were responsible for stopping the contour on the borders with other regions. The usage of many features allowed the method to adapt itself to a specific local situation, where different features were necessary to discriminate the texture from other patterns.

The last two examples present the results of natural image segmentation. Figure 5 shows a situation where different texture features were needed in different parts of the region. Figure 6 compares the results of the proposed method with ACWE and an intensity-based level set (ILS) [14]. Both methods support only image intensity by default and are not suited for regions with high-contrast patterns. Therefore, the ACWE and ILS methods were run on single texture feature maps selected from the third stage of the proposed methods, effectively simulating a single-feature approach. Although the maps were chosen to contain a possibly uniform fea-

tures, ACWE and ILS running on just the single feature map could not give the desired results. The segmentation quality (see Table 1) was assessed with three error measures: volume overlap error (VOE), relative volume difference (RVD) and average symmetric surface distance (ASSD) (see [8] for full definitions).

6 Conclusions and future works

In this paper, a multi-stage texture-based parametric active contour has been presented. The proposed method improves the segmentation results of traditional snake methods on images with textures of various size, contrast and complexity. Moreover, the algorithm does not require any previous information about the texture classes in the segmented image. The contour uses a relatively large number of GLCM and Gabor features while maintaining a high performance thanks to GPU acceleration. Despite its relative simplicity, the proposed approach gives promising results on artificially composed and natural images.

The present form of the algorithm can benefit from many possible improvements. The general nature of the initial feature generation and selection stage makes the method open to incorporation of different texture features and is not limited to the current GLCM/Gabor approach. This stage can be also extended to employ a more advanced feature selection scheme. Furthermore, the simple discrete form of the contour can be replaced with a more robust level set-based method [14, 17]. The final refinement stage is the part of the algorithm that can benefit from some additional improvements. The final result selection mechanism works well in a case of significant changes in the tracked characteristics, but is not yet suited for detection of less abrupt changes (e.g. on a smooth boundary between textured regions).

We are also investigating a level set-based adaptation of the method into 3D and its possible usage on biomedical images, since modern imaging techniques are inherently 3D.

Acknowledgements This work was supported by Bialystok University of Technology under Grants W/WI/1/2016 and S/WI/2/2013.

References

- Awate, S.P., Tasdizen, T., Whitaker, R.T.: Unsupervised texture segmentation with nonparametric neighborhood statistics. In: Leonardis, A., Bischof, H., Pinz, A. (eds.) *Computer Vision—ECCV 2006*, pp. 494–507. Springer, Berlin (2006)
- Caselles, V., Kimmel, R., Sapiro, G.: Geodesic active contours. *Int. J. Comput. Vis.* **22**(1), 61–79 (1997)
- Chan, T., Vese, L.: Active contours without edges. *IEEE Trans. Image Process.* **10**(2), 266–277 (2001)
- Cohen, L.D.: On active contour models and balloons. *CVGIP Image Underst.* **53**, 211–218 (1991)

5. Daugman, J.G.: Uncertainty relation for resolution in space, spatial frequency, and orientation optimized by two-dimensional visual cortical filters. *J. Opt. Soc. Am. A.* **2**(7), 1160–1169 (1985)
6. Esedoglu, S., Ruuth, S., Tsai, R.: Threshold dynamics for shape reconstruction and disocclusion. In: *Proceeding IEEE International Conference on Image Processing*, vol. **2**, pp. 502–505 (2005)
7. Haralick, R., Shanmugam, K., Dinstein, I.: Textural features for image classification. *IEEE Trans. Syst., Man, Cybern. Syst.* **6**, 610–621 (1973)
8. Heimann, T., van Ginneken, B., Styner, M.A., et al.: Comparison and evaluation of methods for liver segmentation from CT datasets. *IEEE Trans. Med. Imaging* **28**, 1251–1265 (2009)
9. Houhou, N., Thiran, J., Bresson, X.: Fast texture segmentation model based on the shape operator and active contour. In: *Proceeding IEEE Conference on Computer Vision and Pattern Recognition*. pp. 1–8 (2008)
10. Huang, X., Qian, Z., Huang, R., Metaxas, D.: Deformable-model based textured object segmentation. *Energy Minimization Methods in Computer Vision and Pattern Recognition* pp. 119–135 (2005)
11. Jain, A., Farrokhnia, F.: Unsupervised texture segmentation using Gabor filters. *Pattern Recognit.* **24**(12), 1167–1186 (1991)
12. Kass, M., Witkin, A., Terzopoulos, D.: Snakes: Active contour models. *Int. J. Comput. Vis.* **1**(4), 321–331 (1988)
13. Laine, A., Fan, J.: Texture classification by wavelet packet signatures. *IEEE Trans. Pattern Anal. Mach. Intell.* **15**(11), 1186–1191 (1993)
14. Lefohn, A.E., Cates, J.E., Whitaker, R.T.: Interactive, GPU-based level sets for 3D segmentation. In: Ellis, R.E., Peters, T.M., (eds.) *Proceeding Medical Image Computing Computer Assisted Intervention (MICCAI)*, pp. 564–572. Springer (2003)
15. Mcinerney, T., Terzopoulos, D.: T-snakes: Topology adaptive snakes. *Med. Image Anal.* **4**(2), 73–91 (2000)
16. Moore, P., Molloy, D.: A survey of computer-based deformable models. *International Machine Vision and Image Processing Conference* pp. 55–66 (2007)
17. Paragios, N., Deriche, R.: Geodesic active regions and level set methods for supervised texture segmentation. *Int. J. Comput. Vis.* **46**(3), 223–247 (2002)
18. Pujol, O., Radeva, P.: Texture segmentation by statistical deformable models. *Int. J. Image Gr.* **4**(03), 433–452 (2004)
19. Reed, T., DuBuf, J.: A review of recent texture segmentation and feature extraction techniques. *CVGIP Image Underst.* **57**(3), 359–372 (1993)
20. Reska, D., Boldak, C., Kretowski, M.: A texture-based energy for active contour image segmentation. In: *Image Processing Communications Challenges 6, Advances in Intelligent Systems and Computing*, vol. 313, pp. 187–194. Springer International Publishing (2015)
21. Reska, D., Jurczuk, K., Boldak, C., Kretowski, M.: MESA: Complete approach for design and evaluation of segmentation methods using real and simulated tomographic images. *Biocybern. Biomed. Eng.* **34**(3), 146–158 (2014)
22. Reska, D., Kretowski, M.: HIST - an application for segmentation of hepatic images. *Zesz. Naukowe Politech. Bialostoc. Inform.* **7**, 71–93 (2011)
23. Ronfard, R.: Region-based strategies for active contour models. *Int. J. Comput. Vis.* **13**(2), 229–251 (1994)
24. Rousson, M., Brox, T., Deriche, R.: Active unsupervised texture segmentation on a diffusion based feature space. In: *Proceeding IEEE Conference on Computer Vision and Pattern Recognition*. pp. 699–704 (2003)
25. Sagiv, C., Sochen, N., Zeevi, Y.: Integrated active contours for texture segmentation. *IEEE Trans. Image Process.* **15**(6), 1633–1646 (2006)
26. Shen, T., Zhang, S., Huang, J., Huang, X., Metaxas, D.: Integrating shape and texture in 3D deformable models: from Metamorphs to Active Volume Models. In: *Multi Modality State-of-the-Art Medical Image Segmentation and Registration Methodologies*, pp. 1–31. Springer (2011)
27. Singh, P., Garg, R.: Fixed point ica based approach for maximizing the non-gaussianity in remote sensing image classification. *J. Indian Soc. Remote Sens.* **43**(4), 851–858 (2015)
28. Sochen, N., Kimmel, R., Malladi, R.: A general framework for low level vision. *IEEE Trans. Image Process.* **7**(3), 310–318 (1998)
29. Tatu, A., Bansal, S.: A novel active contour model for texture segmentation. In: *Energy Minimization Methods Computer Vision Pattern Recognition*. pp. 223–236. Springer (2015)
30. Wu, Q., Gan, Y., Lin, B., Zhang, Q., Chang, H.: An active contour model based on fused texture features for image segmentation. *Neurocomputing* **151**, 1133–1141 (2015)
31. Xu, C., Prince, J.L.: Snakes, shapes, and gradient vector flow. *IEEE Trans. Image Process.* **7**(3), 359–369 (1998)
32. Yadollahi, M., Procházka, A., Kašparová, M., Vyšata, O.: The use of combined illumination in segmentation of orthodontic bodies. *Signal, Image and Video Process.* **9**(1), 243–250 (2015)