CONTOUR LINES EXTRACTION AND RECONSTRUCTION FROM TOPOGRAPHIC MAPS

Tudor GHIRCOIAS and Remus BRAD

Lucian Blaga University of Sibiu, Romania remus.brad@ulbsibiu.ro

ABSTRACT

The past years needs for Digital Elevation Models and consequently for 3D simulations requested in various applications had led to the development of fast and accurate terrain topography measurement techniques. In this paper, we are presenting a novel framework dedicated to the automatic or semiautomatic processing of scanned maps, extracting the contour lines vectors and building a digital elevation model on their basis, fulfilled by a number of stages discussed in detail throughout the work. Despite the good results obtained on a large amount of scanned maps, a completely automatic map processing technique is unrealistic and the semiautomatic one remains an open problem.

Keywords: contour line, curve reconstruction, topographic map, scanned map.

1 INTRODUCTION

The growing need for Digital Elevation Models (DEMs) and 3D simulations in various fields led to the development of fast and accurate terrain topography measurement techniques. Although interpretations photogrammetric of aerial photography and satellite imagery are the most common methods for extracting altimetry information, there is a less expensive alternative for obtaining the same or similar data: the use of scanned topographic maps. Beside the costeffectiveness there are some other advantages of this procedure:

- The inherent problems of satellite and aerial images: obstructions, cloud cover, unclear atmosphere.
- The time needed for obtaining the geographical data is significantly reduced.
- The large number of existing paper maps, which remains the most frequent form of ground surface representation.
- In the case of landscapes, which underwent changes in the past, old historical maps are often the only source of information.

The most interesting information shown by these maps are the contour lines (isolines), which are imaginary lines that join different points located at the same elevation. The isolines result from intersecting the ground surface with horizontal, parallel and equidistant planes. Topographic maps usually show not only the contours but also other thematical layers which overlap: streams and other bodies of water, forest cover or vegetation, roads, towns and other points of interest, toponyms etc.

Unfortunately, contour lines are usually the lowest layer in a map after the background layers. For this reason and not only, the extraction of these isolines becomes very difficult. There are mainly two difficulties in finding a solution to this open problem:

- isolating the contour lines from other thematical layers of the map
- processing the resulting binary image and contour reconstruction, the extracted isolines often being fragmented and affected by unwanted elements (noise).

The research in the field of contour line extraction and vectorization went on for many years, recently being published more and more articles on this subject. Usually, the articles and research focus mainly on one of the two important aspects of the extraction process: contour line segmentation and automatic reconstruction, but a bit more concern is being shown about the second aspect.

The main necessary steps described in most publications for an automatic or semiautomatic extraction procedure are the following:

- 1. preprocessing of the raster scanned map
- 2. segmentation
- 3. processing of the binary image (filtering, thinning, pruning, etc)
- 4. curve reconstruction
- 5. raster to vector conversion

Given the high complexity of most topographic maps and the importance of color information, recent

research focused mainly on color processing of the scanned documents (using color scans of the maps); this is also the case of the proposed application. In addition, because of the many possible color representations of the contour lines (although shades of brown are very common, there is no certain rule concerning the color) the segmentation method is rather semiautomatic: the user interface offers a color palette, from which the user can choose the best match.

Another problem, which makes segmentation even more difficult, is caused by the false colors, a result of the raster digitization of the map, and by the color halftoning technique used in the printing process. As further shown, the resulting noise is very difficult to remove without affecting the thin contours.

Thus, to overcome aliasing and false color Hedley [1] develops a gradient problems. thresholding method, which utilizes spatial and color space information to get rid of noise pixels. Other authors [2] are employing scanned topographic map converted into CMYK space, where the value of the K channel is used in contour line detection. In [3] Lalonde and Li use L*a*b* and modified histogram splitting for color pixel classification in order to obtain a binary image of the contour lines. S. Salvatore and P. [4] are taking advantage of the HSV color space to obtain a single color extraction. Their method implies a quantization of the HSV space to reduce the complexity and building the hue histogram of the pixels, that fall in the chromatic region of the HSV cone, the black or white pixels not being taken into consideration. Dupont et al. [5] use a watershed divide algorithm in RGB space to assign a pure map color to each pixel. [6] and [7] isolate the different map layers, which all contain a dominant color, by transforming the RGB to the L*u*v* color space. Then they determine the peaks in the histogram of the u*, v* coordinates which correspond to the color clusters. Working directly on gray-level Chen et al. [8] are getting the contour line segments by extraction of all of the linear features from histogram analysis and supervised classification. In [9], lines are removed from the image using a novel algorithm based on quantization of the intensity image followed by contrast limited adaptive histogram equalization.

A slightly greater attention reflected on a larger number of articles, has been shown to the second aspect: binary image processing and automatic reconstruction. Thus, many articles treat only the reconstruction problem, considering the contour lines already segmented and separated from the other layers. Arrighi and Soille [10] apply morphological filters to the resulted binary image for noise reduction and then try to reconnect the broken contour lines before a thinning procedure. Their reconnection algorithm uses local information: the Euclidean distances between the extreme points of the curves and their directions are combined for the joining decision. L. Eikvril et al. [11] reconnect the contours using a line tracing technique. An interruption is resolved by searching from an end point of a line. The corresponding end point might be found within a sector around the current direction. The use of Thin Plate Spline interpolation techniques are suggested by [12].

D. Xin et al. [13] also use mathematic morphology to filter the binary image [14]. After thinning the contour lines, they extract a set of key points using the c-means algorithm. These key points become input in the GGVF (Generalized gradient vector flow) snake model used for extracting the curves. Shimada et al. [15] suggest a multi-agent system to solve the problem of contour line reconstruction. The purpose of each "tracing agent", initialized by the user, is to follow a specific contour. interruptions or other problems When are encountered, it's the role of a supervisor agent to decide which route to follow. S. Salvatore and P. Guitton [4] use the global topology of topographic maps to extract and reconstruct the broken contours. Their reconstruction technique is based on two computational geometry concepts: Voronoi diagrams and Delaunay triangulations. The method is inspired from the previous work of [16] and [17] focus on the reconstruction of contour lines using a method based on the gradient orientation field of the contours. A weight is assigned to each pair of end-points according the force needed by its potential reconstructed curve to cross the field. The solution is then found by solving a perfect matching problem.

Most of the reconstruction methods discussed above use only local information and few include information about the global topology of the contour lines. This results in poor reconstructions, with many unsolved gaps or topological errors (ex. intersecting contours), especially in maps with a high curve density.

2 APPLICATION FOR A SEMIAUTOMATIC CONTOUR LINE EXTRACTION FROM SCANNED TOPOGRAPHIC MAPS

This application aims to be a solution for a semiautomatic procedure for extracting contour lines from raster images of scanned topographic maps, avoiding the time consuming effects of manual vectorization by reducing the human intervention as much as possible. Furthermore, in order to view the result of isoline extraction, the application builds a 3-dimensional model of the map terrain.

A raster to vector conversion of the contour lines from a scanned map implies two major steps: color image segmentation for obtaining a binary image of the isolines and the binary processing of this image for curve recognition. At the end of each stage, a limited intervention of the user is needed to choose the color (colors) which best isolate the contour lines and a correction of the errors produced by the automatic vectorization, respectively. Fig. 1 shows the processing-pipeline of the application.



Figure 1: Processing-pipeline of the application

2.1 Color Segmentation of the Scanned Map

Even though the color of the contour lines is not necessary unique, a color segmentation of the scanned map is required for the extraction of all the elements which have the same color with the isolines. But before any segmentation algorithm, a preprocessing of the raster image is indispensable.

2.1.1 Image smoothing

The raster image coming from the scanner has usually plenty of noise, originating from the scanning process or from the map itself, which was printed on paper using a halftoning technique. The computation of the new values on each of the color channels is accomplished by calculating a mean of the values in a neighborhood *N*. Local smoothing of the image can remove some of the impulse noise (salt and pepper), but it can also affect the thin contours, like the isolines. For this reason, the smoothing has to be applied selectively, avoiding strong filters, even though not all the noise would be removed.

In color processing, one of the most common filters is the vector median filter. The algorithm implies scanning the image with a fixed-size window. At each iteration, the distance between each pixel (x) and all the other pixels within the window is computed, the pixel which is "closest" to all the other pixels will be considered the "winning pixel", thus replacing the central pixel. The distance is based on a norm (L), which is often the Euclidean norm.

Another filter used for the same purpose is the gaussian filter, also known as gaussian blur. After approximating a convolution matrix based on the 2D-gaussian function, a convolution between this matrix and the image is performed, the resulting image being a smoothed, blurred version of the original image. The blur level is proportional to the

size of the convolution matrix.

For filtering the scanned image we chose a selective gaussian filter, because it offered the best results without affecting the fine details, such as the contour lines. The algorithm differs somewhat from the classical gaussian filter, because the convolution is applied only between the mask and the corresponding image pixels, which have a similar color with the central pixel. From a mathematical point of view, similar colors means that the Euclidean distance between the two vectors from the RGB space is below a certain threshold: $d = \sqrt{(\Delta r)^2 + (\Delta g)^2 + (\Delta b)^2}$, with $d \le threshold$... Fig. 2 shows the smoothing results using the three filters described above. As it can be noticed, the last filter has issued the best outcome, despite using a map strongly affected by halftoning.

2.1.2 Color quantization of the filtered image

In many cases, the number of different colors in an image is far greater than necessary. In the case of scanned maps, this feature becomes even more evident, because unlike other images, maps consisted of a reduced set of colors before being saved in raster format. Moreover, in order to classify the colors from the map, a reduced complexity of the process is essential.

The idea behind all quantization algorithms is to obtain an image with a reduced set of colors as visually similar with the original image as possible. A well-known algorithm, fast and with good results on a large amount of images is Heckbert's Median-Cut algorithm. Given a certain number of colors wanted to be obtained, the goal of this algorithm is that each of the output colors should represent the same number of pixels from the input image. The resulting quantized images for 128 and 1024 colors respectively are shown below.



Figure 2: Noise filtering of the scanned image a) original (unfiltered) image; b) vector median filter c) gaussian filter; d) proposed method (selective gaussian filter)

2.1.3 Color clustering using K-Means algorithm

The acquired result from the previous step is very useful, because on the basis of the reduced set of colors we can build a histogram, whose local maxima will approximate the color centers, therefore the cluster centers we want to detect. Thus, the color histogram becomes much easier to build and store the number of colors being significantly reduced.



Figure 3: Quantization results for different color numbers. a) 128 colors; b) 1024 colors

What the color histogram provides are actually the input data in the K-Means algorithm. Instead of choosing as initial cluster centers some random color values, the selected colors will be very close to these centers, such that the algorithm can converge rapidly to a optimal solution. These initial cluster centers will be extracted from the local maxima of the histogram, thereby the actual number of these clusters will vary according to each particular map.

As color similarity metric we chose the Euclidean metric in the chrominance plane (a^*, b^*) of the L*a*b* color space and not in the RGB space for perceptual uniformity reasons. K-Means is an unsupervised clustering method, which means that the algorithm is parameterless, an advantage for our purpose. To mention some of the advantages and disadvantages of the method:

Advantages:

- The algorithm offers flexibility, the vectors can easily change the clusters during the process
- Always converging to a local optimal solution
- Fast enough for the majority of the applications Disadvantages:
- The clustering result depends on the initial data set
- The optimal global solution is not always guaranteed

Through the use of the local maxima from the color histogram as the initial data set, precisely these two disadvantages will be eliminated. Not being a random data set, the algorithm's convergence is mostly guaranteed.

As described in Fig. 5 the user can easily choose one or more colors from the resulting color clusters which best match the color of the contour lines (brown in this case). The application will then extract all the pixels from the image that correspond to the cluster of the selected color/colors. The resulting binary image will become input in the next stage: contour line vectorization.



Figure 4: The background impact on contours line binarization. a) incomplete segmentation due to a single brown shade selection; b) complete segmentation by selecting more then one brown shade

Not always, though, all contour lines from a map can be extracted by choosing a single color only. Due to the way the map was printed on paper, because the color layers often overlap, the isoline colors also depend quite much on the background color. In the above example, when choosing the brown color, the contours on the white background will be fully extracted, but those on the green background will be only partially isolated (Fig. 6). Therefore, it is necessary in such cases to increase the cluster numbers and to choose more brown shades, until the binarization result will be satisfactory.



Figure 5: a) Map detail b) Representation of the map colors (input data) in the chrominance (a^*,b^*) plane; c) Representation of the color centers in the (a^*,b^*) plane, following the clustering process.



Figure 6: The original map and the result of the automatic clustering process

2.2 Binary image processing

Apart from the contour lines, the binarized image created in the previous stage contains many other "foreign objects". As a result, a preprocessing of the binary image is needed for noise removal that would otherwise significantly alter the vectorization result.

2.2.1 Binary noise removal

The noise removal can be realized on the basis of minimum surface criteria that the contours have to satisfy. The morphological opening operation has been taken into consideration but the results were not satisfactory because it removed the very thin contour lines despite of using the smallest possible structuring element (3x3).

Another noise removal method was also based on minimum surface criteria. The idea is to find the connected components in the image and to count their pixels. All the components having the number of pixels below a certain threshold will be

disregarded.

The standard algorithm performs a labeling of all the connected components from the image by assigning a unique label to the pixels belonging to the same component. The algorithm is recursive and has the following steps:

1. Scan the image to find unlabeled foreground pixel p and assign it a label *L*.

2. Assign recursively the label L to all p neighbors (belonging to the foreground).

3. If there are no more unlabeled pixels, the algorithm is complete.

4. Go to step 1.

The above recursive algorithm will not be applied on the entire image, but only within a $n \times n$ sized window, with *n* depending on the contour line density. If the density is high, then a smaller value is recommended. Therefore, the connected components having a larger number of pixels then a specified threshold, located within a $n \times n$ sized window, will be removed. In addition to the speed gain, another advantage is that the small curve segments and intermediate points located in the gap between two broken isolines will remain unaffected, thus not influencing the subsequent reconstruction procedure. 2.2.2 Contour line thinning

Since the vectorization process would get unnecessary complicated without reducing the contour line thickness, a skeletonization or thinning procedure becomes essential. Another reason for that comes from the mathematical perspective, because lines and curves are one-dimensional, which means that their thickness doesn't offer any useful information and may only make the vectorization process more difficult.

As in [13], we used the modified Zhang-Suen thinning algorithm [18] for contour thinning. The algorithm is easy to implement, fast and with good results for our problem, where it is critical for the curve end-points to remain unaltered.

The method involves two iterations where, depending on different criteria, pixels are marked for deletion:

Let P_n , n = 1,8, be the 8 neighbors of the current pixel I(i, j), P_1 is located above P, the neighbors are numbered clockwise.

Subiteration 1: The pixel I(i, j) is marked for deletion if the following 4 conditions are satisfied:

- Connectivity = 1
- I(i, j) has at least 2 neighbors but no more then 6
- At least one of the pixels: P_1 , P_3 , P_5 are background pixels
- At least one of the pixels: P_3 , P_5 , P_7 are background pixels. The marked pixels are deleted.

Subiteration 2: The pixel is marked for deletion if the following 4 conditions are satisfied:

- Connectivity = 1
- I(i, j) has at least 2 neighbors but no more then 6
- At least one of the pixels: P_1 , P_3 , P_7 are background pixels
- At least one of the pixels: P_1 , P_5 , P_7 are background pixels. The marked pixels are deleted.

If at the end of any subiteration there are no more marked pixels, the skeleton is complete.



Figure 7: Before and after the thinning process

Following the skeletonization process the resulting image can contain some "parasite" components like small branches which may affect the curve reconstruction. To remove these branches a combination of morphological operations will be used. First of all, the maximum size of the branches has to be established. The branches and segments with the length below this threshold will be removed, but all the other will not suffer any change at all. This restriction is required for the pruning algorithm. The chosen threshold or the length of the segments which will be deleted corresponds to the n number of iterations.

The pruning algorithm has 4 steps that involve morphological operations:

1. *n* iterations of successive erosions using the following structuring elements with all the 4 possible rotations:

0	0	0		1	0	0	
1	1	0	rotated 90° and	0	1	0	rotated 90°
0	0	0		0	0	0	

2. Find the end-points of all the segments from the previous step.

3. n iterations of successive dilations of the endpoints from step 2, using the initial image as a delimiter.

4. The final image is the result of the reunion between the images from step 1 and 3.



Figure 8: Before and after the pruning algorithm

As a result of the skeletonization and even after the described pruning process, the curves may still have some protrusions and sharp corners. Hence, a contour smoothing procedure could be useful for a more natural and smooth aspect.

2.2.3. Reconstruction of the contour lines

One of the most complex and difficult problems in the automatic extraction of contour lines is the reconnection of the broken contours. Linear geographical features and areas overlap in almost alltopographic maps. If two different linear features cross each other or are very close to one another, the color segmentation will determine the occurrence of gaps or interruptions in the resulting contours. The goal of the reconstruction procedure is to obtain as many complete contours as possible, i.e. contours that are either closed curves or curves which touch the map borders. Every end-point of a curve should be joint with another corresponding end-point or should be directed to the physical edge of the map. N. Amenta [16] proposed a reconstruction method which uses the concept of medial axis of a curve λ : a point in the plane belongs to the medial axis of a curve λ , if at least two points located on the curve λ are at the same distance from this point.



Figure 9: The thin black curves are the medial axis of the blue curves

Amenta proved that the "crust" of a curve can be extracted from a planar set of points if the points that

describe the curve are sampled densely enough. His assumption was that the vertices of the Voronoi diagram approximate the medial axis of the curve. When applying a Delaunay triangulation to both the original set of points and the set of the Voronoi vertices, every edge of the Delaunay triangulation which joins a pair of points from the initial set, forms a segment of the reconstructed curve (the crust).

Summarily, Amenta's reconstruction method is the following:

1. Let *S* be a planar set of points and *V* the vertices of the Voronoi diagram of *S*. *S'* is the union of *S* and *V*.

2. *D* is the Delaunay triangulation of S'.

3. An edge of D belongs to the crust of S if both endpoints belong to S.



Figure 10: Result of Amenta's reconstruction method. a) Initial set of points; b) Voronoi diagram; c) Delaunay triangulation (with the crust highlighted); d) original contours; e) Reconstructed contours

As it can be noticed, not all the gaps are being filled. Therefore, only a part of the curves will become complete after applying the method described above. Repeating the algorithm using the binary image of the reconstructed curves as input would produce only minor changes. Therefore, in order for a new reconstruction to be successful, all the newly completed contours have to be deleted from the input image.

At the end of the reconstruction algorithm the final result of the automatic processing will be presented to the user for a manual correction of the remaining errors. Some of these errors are inevitable, an automatic error recognition being very difficult to implement. A common example are the errors that occur because of the elevation values which often have the same color as the contour lines and are inserted between them (see Fig. 11).



Figure 11: Errors resulting from the elevation values

2.3 Contour line vectorization and interpolation

The application automatically verifies if the list of contour lines contains only complete contours. When there are no more incomplete isolines left, the user will be assisted in the following procedure, namely assigning an elevation to each contour line. The isolines can be then saved in a vector format and the resulting DEM based on the original map can be viewed. The most common representation used when saving a curve in a vector format, also employed in our case, is the well-known Freeman chain-code representation. For a good 3-dimensional terrain visualization, the contour line elevations are not sufficient, because a step disposal wouldn't look natural. For this reason, different interpolation techniques have been developed, some of them use a 3D version of the Delaunay triangulation, other use cubic interpolation etc.



Figure 12: Chain-code example. a) coding the 8 directions; b) initial point and edge following

For our application we used a simpler but very efficient technique with good results. The method, implemented by Taud [19], uses a series of morphological operations on a raster grid (similar to a binary representation of the extracted contour lines, where instead of a binary 1, the elevation values of the isolines are stored). Through the use of successive erosions or dilations, new families of intermediate contour lines are generated and their elevations are computed based on the neighboring contours. The process is repeated until the desired "resolution" is obtained, i.e. the elevation values of the majority of the intermediate points are known. Thus, an intermediate curve located between two isolines corresponds to the medial axis of the two.

3 RESULTS

Our tests have been carried out on scanned images of relatively complex topographic maps,

	Quality	Curve	No. of	No. of	No. of gaps filled	Unsolved	Wrong	Success
		Size	contours	gaps	correctly	gaps	connections	rate
Map1	medium	thin	9	26	24	2	0	92.31
Map2	medium	thick	9	26	23	3	0	88.46
Map3	poor	thick	17	74	57	13	4	77.03
Map4	high	thin	10	24	18	5	1	75
Map5	poor	thin	18	38	30	6	2	78.95
Map6	high	thick	16	50	44	6	0	88
Map7	poor	thin	52	135	113	20	2	83.7

Table 1:

with many colors and overlapping layers. The images represent older or more recent maps scanned in high quality JPEG or TIF file formats.

The algorithm was tested on more than 50 scanned images of relatively complex topographic maps, with many colors and overlapping layers. Some of the scanned maps have a poor quality, being affected by noise caused mainly by the printing procedure (halftone) or by the old paper. In the table below, we present an evaluation of the algorithm for 7 different maps. The overall success rate (number of solved errors resulting in complete contours from the total number of errors from the segmentation process) was 83.35%. Out of 64 unsolved gaps or incorrect contour connections 12 came from the elevation values.



e) **Figure 13:** a) scanned map; b) result of color segmentation; c) processed and thinned binary image; d) result of automatic contour line reconstruction; e) 3D-DEM



Figure 14: a) scanned map; b) result of color segmentation; c) processed and thinned binary image; d) result of automatic contour line reconstruction; e) 3D-DEM

4 CONCLUSIONS

This work was dedicated to the automatic and semiautomatic processing of scanned maps, aiming the vectorization of contour lines and building a digital elevation model on their basis. Attaining this objective has implied a number of stages discussed in detail throughout the work. We will recall the most important ones:

1. The quality of the contour line extraction process depends to a great extent on the color processing module. Despite the problems of color image segmentation, using the proposed methods we obtained good results even on maps that were strongly affected by noise, coming either from the scanner or from the map itself (halftone, salt-andpepper, false colors etc.). Reducing the complexity through a quantization of the color space and through the use of CIELab color space, more perceptual uniform then RGB, a proper color clustering has been attained. Nevertheless, if the original map is poor an automatic color classification will fail.

2. The second stage implied a processing of the resulting binary image from the previous step and the implementation of an automatic contour reconstruction method. The morphological filtering and thinning operations applied on the binary image succeeded to remove most of the noise and unnecessary elements. Because isolines are often the lowest foreground layer of a map, many other features were superimposed on these, leading to interruptions and broken contours. The efficiency of the gap-filling algorithm consists in both a global and local approach. Thus, using the geometric properties of the curves, the achieved results were satisfactory, most of the gaps being automatically solved.

Despite the good results obtained on a large amount of scanned maps, a completely automatic map processing technique is unrealistic and the semiautomatic one remains an open problem.

The future development prospects are related mainly to two aspects:

- Image segmentation (color processing)
- Improvement of the automatic reconstruction techniques

It has been proven that the result of contour line extraction is strongly related to the quality of the segmentation process. The background color on which the contours have been printed also plays a major role in the extraction, because it has a big influence on their color. In this respect, algorithms that take into account the background color for every foreground pixel should be searched for, in order to obtain a better segmentation technique.

As an improvement of contour line reconstruction, adding more global information about isolines and introducing OCR techniques for extracting elevation values, could be useful. Thus, some of the common causes that lead to broken contours can be eliminated. At the same time, an automatic or semiautomatic procedure for assigning altitude information to the extracted isolines can be taken advantage of.

5 REFERENCES

- M. Hedley, H. Yan: Segmentation of color images using spatial and color space information, Journal of Electron. Imaging, Vol. 1, pp. 374-380 (1992)
- [2] R. Q. Wu, X. R. Cheng, C. J. Yang: Extracting contour lines from topographic maps based on

cartography and graphics knowledge, Journal of Computer Science & Technology 9(2), pp. 58-64 (2009)

- [3] M. Lalonde, Y Li: Contour Line Extraction from Color Images of Scanned Maps, In Proceedings of the 9th international Conference on Image Analysis and Processing, Vol. I, Ed. Lecture Notes In Computer Science, vol. 1310, Springer-Verlag, London, pp.111-118 (1997)
- [4] S. Spinello, P. Guitton:. Contour Lines Recognition from Scanned Topographic Maps, Journal of Winter School of Computer Graphics, Vol.12, No.1-3 (2004)
- [5] F. Dupont, M. Deseilligny, M. Gondran: Terrain Reconstruction from Scanned Topographic Maps, Proceeding Third Internationl Workshop Graphics Recognition, pp. 53-60 (1999)
- [6] N. Ebi, B. Lauterbach, W. Anheier: An imageanalysis system for automatic dat:-acquisition from colored scanned maps, Machine Vision and Applications, 7(3), pp. 148-164 (1994)
- [7] L. W. Kheng: Color Spaces and Color-Difference Equations, Technical Report, Dept. of Computer Science, National University of Singapore (2002)
- [8] Y. Chen, Runsheng Wang, Jing Qian: Extracting contour lines from commonconditioned topographic maps. Geoscience and Remote Sensing, IEEE Transactions on, 44(4), pp. 1048-1057 (2006)
- [9] A. Pezeshk, R. L. Tutwiler: Contour Line Recognition & Extraction from Scanned Colour Maps Using Dual Quantization of the Intensity Image, Proceedings of the 2008 IEEE Southwest Symposium on Image Analysis and interpretation, pp. 173-176 (2008)
- [10] P. Arrighi, P. Soille: From scanned topographic maps to digital elevation models, Geovision '99, International Symposium on Imaging Applications in Geology, pp. 1-4 (1999)
- [11] L. Eikvil, K. Aas, H. Koren: Tools for interactive map conversion and vectorization, International Conference on Document Analysis and Recognition, Third International Conference on Document Analysis and Recognition, Vol. 2, pp. 927-930 (1995)
- [12] A. Soycan, M. Soycan: Digital Elevation Model Production from Scanned Topographic Contour Maps Via Thin Plate Spline Interpolation, The Arabian Journal for Science and Engineering Science, 34(1A), pp. 121-134 (2009)
- [13] D. Xin, X. Zhou, H. Zhenz: Contour Line Extraction from Paper-based Topographic Maps, Journal of Information and Computing Science, 1(275–283) (2006).
- [14] J. Du, Y. Zhang: Automatic extraction of contour lines from scanned topographic map,. Geoscience and Remote Sensing Symposium,

IGARSS '04, Proceedings, IEEE International, Vol. 5, pp. 2886-2888 (2004)

- [15] S. Shimada, K. Maruyama, A. Matsumoto and K. Hiraki: Agent-based parallel recognition method of contour lines, In Proceedings of the Third international Conference on Document Analysis and Recognition, Vol. 1, IEEE Computer Society, Washington, DC, pp. 154 (1995)
- [16] N. Amenta, M. Bern, D. Eppstein: The Crust and the Beta-Skeleton: Combinatorial Curve Reconstruction, Graphical models and image processing, 60(2), pp. 125-135 (1998)
- [17] J. Pouderoux, S. Spinello: Global Contour Lines Reconstruction in Topographic Maps, In Proceedings of the Ninth international Conference on Document Analysis and Recognition, Vol. 02, ICDAR, IEEE Computer Society, Washington, DC, pp. 779-783 (2007)
- [18] T. Y. Zhang, C. Y. Suen: A fast parallel algorithm for thinning digital patterns. Commun, ACM 27, 3, pp. 236-239 (1984)
- [19] H. Taud, J. Parrot, R. Alvarez: DEM generation by contour line dilation, Comput. Geosci, 25, 7, pp. 775-783 (1999)