Peter Tarabek *

GENERAL VECTORIZATION OF LINE OBJECTS IN DRAWN MAPS

Drawn maps consist of multiple object types. The most important are line objects which represent infrastructure. Attributes of these objects are essential for many tasks but in raster format they provide only low level information. Vectorization must be used to obtain vector data. In this paper general vectorization process consisting of five stages is proposed. For these stages short discussion and basic recommendations are given and some proper methods are presented.

Keywords: vectorization, skeletonization, skeleton, thinning, raster-to-vector

1. Introduction

Extraction of graphical information from raster maps is a complex problem. Vector data which represent the necessary information on networks and need less space for their storage are used in many tasks and are essential for GIS and CAD applications. Drawn maps consist mainly of line objects whose length is much larger than the thickness. These objects may represent road infrastructure but the raster data provide only low level information. Vectorization process needs to be used to transform this data to vector form. Conservation of connectivity and shape of the line objects are two most important conditions for vectorization process.

2. Vectorization process

There are three main approaches which can be used in vectorization process: sparse-pixel vectorization (SPV), matching opposite contours (MOC) and skeletonization.

SPV [1, 2] is fastest from the mentioned approaches because it processes only small portion of pixels in image. The main disadvantage of SPV methods is that they not preserve connectivity if the parameters are not set up correctly. MOC methods [2, 3] don't visit all pixels and work either directly on the contours or on a polygonal approximation of it. However hard-to-master thresholds and heuristics need to be used for complex drawings [2].

Skeletonization methods compute medial axis called skeleton. There are many papers about different skeletonization techniques [4, 5]. Some of these techniques can directly produce vectors. The most common are thinning techniques [6, 7, 8] which remove outer pixels layer by layer in iterative process until one pixel thick skeleton is produced. Thinning conserve connectivity and shape of the

objects and is used in the proposed vectorization process. On the other hand thinning is sensitive to noise and sometimes produces distortion in junction points.

The process of vectorization proposed in this paper consists of 5 stages (Fig. 1) and it is focused on line objects. Because of huge variety of possible representations of individual objects, there is no specific vectorization process which can deal with all drawn maps. When dealing with line objects this situation is easier. These objects are represented in a very similar manner so if the right methods are applied for each stage this process can be used for a wide range of different maps.
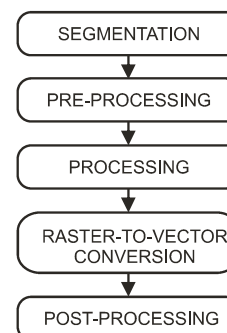

Fig. 1 Stages of the proposed general vectorization process

2.1 Segmentation

Besides line objects, maps usually contain other objects like text, symbols and regions. In this stage line objects are separated from others. The result is a binary image where all line objects

* Peter Tarabek
  Department of Transportation Networks, Faculty of Management Science and Informatics, University of Zilina, Slovakia,
  E-mail: Peter.Tarabek@fri.uniza.sk

should be represented as black pixels and all other objects including the background as white pixels. This reduces amount of data and it also speeds up and simplifies future processing.

The separation of line objects is accomplished by thresholding. A general condition for threshold can be defined as follows:

$$g(x,y) = \begin{cases} 0, & \text{if } f(x,y) < T \\ 1, & \text{if } f(x,y) \geq T \end{cases} \quad (1)$$

where $g(x,y)$ is the value of pixel with coordinates x, y in a resulting binary image, $f(x,y)$ is the original value of pixel and T is a threshold value.

Several threshold values are usually used to correctly separate objects in image. There are many local and global threshold techniques which can be used. Although a correct automatic set up of threshold parameters for all kinds of maps is not possible some type of automation is achievable [9, 10, 11].

2.2 Pre-processing

In pre-processing, a binary image is processed to remove imperfections and to amplify desired features of line objects. A binary image should be improved according to weaknesses of techniques used in a processing step to prevent future errors. Imperfections in a binary image are caused by a bad condition of paper maps, process of scanning, usage of segmentation methods, complexity of map and variety of possible representations. In Fig. 2 thinning was used to create a skeleton from an inaccurate input. The acquired skeleton differs from the desired skeleton because thinning is sensitive to imperfections in the original image.
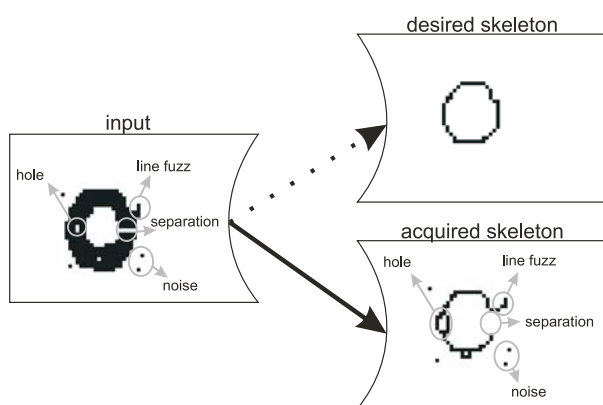

Fig. 2 Desired and acquired skeleton of input image

Accurate pre-processing for thinning should fulfill these tasks:
- remove isolated small objects
- reduce boundary noise (contour intrusions and protrusions)
- fill small holes in objects
- connect disconnected objects

Binary morphology operators opening and closing provide very good results for this kind of tasks. Operations opening and closing can be combined to remove all the mentioned imperfections. Order of operations and number of repetitions are the most important decisions which need to be made in order to obtain accurate results [12].

2.3 Processing

In this step the binary image is prepared for raster-to-vector conversion. Features like thickness of objects, information about their contours and number of components can be extracted and some level of feed back is possible by measuring the quality of result [13] and analyzing extracted features.

Thinning which produces a skeleton is used in the proposed vectorization process. The skeleton is ideal for line objects because it is represented by a set of one pixel thick lines which are a natural representation of line objects such as roads. Thinning should fulfill these requirements:
- Skeleton should be one pixel thick
- Connectivity should be preserved
- Shape and position of the junction points should be preserved
- Skeleton should lie in the middle of a shape (medial axis)
- Skeleton should be immune to noise (especially to boundary noise)
- Excessive erosion should be prevented (length of lines and curves should be preserved)

As noted previously, thinning is sensitive to noise and to other imperfections like holes in objects. These problems can be solved by binary morphology operators recommended in the previous chapter. Another often mentioned disadvantage is a low computational speed which can be improved by using the contour approach to thinning proposed in [14].

2.4 Raster-to-vector conversion

Conversion to vector form for one pixel thick skeleton can be performed in two main steps: nodes and edges recognition. Usually the local approach is used. In this approach 3×3 neighborhood of each pixel is inspected. In this case candidates for nodes can be recognized by having a connectivity number (CN) > 2 (number of black to white transitions in 3×3 neighborhood). Nodes are selected from these candidates based on additional rules. A special case of nodes are end points which can be recognized by having CN $= 1$. In the next step edges are recognized. Edge pixels have CN $= 2$ so they can be easily traced. This approach produces excessive nodes and edges and need further processing to yield accurate results.

Another possibility is to form clusters of candidates for nodes. For each candidate the priority number based on the number of 8-neighbors (N8) candidates and the number of 4-neighbors (N4) candidates is computed. For each cluster, the candidate with
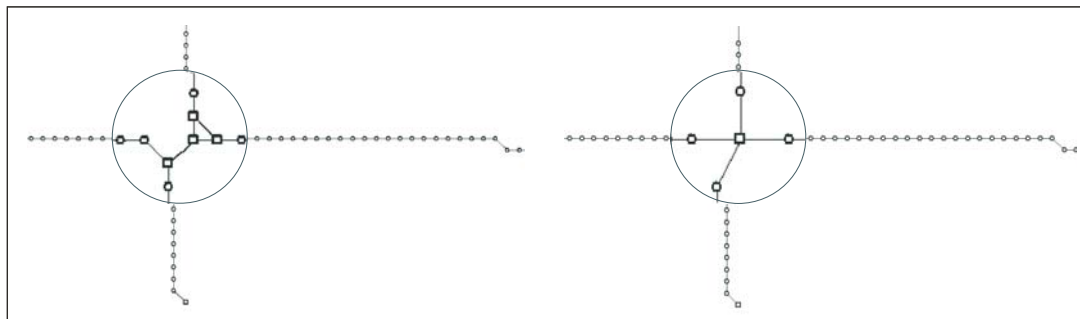
Fig. 3 Local approach (left) and cluster approach (right)

largest priority is selected for a node. This approach is used in the proposed vectorization process. The difference between the local approach and cluster approach is shown in Fig.3.

2.5 Post-processing

After the raster-to-vector conversion is done vector data usually contain a large number of vertices that can be reduced by some kind of polygonal approximation [15, 16]. Also straight lines and arcs can be recognized in this phase [17]. Vector data can be used to perform additional processing including pruning, removing incorrectly separated objects, improving quality of junction points and recognizing attributes such as length, width and color of the edges.

3. Conclusion

The general vectorization process for line objects in drawn maps was discussed in this paper. For each of its steps a short discussion of the problem was given and suitable methods were recommended. The vectorization process is based on thinning which conserves connectivity and shape of line objects. The operator is required only at the beginning of the process to set up parameters. Processing and raster-to-vector conversion are fully automatic and do not require any settings.

The results of the proposed vectorization process are shown in Figs. 4 and 5. The processing time for the map in Fig. 4 with dimensions 985×1114 pixels was less than 2 seconds. For the map with dimensions 600x396 pixels shown in Fig. 5, processing time was less than 0.1 second.
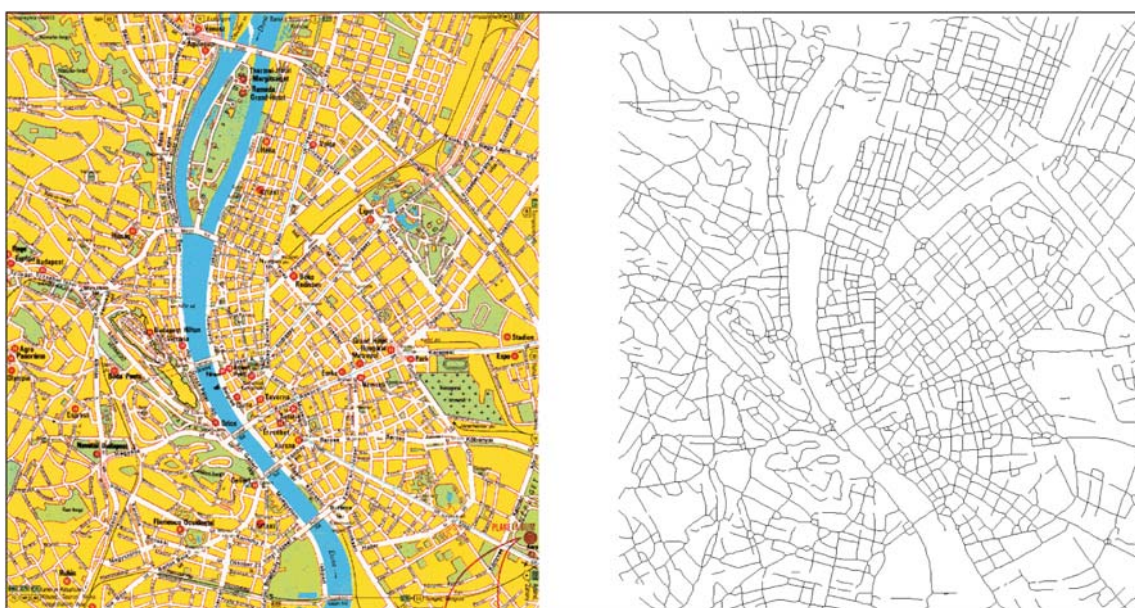

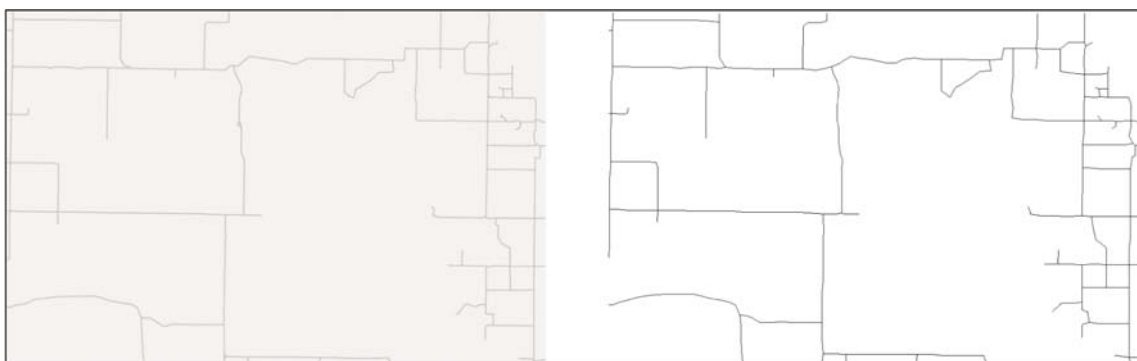*Fig. 4 Result of proposed vectorization process*

Fig. 5 Result of proposed vectorization process

References

[1] DORI, D., LIU, W.: *Sparse Pixel Vectorization: An Algorithm and Its Performance Evaluation,* IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 21, 1999, ISSN 0162-8828, pp. 202–215.

[2] TOMBRE, K., AH-SOON, C., DOSCH, P., MASINI, G., TABBONE, S.: *Stable and Robust Vectorization: How to Make the Right Choices,* Graphics Recognition-Recent Advances, 2000, ISBN 978-3-540-41222-9, pp. 3–18.

[3] HORI, O., TANIGAWA, S.: *Raster-to-Vector Conversion by Line Fitting Based on Contours and Skeletons,* Proceedings of the Second International Conference on Document Analysis and Recognition, 1993, pp. 353–358.

[4] ZOU, J. J., HONG, Y.: *Vectorization of Cartoon Drawings,* Selected papers from the Pan-Sydney workshop on Visualisation - Volume 2, 2000, ISSN 1445-1336, pp. 77–78.

[5] SUZUKI, S., UEDA, N.: *Robust Vectorization Using Graph-Based Thinning And Reliability-Based Line Approximation,* Proceedings CVPR '91, 1991, ISBN 0-8186-2148-6, pp. 494–500.

[6] BERNARD, T. M., MANZANERA, A.: *Improved Low Complexity Fully Parallel Thinning Algorithm,* In Proceedings of the 10th international Conference on Image Analysis and Processing, 1999, ISBN 0-7695-0040-4.

[7] ZHANG, T. Y., SUEN, C. Y.: *A fast parallel algorithm for thinning digital patterns,* Commun. ACM, 27(3), 1984, ISSN 0001-0782, pp. 236–239.

[8] LAM, L., LEE, S., SUEN, C. Y.: *Thinning Methodologies-A Comprehensive Survey,* IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 14, No. 9, 1992, ISSN 0162-8828, pp. 869–885.

[9] BOBIN, J., STARCK, J.-L., FADILI, J., MOUDDEN, Y., DONOHO, D.L.: *Morphological Component Analysis: an adaptive thresholding strategy,* IEEE Transactions on Image Processing, Vol. 16, No. 11, 2007, ISSN 1057-7149, pp. 2675–2681.

[10] OTSU, N.: *A threshold selection method from gray-level histogram,* IEEE Transactions on System, Man and Cybernetics, Vol. 9, No. 1, 1979, ISSN 0018-9472, pp. 62–66.

[11] LIAO, P. S., CHEN, T. S., CHUNG, P. C.: *A fast Algorithm for Multilevel Thresholding,* Journal of Information Science and Engineering 17, 2001, pp. 713–727.

[12] TARABEK, P.: *Morphology image pre-processing for thinning algorithms,* Journal of Information, Control and Management Systems, Vol. 5, No.1, 2007, ISSN 1336-1716, pp. 131–138.

[13] TARABEK, P.: *Performance measurements of thinning algorithms,* Journal of Information, Control and Management Systems, Vol. 6, No. 2, 2008, ISSN 1336-1716, pp. 125–132.

[14] TARABEK, P.: *A Contour Approach to Thinning Algorithms,* Communications – Scientific Letters of the University of Zilina, Vol. 11, 3/2009, ISSN 1335-4205, pp. 43–48.

[15] KOLESNIKOV, A., FRANTI, P.: *Data Reduction of Large Vector Graphics,* Research Report, A-2003-2, Department of Computer Science, University of Joensuu, Finland, 2003, ISBN 952-458-311-9.

[16] SARFRAZ, M.: *Optimal Curve Fitting to Digital Data,* International Journal of WSCG, Vol. 11, No. 1, 2003, ISSN 1213-6972.

[17] DOSCH, P., MASINI, G., TOMBRE, K.: *Improving Arc Detection in Graphics Recognition,* Proc. 15th Int'l Conf. Pattern Recognition (ICPR '00), Vol. 2, 2000, ISBN 0-7695-0750-6, pp. 243–246.