RIDGE AND TREE FEATURE DETECTION ON IMAGES[1]

A. Levashov [2], D. Yurin[3]

[2, 3] Laboratory of Mathematical Methods of Image Processing
Faculty of Computational Mathematics and Cybernetics
Lomonosov Moscow State University
Leninskie Gory, Moscow 119991, Russia
[2]alexeylevashov89@gmail.com , [3]yurin@cs.msu.ru

In this article a new ridge detector with improved non-maxima suppression procedure in scale-space is proposed. Using the results of scale space ridge detection as primary markup the lines are prolonged up to branch points and restored in places where they are not detected. In our method the adaptive choice of threshold and flood fill are used. The algorithm is tested on synthetic, landscape and medical images.

Introduction

There are several major algorithms for ridge detection. Apparently the first of them was the approach [2] that does not use scale-space analysis and therefore its disadvantage is that it detects only thin lines. The approach [5] is based on the scale-space analysis, but it represents the theoretical basis for it and not practically effective algorithm. For the last 15 years there have been only a few cases where it was used in scientific researches. The detector [3] based on generalization of NMS for the case of 3d scale space (x, y, t) can been seen as more effective for ridge detection. However one of this approach`s disadvantages is that it does not detect the branch points on the lines and in the adjacent areas. It interrupts the lines connection and significantly complicates the tree structure analysis on images. The examples of such tasks are masking of tree branches and wires on the non-professional photos for retouching, road and river net finding on aerospace images for the cartographic information updating, selection of blood veins on medical images and processing of 3D images of computer tomography for medical diagnostic. The branch points are essential for such tasks, but often the lines are not detected in the adjacent areas for several reasons. First, there is an ambiguity in choosing the direction for NMS procedure, because the eigenvalues are almost equal. Second, in the adjacent areas two or more ridges lie closely and, as a result, Laplace operator can give a maximal response on bigger scale (two times or more which leads to lines breakdown). Third, the filter response can diminish due to inconsistency between actual image structure and ridge mathematical model, which implies: the solid color line in the range of central lobe of the second derivative of Gaussian function and background with another color on the sidelobes. This article is devoted to suppression of disadvantages described above.

The ridge detection

In this section we shortly recall the algorithm [3], let $I(x,y)$ is the intensity of input gray-scale image. Scale space [6] is defined as convolution of the image $I(x,y)$ with Gaussian kernel:

$$L(x,y,t) = G(x,y,t) * I(x,y),$$
$$G(x,y,t) = \frac{1}{2\pi t} e^{-\frac{x^2+y^2}{2t}}, \quad t = \sigma^2, \tag{1}$$

We denote smoothed image $L(x,y,t)$ derivatives with subscripts as $L_x, L_y, L_{xx}, L_{xy}, L_{yy}, \ldots$. Such derivatives are equivalent to original image $I(x,y)$ convolution with appropriate derivatives of Gauss function $G(x,y,t)$. By direct differentiation it is easy to check [6] that scale-space representation of image $L(x,y,t)$ is satisfied to diffusion equation:

$$L_t = \frac{1}{2}\left(L_{xx} + L_{yy}\right), \tag{2}$$

where derivatives are computed at scale t.

[1] **The research was supported by the RFBR grant 13-07-00584.**

Consider the algorithm [3] for ridge detection with branch points missing. Let the thickness of lines is fixed and equal 2σ, then filter for ridge lines finding can be the Laplace operator or maximal eigenvalue of Hessian $\mathbf{H}_{22}$ in point (x,y)

$$\mathbf{H}_{22} = \begin{pmatrix} L_{xx} & L_{xy} \\ L_{xy} & L_{yy} \end{pmatrix}, \quad (3)$$

where the derivatives are computed on scale $t = \sigma^2$. In each point (x,y) we have a pair of eigenvectors: $(\bar{\mathbf{v}}_1, \bar{\mathbf{v}}_2)$, where $\bar{\mathbf{v}}_1$ corresponds to eigenvalue λ_1 such that $|\lambda_1| \geq |\lambda_2|$ and $\bar{\mathbf{v}}_1$ is normal to ridge. On this basis we obtain the algorithm of ridge lines detection as local maxima finding task (we find directional local maxima and minima of values λ_1 in direction $\bar{\mathbf{v}}_1$), i.e. we use the Non Maxima Supression (NMS) procedure with 3x3 window from Canny edge detector. If the detection scale t does not equal to thickness of line, the response of Laplacian (which is equal to $\Delta L = L_{xx} + L_{yy} = \lambda_1 + \lambda_2$) or eigen value $|\lambda_1|$ is less than that of for scale is equal to thickness. Then if we analyze the response on neighborhoods scales we can choose the optimal scale. In scale-space the 3D Hessian is:

$$\mathbf{H}_{33} = \begin{pmatrix} L_{xx} & L_{xy} & L_{xt} \\ L_{xy} & L_{yy} & L_{yt} \\ L_{xt} & L_{yt} & L_{tt} \end{pmatrix}, \text{ where using (2)}$$

$$L_{xt} = \frac{1}{2}\left(L_{xxx} + L_{xyy}\right), \; L_{yt} = \frac{1}{2}\left(L_{xxy} + L_{yyy}\right), \quad (4)$$

$$L_{tt} = \frac{1}{4}\left(L_{xxxx} + 2L_{xxyy} + L_{yyyy}\right).$$

In [3] the NMS procedure was modified as follows: the suppression is performed simultaneously in 2 directions of 3D scale space (x,y,t) using 3x3x3 window. The directions are selected as projections of 2 eigenvectors corresponding to 2 maximal eigenvalues of (4) on the plane normal to eigenvector $\bar{\mathbf{v}}_2$ of 2D Hessian (3). The discrete step on scales $\sigma_0, ..., \sigma_i, ..., \sigma_n$ in [3] are chosen as in [7]: $\sigma_0 = 1, \sigma_{i+1} = \sigma_i \sqrt[4]{2}$. For 3D NMS procedure implementation it is sufficient to hold in memory only 3 layers corresponding to 3 sequential scales. Processing such layers triplets sequentially under σ_i increasing, the local maximum points are collected.

False detection suppression

The method [3] outlined above detects also such features as blobs and some edges. In our approach the final results are presented in vector form instead of bitmap. After detection of ridges their spine pixel chains are collected in lists and the lists are united in a graph. For excluding blobs, the lists with small count of pixels are discarded. It is important that the minimal count of pixels in list is proportional to the ridge (or thick line) width. That is we impose a constraint for minimum elongation of the blobs, which are considered as ridges. This approach is differ from [3] where separation of ridges and blobs is based on ratio of eigen values of (3).

For suppression of edges in [3] in NMS procedure the points are discarded if next dot product is negative:

$$\left(\vec{\mathbf{g}}\left(\vec{\mathbf{x}} + \vec{\mathbf{v}}_1\right), \vec{\mathbf{g}}\left(\vec{\mathbf{x}} - \vec{\mathbf{v}}_1\right)\right) < 0. \quad (5)$$

Our experiments show that condition (5) can lead to erroneous discarding true ridge centers for example in cases when different side's backgrounds relatively to ridge have a different average intensity. We propose the modification of this condition:

$$\left(\vec{\mathbf{g}}\left(\vec{\mathbf{x}} + \sigma_i \vec{\mathbf{v}}_1\right), \vec{\mathbf{g}}\left(\vec{\mathbf{x}} - \sigma_i \vec{\mathbf{v}}_1\right)\right) < T_g \quad (6)$$

where σ_i corresponds to current scale, the threshold constant satisfied the inequality:

$$-1 < T_g < 0, \quad (7)$$

$\vec{\mathbf{g}} = (L_x, L_y)^T$ is the image intensity gradient.

Adaptive flood-fill

After the previous step we obtains the set of detected ridges represented as set of points $S = \left\{\left(\bar{\mathbf{x}}_j, \sigma_j, l_j\right)\right\}$, where $\bar{\mathbf{x}}_j$ – the coordinates of center points of ridge lines, σ_j – the scale, at which we detect ridge point with coordinates $\bar{\mathbf{x}}_j$, l_j – the response value in this point (the response of Laplacian convolution computed on scale σ_j). Using this data, we build 3 images $M, I_{\min}, I_{\max}$. We set $M(\bar{\mathbf{x}}) = 1$ if pixel

belongs to ridges, and $M(\bar{\mathbf{x}})=0$ elsewhere. $I_{\min}, I_{\max}$ – additional images showing the available ridges points intensity range. Let initially in all points $M(\bar{\mathbf{x}})=0$. Then for each points $\bar{\mathbf{x}}_j \in S$ we set

$$M(\bar{\mathbf{x}}_j)=1, \quad I_{\max}(\bar{\mathbf{x}}_j)=I(\bar{\mathbf{x}}_j)+0.5|l_j|, \quad I_{\min}(\bar{\mathbf{x}}_j)=I(\bar{\mathbf{x}}_j)-0.5|l_j|. \tag{8}$$

As shown in [3], the Laplacian value on scale corresponding to ridge thickness is equal to difference of average intensities of background and of ridge.

Then we make flood fill of image M. The proposed algorithm is similar to [1], but we use additional conditions for pixel filling. Consider two neighborhood pixels $\bar{\mathbf{y}}_k$ and $\bar{\mathbf{y}}_j$ - $M(\bar{\mathbf{y}}_k)=1$, $M(\bar{\mathbf{y}}_j)=0$. If the condition

$$I_{\min}(\bar{\mathbf{y}}_k)<I(\bar{\mathbf{y}}_j)<I_{\max}(\bar{\mathbf{y}}_k), \tag{9}$$

is satisfied, we modify the pixel $\bar{\mathbf{y}}_j$ as:

$$M(\bar{\mathbf{y}}_j)=1, \quad I_{\min}(\bar{\mathbf{y}}_j)=I_{\min}(\bar{\mathbf{y}}_k), \quad I_{\max}(\bar{\mathbf{y}}_j)=I_{\max}(\bar{\mathbf{y}}_k). \tag{10}$$

Now we can describe the algorithm of adaptive flood fill:

Алгоритм **AdaptiveFloodFill**$(\bar{\mathbf{x}}, \bar{\mathbf{d}})$

1. **if** $M(\bar{\mathbf{x}})=1$ **return**
2. **if** $NOT\left(I_{\min}(\bar{\mathbf{x}})<I(\bar{\mathbf{x}}+\bar{\mathbf{d}})<I_{\max}(\bar{\mathbf{x}})\right)$ **return**
3. $M(\bar{\mathbf{x}}+\bar{\mathbf{d}})=1$
   $I_{\min}(\bar{\mathbf{x}}+\bar{\mathbf{d}})=I_{\min}(\bar{\mathbf{x}})$
   $I_{\max}(\bar{\mathbf{x}}+\bar{\mathbf{d}})=I_{\max}(\bar{\mathbf{x}})$
4. **AdaptiveFloodFill** $(\bar{\mathbf{x}}+\bar{\mathbf{d}}, (0 \quad 1)^T)$
5. **AdaptiveFloodFill** $(\bar{\mathbf{x}}+\bar{\mathbf{d}}, (1 \quad 0)^T)$
6. **AdaptiveFloodFill** $(\bar{\mathbf{x}}+\bar{\mathbf{d}}, (0 \quad -1)^T)$
7. **AdaptiveFloodFill** $(\bar{\mathbf{x}}+\bar{\mathbf{d}}, (-1 \quad 0)^T)$

In the beginning of the algorithm for each points $\bar{\mathbf{x}}_i : M(\bar{\mathbf{x}}_i)\neq 0$ and for each neighborhoods $\bar{\mathbf{x}}_i+\bar{\mathbf{d}}$ we invoke the algorithm **AdaptiveFloodFill**$(\bar{\mathbf{x}}_i, \bar{\mathbf{d}})$, which differs from [1] by 2 and 3 lines. For near border pixels we additionally check when coordinates $\bar{\mathbf{x}}+\bar{\mathbf{d}}$ are outside of image rectangle. This is the scheme of simple and not optimal realization of the flood filling algorithm. The faster algorithm is described, for example, in [1], and can easy be modified for our problem by including additional conditions for pixel filling (9), (10).

Ridge skeletization

The binary image M after the previous step will segment the image into the pixels belonging to the ridges and the background pixels. When at branch points the ridges change their color slightly in comparison with the background, the junction points will be added to the ridges, even if they were not found by the scale-space detector. Further the continuous skeletization of binary image can be applied and the skeleton can be used as the result. We use the algorithm [9], because in our approach it is desirable to retain the ridge points found by scale-space detector and only add junction and junction neighborhood ridge points as a result of thinning. The algorithm of skeletization via line thinning is based on step-by-step removing of the points which are not the centers in lines under additional condition that it is not ridge point obtained with scale-space detector. Each algorithm iteration includes two passes through all the pixels. On each pass the non-skeleton points are marked but removed only after the end of the pass. Around each point $\bar{\mathbf{x}}$ of the image M the window 3x3 is selected. We can enumerate pixels $P_1, P_2, \ldots, P_9$ is the window in order shown on the Fig.1. Descriptors $B(\bar{\mathbf{x}})=\sum_{i=2}^{9} P_i$ and $A(\bar{\mathbf{x}})$ which is equal to the number of patterns 01 in the sequence $P_2, \ldots, P_9, P_2$ are calculated.

P_9 $(i-1, j-1)$	P_2 $(i-1, j)$	P_3 $(i-1, j+1)$
P_8 $(i, j-1)$	P_1 (i, j)	P_4 $(i, j+1)$
P_7 $(i+1, j-1)$	P_6 $(i+1, j)$	P_5 $(i+1, j+1)$

Figure 1: pixel`s order in 3x3 window

On the first pass the pixel $\bar{\mathbf{x}}$ is marked to be removed from image M if all four next conditions are satisfied:

a) $2 \le B(\bar{\mathbf{x}}) \le 6$

b) $A(\bar{\mathbf{x}})=1$

c) $M(P_2)\cdot M(P_4)\cdot M(P_6)=1$

d) $M(P_4)\cdot M(P_6)\cdot M(P_8)=1$

On the second pass clauses c) and d) are changed to the following:

c') $M(P_2)\cdot M(P_4)\cdot M(P_8)=1$

d') $M(P_2)\cdot M(P_6)\cdot M(P_8)=1$

The iterations are being done until no pixels are removed during two passes. As we don't remove pixels detected by the scale-space algorithm, all the lines found initially retains and only the junction points between them are appended.

Results and conclusion

The ridge detection algorithm without brakes near branch points is proposed. We make some improvements in algorithm [3] and restore breaks in ridges near branch points with adaptive flood fill algorithm. We test our approach on synthetic and medical images. Some results for medical retina images are shown on Fig. 2. The ridges in the Fig. 2b have brakes near branch points while in the Fig. 2c most of these brakes are restored and shown in red. One of the disadvantages of the proposed algorithm is that our method does not detect the accurate ridge edges. But after we restore the ridges interconnections, the problem simplifies to finding edges between object (ridges united in tree) and the background. The task becomes bipolar labeling problem and can be solved via graph minimal cut between source (the set of connected ridge central points) and sink (background, or the points lying far from ridges longer than the ridge width).

References

[1] S.V. Burtsev, Y.P. Kuzmin. "An efficient flood-filling algorithm" // Computers & graphics, Elsevier, -V. 17, -No 5, -P. 549–561, 1993

[2] R. Haralick, "Ridges and Valleys on Digital Images" // Computer Vision, Graphics, and Image Proc., -V. 22, -No. 10, 1983

[3] N. A. Khanina, E. V. Semeikina, D. V. Yurin. "Scale-space color blob and ridge detection" // Pattern Recognition and Image Analysis, -No. 1, -P. 221−227, 2012

[4] M. Klaiber, L. Rockstroh, Z. Wang, Y. Baroud, S. Simon. "A Memory-Efficient Parallel Single Pass Architecture for Connected Component Labeling of Streamed Images" // Field-Programmable Technology, Int. Conf, -P. 159−165, 2012

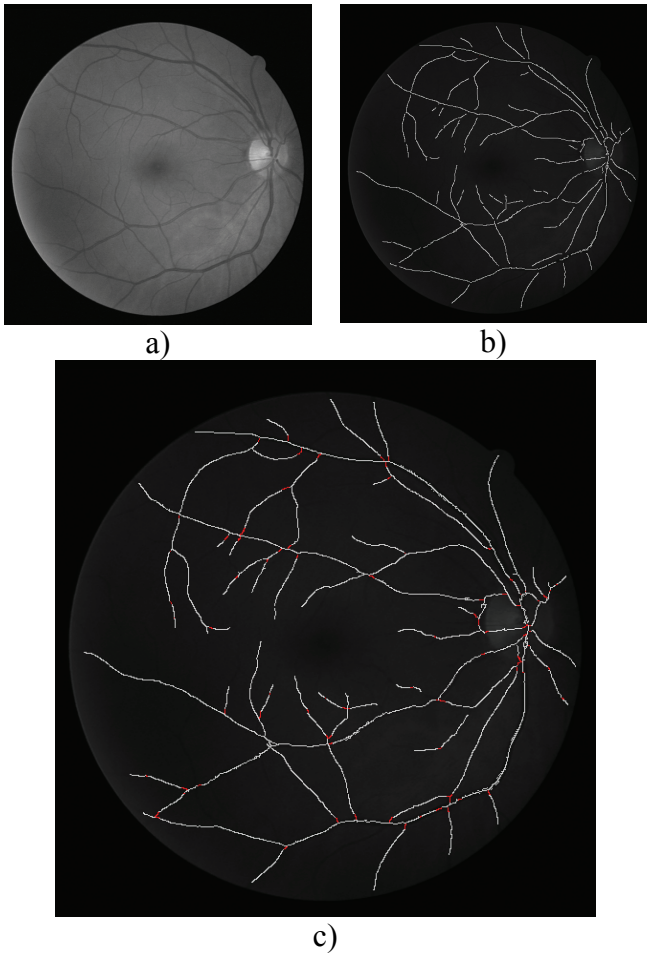
[5] T. Lindeberg, "Edge detection and ridge detection with automatic scale selection" // International Journal of Computer Vision, -V. 30, -No. 2, -P. 117–154, 1998

[6] T. Lindeberg "Scale-Space Theory in Computer Vision" // Kluwer Academic Publishers/Springer, Dordrecht, Netherlands, 1994.

[7] D.G. Lowe. "Distinctive image features from scale-invariant keypoints" // Int. Journal of Computer Vision, -V. 60, -No. 2, -P. 91−110, 2004

[8] M.B. Dillencourt, H. Samet, M. Tamminen (1992). "A general approach to connected-component labeling for arbitrary image representations" // Journal of the ACM, -V. 39, -N. 2, -P. 253−280, 1992

[9] T.Y. Zhang, C.Y. Suen "A fast parallel algorithm for thinning digital patterns" // Communications of the ACM, -V. 27, -No. 3, -P. 236−239, 1984.



a) b)

c)

Figure 2: a) Input image, b) scale-space ridge detection, c) the result of line connection