Linear Feature Extraction from Scanned Color Maps with Complex Background

Yun Yang, Xiao-ya An

State Key Laboratory of Geo-Information Engineering, Xi'an, 710054, China
Xi'an Research Institute of Surveying and Mapping, Xi'an, 710054, China

Abstract. Map digitization is one of the most important means of geographic data acquisition. Automatic extraction of geographic features from scanned maps is the key technique to improve the efficiency of map digitization. With regard to linear feature extraction from color topographic map images, it is usually performed by interactive or automatic line tracking based on color separated binary images. But for those color maps with heavy interconnectedness of geographic features and complex background such as vegetation tints and relief shadings, the results of color image segmentation cannot meet the demand of automatic extraction, and time-consuming manual tracking is inevitable. This paper presents a new semi-automatic method to extract linear features directly from original scanned color maps without color layer separation. In the proposed method, a sliding window is added on a user-specified linear feature, and the current line in the window is first segmented adaptively by using color space conversion, k-means clustering and directional region growing. After that, a thinning operation is performed and the line in the window is tracked from the centre to the edge. By moving the window continuously along the line, iterative operations of image segmentation, thinning and line tracking are accomplished, thus the specified line is tracked automatically. Meanwhile, a little manual processing is incorporated in case automatic tracking fails. The performance of the proposed method is tested on a number of color map samples with complex background.

Keywords: Map digitization, Color topographic map, Linear feature, Extraction, Image segmentation, Line tracking

1. Introduction

Map digitization is one of the most important means of geographic data acquisition for Geographic Information System (GIS) applications. In the past decades, map digitization has undergone the stage of manual tracking on digitizing tablets and the stage of heads-up screen digitizing. Today, human-machine cooperative map digitization based on image processing and pattern recognition techniques has been more and more utilized.

Many commercial software systems such as VPStudio (http://www.softelec.

com/), RxAutoImage (http://www.rasterex.com/), R2V (http://www.ables

w.com/) and MapGIS (http://www.mapgis.com.cn/) are available for map digitization. Linear features in high quality black-and-white map images can be extracted automatically by using line tracking methods. For color topographic maps with heavy interconnectedness of geographic features, however, automatic extraction is still a very difficult challenge.

The existing methods of linear feature extraction from color topographic maps can be grouped into two main categories: color segmentation based method and original map based method. In the color segmentation based method, a color map image is firstly separated into several layers with predefined colors, and then linear features are extracted from separated binary layers interactively or automatically. This kind of method can significantly reduce the complexity of cartographic features, and make the extraction work relatively easier. Here, color image segmentation is a fundamental and critical step. The accuracy and speed of linear feature extraction is dependent on the quality of the segmented results. Many algorithms for color map image segmentation have been developed (Khotanzad & Zink 2003, Ablameyko *et al.* 2003, Pezeshk & Tutwiler 2008). Nevertheless, these algorithms are far from being satisfactory to segment a color map image into desired color layers due to the problem of mixed color pixels and aliasing induced by the scanning processes, especially for the maps with complex background such as vegetation tints and relief shadings. Noises, gaps and adhesions exist everywhere in the separated layers, particularly in the contour line layer. Although some algorithms for removing noises and reconnecting the broken lines have been developed (Cheng *et al.* 2003, Zeng *et al.* 2004), a great deal of human editing is inevitable to acquire high quality image for automatic extraction.

Another way to extract linear features is based on original scanned color maps. A linear feature is tracked automatically starting from a user-specified point, and some kinds of flexible user interventions are allowed in case where automatic tracking fails. The strategy of human-machine coope-

ration is utilized here, which makes the line tracking process under human control, and provides the ability to correct data immediately if required. Therefore, it is more practical and should be a preferred one for the color map images with poor quality and complex background.

To the best of our knowledge, few research works have been done to extract linear features directly from original scanned color maps. The rule of minimum color distance is adopted in the methods proposed by Wu & Wang (1998) and Huang *et al.* (2005) to determine the next point in the line tracking process. From the point of view of applications, such methods have two main disadvantages in performance:

(1) The line tracking process depends greatly on the user-specified starting point. The user needs to magnify the image and select the midpoint of a line exactly as the starting point. If the selected point has a little deviation, the following line tracking cannot be ensured to be along the centerline.

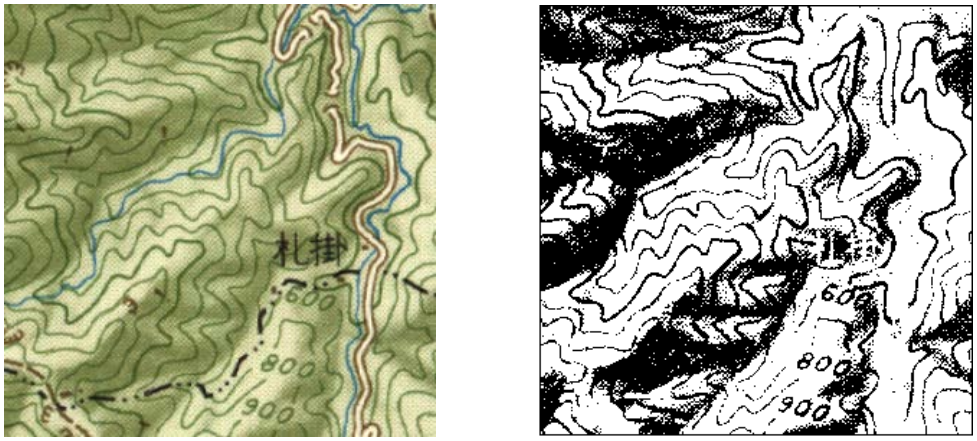
(2) Automatic line tracking often fails when meeting other cartographic features with similar color. Linear features can hardly be tracked on those maps with vegetation tints and relief shadings.

In order to overcome the above shortcomings, this paper presents a new method to extract linear features directly from color topographic map images. It is implemented by using adaptive image segmentation and sequential line tracking based on sliding window.

2. The Analysis of Color Topographic Map Images

Topographic maps typically use only a few distinct colors to represent different cartographic feature layers, for example, black cultural features, brown geomorphologic features, blue drainage features, and green vegetation features. Influenced by the degradation of paper map, the interconnectedness of cartographic features, and the RGB misalignment in the scanning process, large numbers of scattered colors and noises are generated in a scanned map image. This lead to the phenomenon that features in the same layer do not have same color, and similar colors do not belong to same feature layer, therefore introduce great difficulty for image segmentation based on color information.

Figure 1 shows a part of a color topographic map with relief shadings and the color segmentation result of contour line layer. We can see that shading areas adhere together and many broken lines occur in the segmented layer. Automatic linear feature extraction can not be performed at all on such low quality image.



(a) (b)

Figure 1. A part of a color topographic map with relief shadings. (a) Original image. (b) Color segmentation result of contour line layer.

3. The proposed approach

In order to extract linear features directly from original color map images without color segmentation, it is necessary to distinguish linear features adaptively from complex background. In a color topographic map image, different regions usually show marked differences in color, brightness and contrast, especially those regions with vegetation tints and relief shadings. So it is difficult to distinguish linear features from the background using a global method. Considering this, we propose a local adaptive segmentation method based on sliding window to separate a specified linear feature from background, followed by a sequential line tracking to extract the line.

Figure 2 shows the procedure of linear feature extraction. For a specified linear feature, a starting point and initial direction are first input by the operator, and a predefined rectangle window (which is called sliding window) is added on the line. Then, adaptive image segmentation, thinning, and line tracking are performed in the window. By moving the window continuously along the line and doing the above operations iteratively, the line is tracked sequentially until an endpoint or an intersection is met. If the tracking is broken or a tracking error occurs, manual operation is necessary to cross the intersection or move back to the correct point. After that, the sequential line tracking continues until the whole line is extracted.

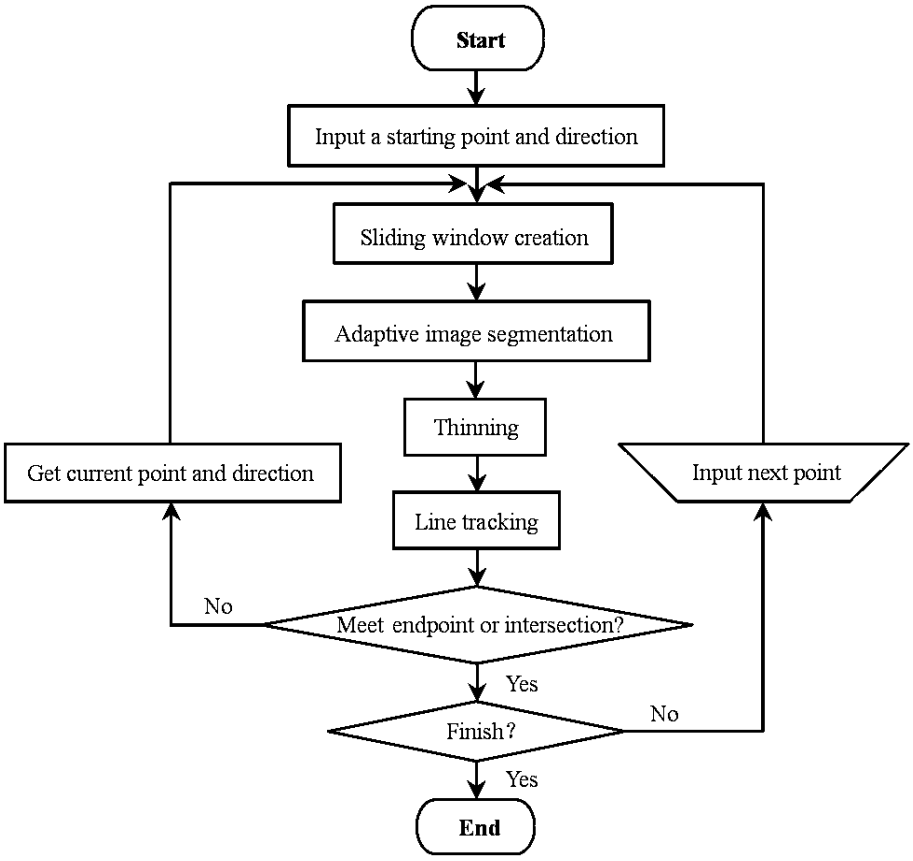


Figure 2. The procedure of linear feature extraction from scanned color maps.

3.1. Adaptive image segmentation based on sliding window

The proposed image segmentation approach based on sliding window is performed by using color space conversion, k-means clustering and directional region growing.

Color space conversion

The color image in the sliding window is first converted into a 256 grey-scale image so as to reduce the complexity of the problem. This is due to the following considerations: Firstly, color confusion can be improved in the image with limited grey-scale. Secondly, it is relatively easy to separate objects from the grey-scale image because there is a marked contrast between foreground and background.

YIQ color space is adopted here to separate grey-scale information from color data. In this color space, image data consists of three components:

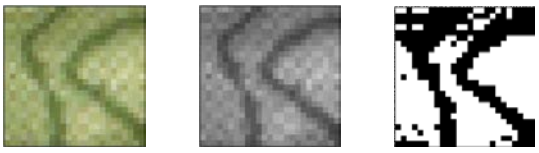
luminance (Y), hue (I), and saturation (Q). The first component represents grey-scale information, while the last two components represent color information. By converting an RGB image into YIQ format, the grey-scale information can be extracted without loss (Chen et al. 2000).

The conversion from RGB to YIQ (Ford & Roberts 1998) can be expressed as:

$$\begin{bmatrix} Y \\ I \\ Q \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ 0.596 & -0.274 & -0.322 \\ 0.211 & -0.523 & 0.312 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} \quad (1)$$

where Y component is the grey-scale value converted from RGB value.

Figure 3(a) and *3(b)* shows the color image and the converted grey-scale image in a 25×25 sliding window of *Figure 1(a)*.



(a) (b) (c)

Figure 3. The image in a 25×25 sliding window. (a) Color image. (b) Gray-scale image. (c) Segmentation by thresholding.

In the image with low contrast and low signal-to-noise ratio (SNR), it is difficult to separate the object from background by using traditional image segmentation methods such as thresholding method (see *Figure 3(c)*). Next, a new algorithm combining k-means clustering with directional region growing is presented to segment the specified linear feature in the sliding window.

K-means clustering

K-means clustering is one of the simplest unsupervised learning algorithms for solving clustering problem (Wagstaff et al., 2001). In our algorithm, it is applied to a small neighbourhood in the centre of the sliding window to divide the pixels into object and background regions. For a 300dpi topographic map image, a 5×5 neighbourhood is selected considering the line width and the interval between two lines.

The process of k-means clustering is as follows.

Step 1: Choose a seed point with minimum grey-scale from the 5×5 neighbourhood of the centre of the sliding window.

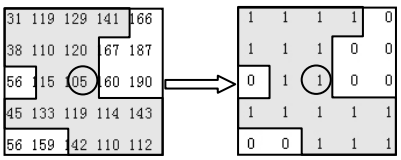
Step 2: Create the initial clustering centres of the object and background region c_1 and c_2 respectively by finding the maximum and minimum grey-scale values in the 5×5 neighbourhood of the seed pixel.

Step 3: Compute $d_1 = |g_{ij} - c_1|$ and $d_2 = |g_{ij} - c_2|$ for each pixel with grey-scale g_{ij} in the 5×5 neighbourhood. If $d_1 < d_2$, then the pixel belongs to the object region and the background otherwise.

Step 4: Compute the average grey-scales of the object and background region m_1 and m_2 respectively. If they converge to c_1 and c_2, go to Step 5; otherwise, assign m_1 and m_2 to the clustering centres c_1 and c_2, go to Step 3.

Step 5: Set all the grey-scale values of the pixels belonging to the target region within the 5×5 neighbourhood to be 1 and 0 otherwise.

For the grey-scale matrix of *Figure 3(b)*, k-means clustering is performed in the 5×5 neighbourhood of the seed pixel (Its grey-scale value is 105). *Figure 4* shows the object and background region after k-means clustering.



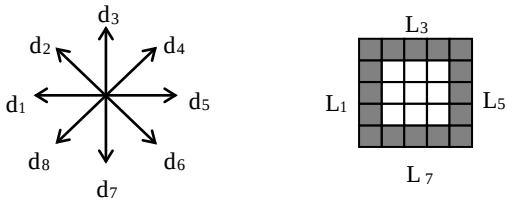
(a) (b)

Figure 4. K-means clustering within a 5×5 neighbourhood. (a) Grey-scale matrix. (b) Clustering result.

It should be noted that the small neighbourhood of k-means clustering can be adjusted with the change of line width and the resolution of map image.

Directional region growing

Based on the k-means clustering result, the object region is expanded to the whole sliding window by using the proposed directional region growing algorithm. Before giving the algorithm, the initial direction given by the operator is transformed into eight discrete directions d_1, d_2, ..., d_8 (see *Figure 5(a)*), and the four sides of a 5×5 neighbourhood centred at a seed pixel are defined as L_1, L_3, L_5 and L_7 (see *Figure 5(b)*).



(a) (b)

Figure 5. (a) 8 discrete directions. (b) The four sides of a 5×5 neighbourhood.

The directional region growing is described as follows.

Step 1: Initialization.

① Find the pixel with minimum original grey-scale value along L_i (i=1,3,5,7) within the target region as a new seed if the initial direction is d_i (i=1,3,5,7).

② Find the pixel with minimum original grey-scale value along L_{i-1} and L_{i+1} (i=2,4,6,8; $L_{8+1}=L_1$) within the target region as a new seed if the initial direction is d_i (i=2,4,6,8).

Step 2: Perform k-means clustering within the 5×5 neighbourhood of the seed, obtaining new clustering centre c_1 and c_2.

Step 3: Find the pixels in the 5×5 neighbourhood meeting the following two conditions. Add them into the object region, and set their grey-scale values to be 1.

Condition 1: The grey-scale difference between the pixel and the object clustering centre is smaller than that between the pixel and background clustering centre, i.e., $\left|g_{ij}-c_1\right|<\left|g_{ij}-c_2\right|$.

Condition 2: There is at least one pixel with binary value 1 in the 8-neighborhood.

Step 4: Find the pixel with minimum original grey-scale value in those pixels belonging to the newly grown target region at the four sides of the 5×5 neighbourhood as a new seed.

Step 5: Repeat *Step 2*-*Step 4* until the edge of the sliding window is reached.

Step 6: Set the grey-scale values of pixels belonging to the background region to be 0.

After the above steps, the segmentation result in the sliding window is obtained (see *Figure 6*).

Figure 6. K-means clustering and directional region growing in the sliding window.

In the algorithm, k-means clustering and directional region growing are performed automatically in the sliding window to separate the object from background no matter how the brightness and contrast change, therefore it is an adaptive segmentation algorithm.

3.2. Sequential line tracking

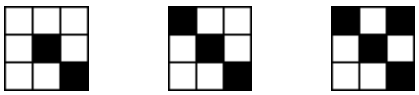
After image segmentation, a thinning operation is performed, followed by a line tracking process in the sliding window. By moving the window continuously along the line and doing the above operations iteratively, the line is tracked sequentially until an endpoint or an intersection is met.

Before giving the algorithm, we first introduce several terms in binary thinned images (see *Figure 7*).

Endpoint: The point with one black pixel in the 3×3 neighbourhood.

Connecting point: The point with two black pixels in the 3×3 neighbourhood.

Crossing point: The point with more than three black pixels in the 3×3 neighbourhood.



(a) (b) (c)

Figure 7. (a) Endpoint. (b) Connecting point. (c) Crossing point.

The sequential line tracking algorithm is described as follows.

Step 1: Find a connecting point P_0 as the starting point in a 3×3 neighbourhood in the centre area of the sliding window.

Step 2: Select a point from the two black pixels in the 3×3 neighbourhood of P_0 with a close direction d to the current direction (the initial direction is input by a human operator).

Step 3: Set the current tracking point to be white, and track to the next black point P_i in direction d (i is counted from 1).

Step 4: Distinguish P_i in the 3×3 neighbourhood. There are three cases as follows.

① If P_i is a connecting point, there remains only one black pixel in its 3×3 neighbourhood except the tracked point, use its direction as tracking direction d, go to *Step 3*.

② If P_i is a crossing point, judge that it is a true or pseudo crossing point (more is said about this in the following). If it is true, stop tracking; otherwise, judge the forward direction d, go to *Step 3*.

③ If P_i is an endpoint or a side point of the sliding window, go to *Step 5*.

Step 5: Count the number of the tracking points. If it is less than 3, stop tracking; otherwise, move the centre of the sliding window to the current point, and go to *Step 1*.

In the above process, crossing points may be encountered due to intersections between different linear features, or due to noise. The former is called true crossing point while the latter is called pseudo crossing point. When a true crossing point is met, automatic line tracking stops for a moment, and the next point is input manually. After that, the line tracking continues as before. When a pseudo crossing point is met, an automatic judgment of forward direction is needed to across the pseudo crossing point.

To handle the pseudo crossing point, a concentration degree $\sum f_{ij}$ is defined, which means the total number of black pixels within a limited region. Suppose that S is the point in the un-thinned binary image corresponding to the crossing point P. If the concentration degree around S is larger than a

preset value, then P is treated as a true crossing point. Otherwise it is a pseudo one. For topographic maps with resolution of 300 dpi, the concentration degree can be set as 20 within a 5×5 region. When meeting a pseudo crossing point, a trial-tracking is done to determine the next tracking direction. As shown in *Figure 8*, there are two forward directions d1 and d2 at the pseudo crossing point P. First, three connecting pixels along d1 and d2 are tracked respectively, and each of their corresponding grey-scale values in the grey-scale image are recorded. Then, the average grey-scale values are calculated for d1 and d2, respectively. If the former is smaller, d1 is determined as the next tracking direction, else d2.

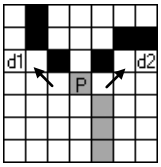
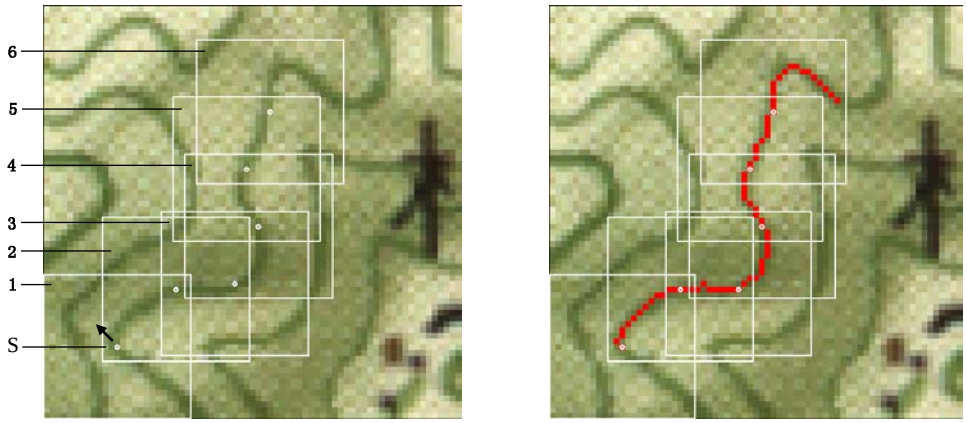


Figure 8. Determination of the forward direction at a pseudo crossing point.

Figure 9 illustrates the continuous sliding window and sequential line tracking from a starting point S with an initial direction pointed by the arrow. *Figure 10* shows the results of image segmentation, thinning and line tracking in window 1-6 in *Figure 9(a)*. The grey line marked in each window draws a tracking path. By connecting all the grey lines in order, the tracking result is obtained (see *Figure 9(b)*).



(a) (b)

Figure 9. Sequential line tracking. (a) Continuous sliding window. (b) The result of line tracking.

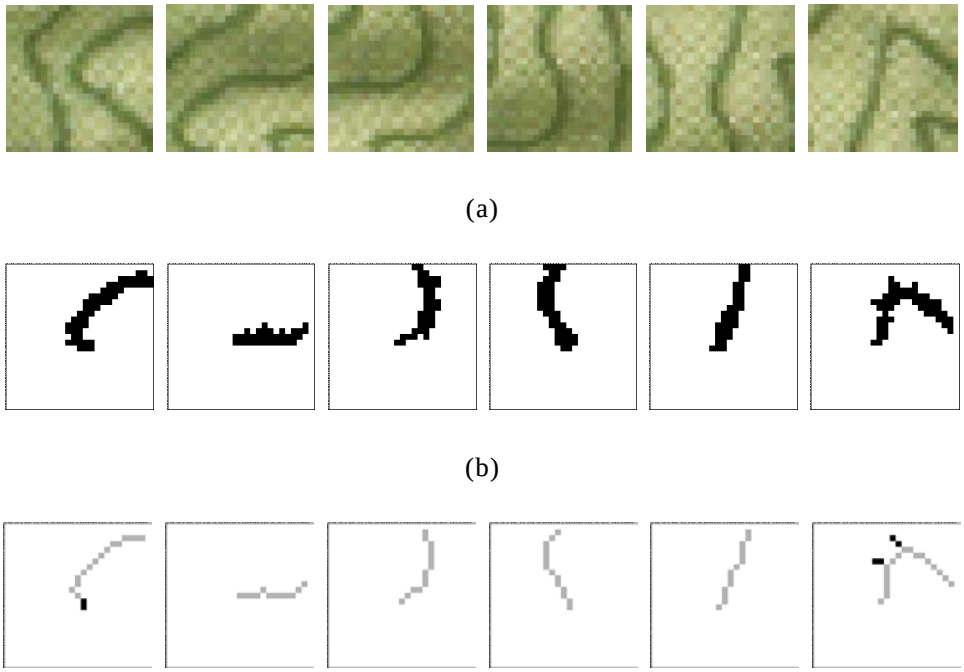
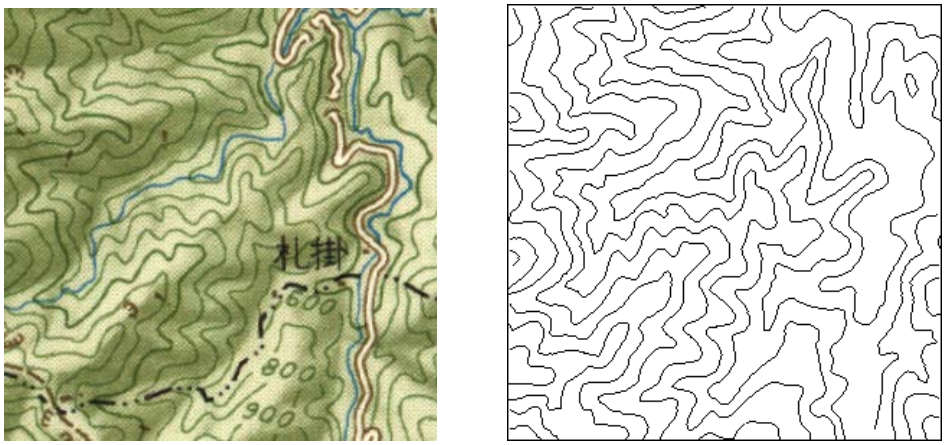


Figure 10. (a) Images in window 1-6 in Figure 9a. (b) Corresponding segmentation results of current linear feature. (c) Corresponding results of thinning and line tracking.

4. Experiments and analysis

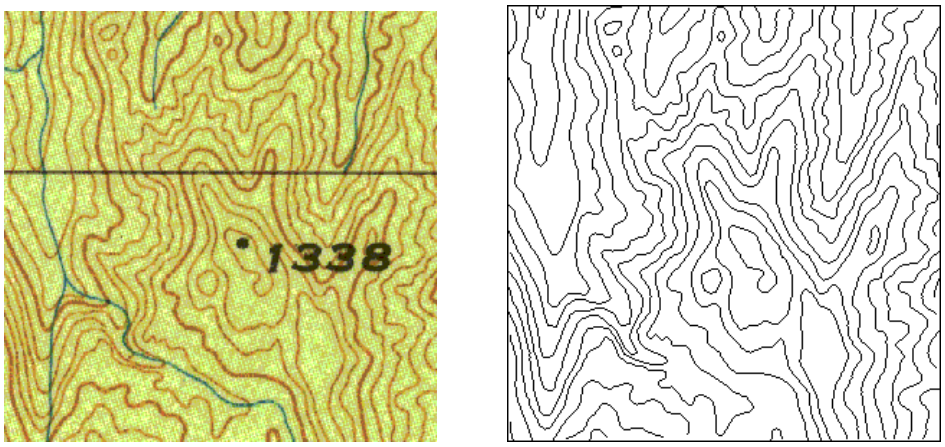
Experiments have been conducted to test our proposed method. *Figure 11(a)* is a part of a topographic map with relief shadings. The size of the image is 300×300 pixels, the resolution is 300 dpi, and the sliding window is 25×25 pixels. For each contour line, once a starting point and direction are input by a human operator, it can be tracked automatically. In the case that an intersection or a gap is met, automatic tracking will stop, and a next point on the line is input manually. After that, the line tracking continues. If the gap is wider, a few points should be collected manually to pass it. *Figure 11(b)* is the extracted result of contour lines. *Figure 12(a)* is a part of another topographic map with vegetation tints. The image size, the scanning resolution, and the window size remain unchanged. *Figure 12(b)* is the extracted result of contour lines. The average time of extracting contour lines in *Figure 11* and *Figure 12* are 200 seconds and 160 seconds respectively on

a 3 GHz Pentium (R) 4 computer. Most of the time was taken by manual input of starting points and some interventions during the tracking process, while the time required by automatic tracking is very short. A number of topographic map samples have been used to test our algorithm, and satisfactory results have been achieved.



(a) (b)

Figure 11. A part of a scanned color map with relief shadings. (a) Original image. (b) Extracted result of contour lines.



(a) (b)

Figure 12. A part of another topographic map with vegetation tints. (a) Original image. (b) Extracted result of contour lines.

A comparison with commercial software MapGIS has been made. For color maps with higher contrast between topographic features and background, it took nearly the same time to extract linear features by using the proposed method and MapGIS. But for the map images with lower contrast and lower SNR, our method is obviously more efficient. It took about 540 seconds to extract contour lines in *Figure 11* by using MapGIS. Errors (as shown in the white circle areas in *Figure 13*) often occur in the line tracking process, and more human interventions are needed to handle these problems.

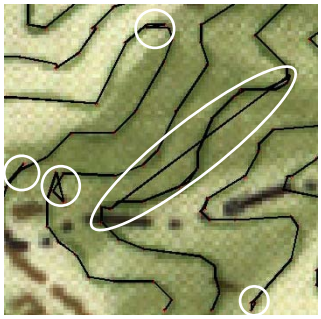


Figure 13. Extracted result of Figure 11 by MAPGIS (locally enlarged)

From the experiments, the proposed method demonstrates the following advantages:

(1) Most of the work of line tracking can be finished automatically while human operators only need to give the starting point and the directional point. Some kinds of manual interventions are allowed in case automatic tracking fails, which makes line tracking under human control, and provides the ability to correct data immediately if required.

(2) Linear features can be tracked accurately along the centerline after image segmentation and thinning. This can avoid deviation from the centerline by using only color distance to track points.

(3) The sliding window is updated continuously and the segmentation result in each window depends on the grey-level distribution and spatial relationship in current window. This makes line tracking adapt to color variations.

5. Conclusion

This paper presents a new method to extract linear features directly from scanned color topographic maps. The process of sliding window creation, adaptive image segmentation, thinning and sequential tracking can be used

as general steps for linear feature extraction. Future improvement mainly focuses on automatic tracking across the intersections and gaps so as to reduce human intervention and improve the efficiency of map digitization.

Acknowledgements

This work was supported by Natural Science Foundation of China (Grant No. 41271447). The authors would like to thank Zondy Cyber Group Co., LTD for providing MAPGIS K9 to do the experiments.

References

Ablameyko S V, Bereishik V, Homenko M, et al. (2003) A Complete System for Interpretation of Color Maps. Internation Journal of Image and Graphics 2(3): 453-479

Chen Z-Y, Qi F-H, Chen G (2000) Human Face Detection Using Color Information. Journal of Shanghai Jiaotong University 34(6):793–795

Cheng Q-M, Yang C-J, Shao Z-F (2003) Automatic Color Separation and Processing of Scanned Color Contour Maps. Journal of Geomatics 28(5):4–6

Ford A, Roberts A (1998) Color Space Conversions. http://www.poynton.com/PDFs/coloreq.pdf. Accessed 16 April 2012

Huang Z-L, Fan Y-Y, Hao C-Y (2005) Interactive Vectorization for Chromatic Scan Map. Computer Application 25(3):577–579

Khotanzad A, Zink E (2003) Contour Line and Geographic Feature Extraction from USGS Color Topographic Paper Maps. IEEE Transactions on Pattern Analysis and Machine Intelligence 25(1):18-31

Pezeshk A, Tutwiler R L (2008) Contour Line Recognition & Extraction from Scanned Colour Maps using Dual Quantization of the Intensity Image. Proceedings of the IEEE Southwest Symposium on Image Analysis and Interpretation. LA: ACM Press, New Orleans, 173–176.

Wu X-C, Wang X-R (1998) The Method of Interactive Vectorization Based on Color Map. Mini-Micro Systems 19(5):36–39

Wagstaff K, Cardie C, Rogers S, et al. (2001) Constrained K-means Clustering with Background Knowledge. Proceedings of the Eighteenth International Conference on Machine Learning. Massachusetts, USA, 577–584

Zeng Y-S, Feng J-G, He H-G (2004) A Categorizing Algorithm in the Recognition of Contour Line. Computer Engineering and Application 40(3):56–57+62