A new framework for the extraction of contour lines in scanned topographic maps

Tudor GHIRCOIAS and Remus BRAD

Abstract 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 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 remains an open problem.

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 photogrammetric interpretation 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.

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. 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

Tudor GHIRCOIAS
Lucian Blaga University of Sibiu, B-dul Victoriei 10 550024 Sibiu, Romania,
e-mail: tudor.ghircoias@ulbsibiu.ro

Remus BRAD
Lucian Blaga University of Sibiu, B-dul Victoriei 10 550024 Sibiu, Romania,
e-mail: remus.brad@ulbsibiu.ro

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, isolines being often fragmented and affected by unwanted elements (noise).

Thus, to overcome aliasing and false color problems, [7] are employing scanned topographic map converted into CMYK space, where the value of the K channel is used in contour line detection. S. Salvatore and P. Guitton [5] are taking advantage of the HSV color space to obtain a single color extraction. Working directly on gray-level Chen et al. [9] are getting the contour line segments by extraction of all of the linear features from histogram analysis and supervised classification. In [2] lines are removed from the image using a novel algorithm based on quantization of the intensity image followed by contrast limited adaptive histogram equalization.

Many papers treat only the reconstruction problem, considering the contour lines already segmented and separated from the other layers. The use of Thin Plate Spline interpolation techniques are suggested by [4]. D. Xin et al. [8] also use mathematic morphology to filter the binary image. After thinning the contour lines, they extract a set of key points using the c-means algorithm. S. Salvatore and P. Guitton [3] use the global topology of topographic maps to extract and reconstruct the broken contours. The method is inspired from the previous work of Amenta et al. [1].

Most of the reconstruction methods discussed above use only local information and few include information about the global topology of the contour lines. It 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.

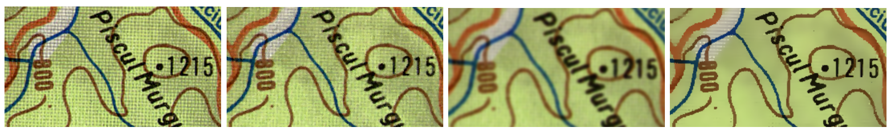
The processing-pipeline of the application is the following:

Scanned topographic map → Preprocessing → Color quantization → Color clustering → *User intervention* → Binary image processing → Skeletonization → Gap filling → *User intervention* → Vectorization → Contour line interpolation → Digital Elevation Model (DEM)

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.

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. 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.



(a) Original image (b) Vector median filter (c) Gaussian filter (d) Proposed method

Fig. 1 Noise filtering of the scanned image. As it can be noticed, the last filter has issued the best outcome, despite using a map strongly affected by halftoning.

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. A well-known algorithm, fast and with good results on a large amount of images is Heckberts 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 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. The color histogram provides 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 an optimal solution. These initial cluster centers will be extracted from the local maxima of the histogram.

As color similarity metric we chose the Euclidean metric in the chrominance plane *(a*, b*)* of the *L*a*b** color space. 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 algorithms convergence is mostly guaranteed. As described in Fig. 2, the application will extract all the pixels from the image that correspond to the cluster of the selected color/colors.

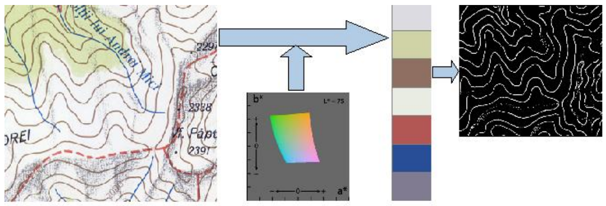


Fig. 2 The user can easily choose one or more colors from the resulting color clusters which best match the color of the contour lines (brown in this case). The resulting binary image will become input in the next stage: contour line vectorization

2.2 Binary image processing and vectorization

Apart from the contour lines, the binary 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. The noise removal can be accomplished on the basis of minimum surface criteria that contours have to satisfy. As in [8], we used the modified Zhang-Suen thinning algorithm 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.

One of the most complex problems we came across is the reconnection of the broken contours. Linear geographical features and areas overlap in almost all topographic 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 in the resulting contours. The goal of the reconstruction procedure is to obtain as many complete contours as possible. Every end-point of a curve should be joined with another corresponding end-point or should be directed to the edge of the map.

N. Amenta [1] proposed a reconstruction method which uses the concept of medial axis of a curve λ. 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.

(a) Original contours (b) Reconstructed contours

Fig. 3 As it can be noticed, not all the gaps are being filled. Therefore, repeating the algorithm for the remaining incomplete contours might be necessary

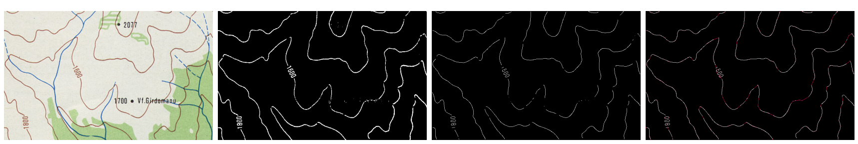
At the end of the reconstruction algorithm the final result of the automatic processing will be presented to the user for a manual correction of the remaining errors and assigning an elevation to each contour line. For the contour line interpolation we followed H. Taud's idea [6]. The isolines can be then saved in a vector format.

3 Results

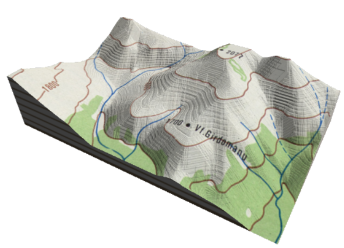
The algorithm was tested on more than 50 scanned images of relatively complex topographic maps, with many colors and overlapping layers. Some of the scanned maps have a poor quality, being affected by noise caused mainly by the printing procedure (halftone) or by the old paper.In the table below, we 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.

Table 1 Algorithm evaluation for 7 different maps

	Quality	Curve Size	No.of contours	No.of gaps	No.of gaps filled correctly	Unsolved gaps	Wrong connec-tions	Success rate
Map 1	medium	thin	9	26	24	2	0	92.31
Map 2	medium	thick	9	26	23	3	0	88.46
Map 3	poor	thick	17	74	57	13	4	77.03
Map 4	high	thin	10	24	18	5	1	75
Map 5	poor	thin	18	38	30	6	2	78.95
Map 6	high	thick	16	50	44	6	0	88
Map 7	poor	thin	52	135	113	20	2	83.7



(a) Scanned map (b) Result of color segmentation (c) Processed and thinned binary image (d) Result of automatic contour line reconstruction



(e) 3D-DEM

Fig. 4 Results

4 Conclusions

We will recall the most important stages, discussed throughout the work:

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. 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, the clustering will fail.

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.

References

1. Amenta, N., Bern, M., Eppstein (1998) The Crust and the Beta-Skeleton: Combinatorial Curve Reconstruction. Graphical models and image processing. 60(2):125-135. doi: 10.1006/gmip.1998.0465
2. Pezeshk A., Tutwiler R. L. (2008) Contour Line Recognition and Extraction from Scanned Colour Maps Using Dual Quantization of the Intensity Image. Proceedings of the 2008 IEEE Southwest Symposium on Image Analysis and interpretation, 173-176. doi:10.1109/SSIAI.2008.4512313
3. Pouderoux, J. Spinello, S. (2007) Global Contour Lines Reconstruction in Topographic Maps. Proceedings of the Ninth international Conference on Document Analysis and Recognition (02) 779-783. doi: 10.1109/ICDAR.2007.4377021
4. Soycan A., Soycan M. (2009) Digital Elevation Model Production from Scanned Topographic Contour Maps Via Thin Plate Spline Interpolation. The Arabian Journal for Science and Engineering Science, 34(1A):121-134
5. Spinello, S., Guitton, P. (2004) Contour Lines Recognition from Scanned Topographic Maps. Journal of Winter School of Computer Graphics, vol.12, no.1-3.
6. Taud, H., Parrot, J., Alvarez, R. (1999) DEM generation by contour line dilation. Comput. Geosci. 25, 7 (Aug. 1999), 775-783. doi:10.1016/S0098-3004(99)00019-9
7. Wu R.Q., Cheng X.R., Yang C.J. (2009) Extracting contour lines from topographic maps based on cartography and graphics knowledge. Journal of Computer Science & Technology 9(2):58-64
8. Xin, D., Zhou, X., Zhenz, H. (2006) Contour Line Extraction from Paper-based Topographic Maps. Journal of Information and Computing Science, 1(275283). doi:10.1.1.94.8371
9. Chen Y., Runsheng Wang, Jing Qian, (2006) Extracting contour lines from common-conditioned topographic maps. Geoscience and Remote Sensing, IEEE Transactions on, 44(4):1048-1057. doi:10.1109/TGRS.2005.861478