Principle of vectorization of classified images

Jaroslav Fojtík

Czech Technical University, Faculty of Electrical Engineering
Center for Machine Perception
121 35 Prague 2, Karlovo náměstí 13, Czech Republc
phone +42 2 24357465, FAX +42 2 290159
Email:fojtik@vision.felk.cvut.cz

I. Introduction

This paper decribes conversion of raster images into vector representation.
Processing consists from these steps:

- Image
- Edge points
- Node points
- Vector representation

II. Constraint on input data

The algorithm is designed for vectorisation of edges of adjacent domains. The input picture is preclassified aerial image by neural network. There are homogenous domains with same labelling (which is represented by a color). Each label represents crucial attribute for example wood, lake etc. The boundaryes of the domains are converted into vector representation.

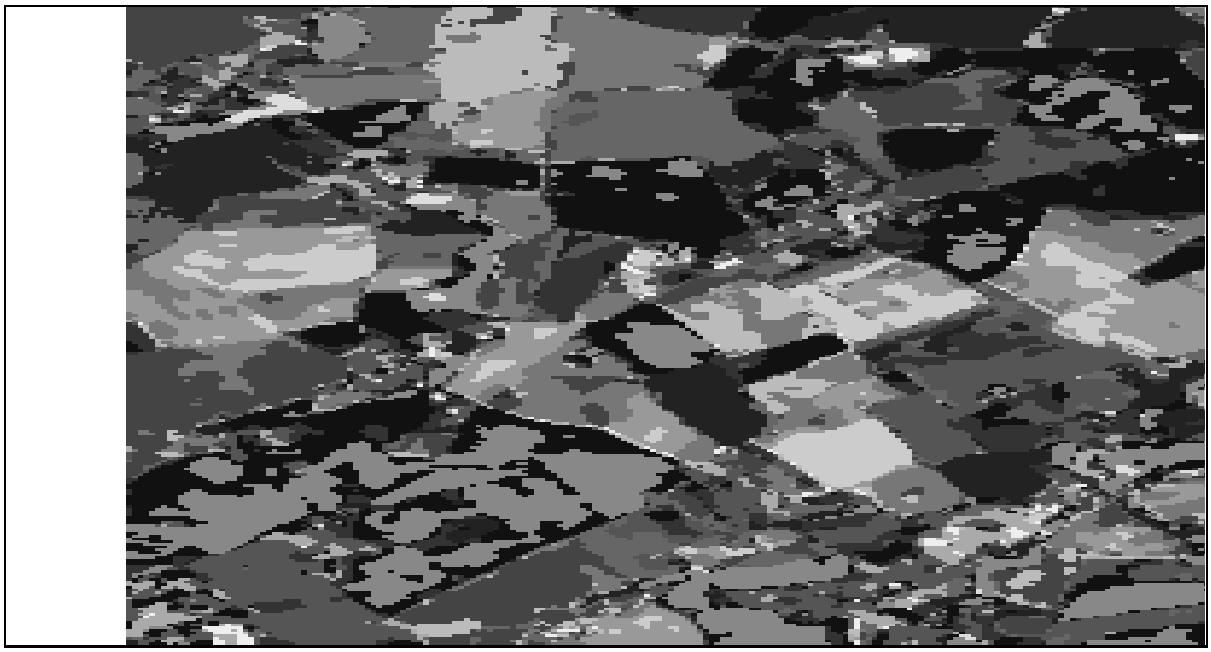


Figure 1: Example of input image

This algorithm isn't designed for finding edges in gray-scaled image. For this it must be first preprocessed. After preprocessing can algorithm continue from 2nd step see below. For preprocessing any edge detector can be used eg Canny.

III. Previous methods of area vectorisation

The whole image is traversed using operator, which tests equality of values two adjacent pixels. Equality is tested in all 8 neighborhood directions of actual point (may be 4 neighborhood too). If inequality occurs in one or some directions together, then output pixel is set to 1. Otherwise it is set to 0. The order of operations is shown in following images.

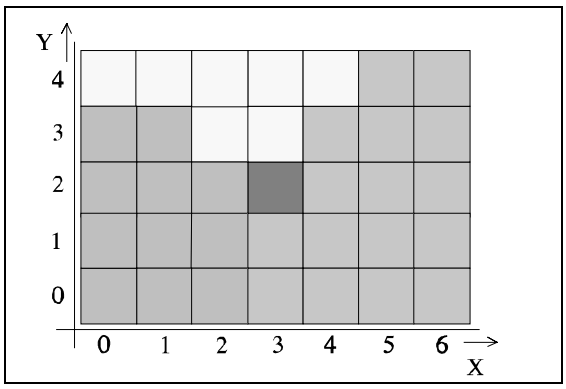


Figure 2: The original bitmap.

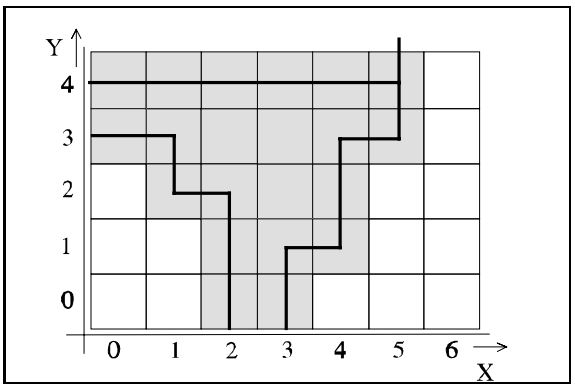


Figure 3: The bitmap after vectorization.

This process has essential disadvantage to double response for each edge. Of course there exists improved variants which tests equality only in some directions e.g. only from right to left and from up to down. This way is able to partially compensate double response but not absolutely. Problems approves around occurrence single point which lays on any other edge. Moreover all edges will be shifted in some direction.

Previous problems are caused by incorrect representation of boundaries. This representation was used in oldest TopoL releases. Therefore results of topology of finite spaces are used [Kovalevski].

IV. The description of abstract celuar complexes

The original bitmap isn't too advisable for representation of boundaries. Therefore the new method was used, which guarantee much better quality of detected edges.

Pixels are added to the original bitmap at corners and on edges of original pixels. These new pixels have special properties and are used for expressing of topological relations between image data. Procedure of extension is graphically shown on series of figures Figure 4,Figure 5,Figure 3.

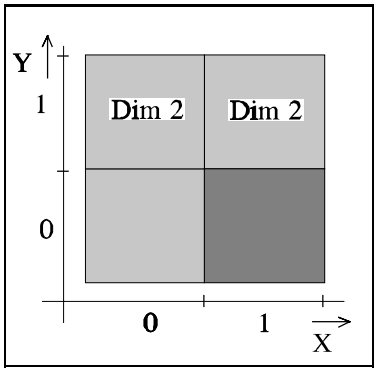


Figure 4 Original image

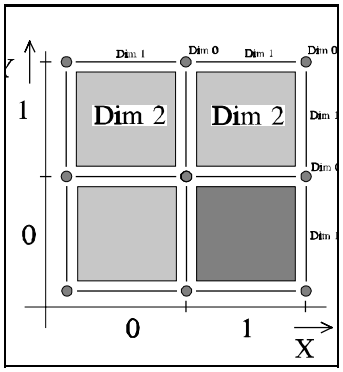


Figure 6 Celular complexes

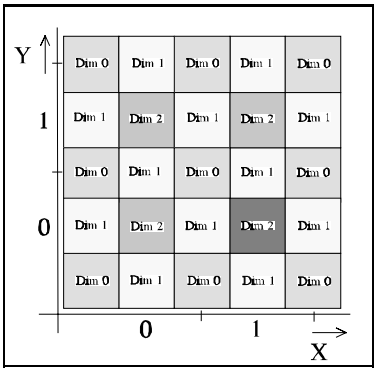


Figure 5 New representation

New pixels are added into original bitmap. These pixels describes edges of the pixels and corner points. The paradox of two lines doesn't arise in this configuration [see Počítačové Vidění page:30].

Original pixels are preserved and they represents elements with dim (dimension) 2. Edges of two pixels have dim 1 and elements in corners have dim 0.

V. Algorithmic aspects of vektorization

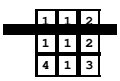
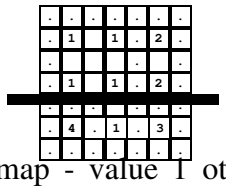
The vectorizator has 5 main passes. In first one there are processed image data only in raster representation, in 3rd it is constructed vector representation and last two works only with vector representation:

VI. Description of passes

The 1st Pass - **finding edges**

Edges of domains are finded in this pass. It is constructed new auxiliary binary raster KDT.RAS with sizes [2*x+1;2*y+1]. Its borders are set into value 1. Inside points are set only if condition, that lies between two points with different level of brightness. It is evaluated differences in longitudal, lateral and diagonal directions.

Following two tables shows this process of bitmap transformation:

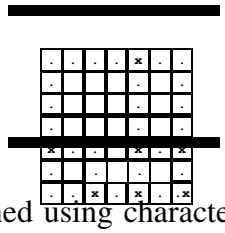
Original image  is transformated to 

(Only dot signed '.' points are stored in bitmap - value 1 other points has value 0. The numbers are inserted only for highlighting the correspondence with the previous image)

The 2nd Pass - **finding of nodes**

Nodes are detected in this pass. Second auxiliary binary raster of nodes KDU.RAS with sizes [2*x+1;2*y+1] is constructed. A node is such point, which has more than two nonzero neighboring points. For node detection is used binary operators similar of that used for dilatation.

In so far that our image is transformed to:



(In bitmap there are stored only points signed using character 'x' - value 1 other points has value 0. The dot points are inserted only for highlighting the correspondence with the previous image)

The 3rd pass - **detection of the vector lines**

In this pass are accounted vector lines. Each vector line must rise from any node and lead into any (may be other) node. Next 3 auxiliary files BODY.BIN, UZLY.BIN a LINIE.BIN are constructed. Checking of lines runs following way: The bitmap is sequentially traversed as late as first node is found. From this node is tried to go out (trace lines) to all four directions. In process of tracing line are found neighboring and these are signed into internal structure. In this process are points by sequel erased from auxiliary. Process end when node point is found. Then line is represented by list of vertexes. The count of vertexes is optimized by special algorithm.

Previous algorithm isn't abbe to find closed circles without nodes. Thus 3rd pass is executed twice. In 2nd process is any finded point considered as node (with rank 2) and all circle is founded (and erased) by sequential pass. (All other was already erased from raster KDT.RAS !)

At the foot of this pass are in future unneeded rasters KDU.RAS a KDT.RAS erased.

The 4th Pass - **checking areas**

From each node point is searched closed areas. Progress of searching begins in node, it continue through the line and next node point. After this the progress continues clockwise until beginning node is reached. From specification (in respect of previous steps) follows, that graph must be planar. Above specified procedure must every time generate closed circle.

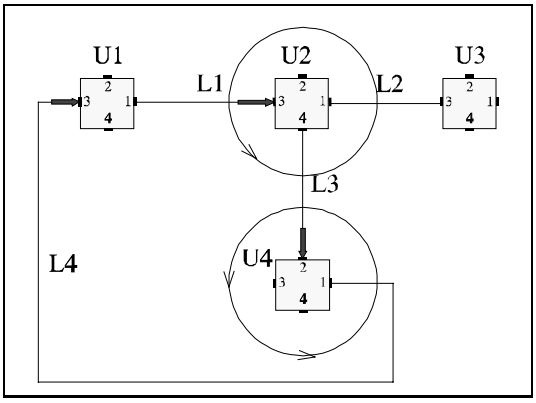


Figure 7 Scanning areas

Example of scanning progress:

Let U1 is a start point and let socket 1 points to the valid line. Searching continues along the line L1 to the node U2. Input direction in U2 is realized through socket No 3. First clockwise found socket with line has No 4. Scanning goes out from node U2 along line L3 into next node U4. Input socket in U4 has No2. First clockwise found socket with line has No 1. Output line L4 returns to start node.

When going around the area its size is counted. If is greater then limit, then all lines belonged to mentioned area are marked as valid (flag is placed in node points).

In node points are also stored all informations about progress. This ensure only one scanning of each area.

The 5th Pass - **building topological structure**

All nodes are checked. In each node are hitherto unerased lines joined together.

Last lines can be glued because of its neighborhood lines was already erased in the previous pass and degree of node decreases into value 2. Each node with degree 2 (in each circle must one left) is found and erased from graph. When the lines are created the areas are generated too.

At the end of this pass are in future unneeded files LINIE.BIN, BODY.BIN and UZLY.BIN erased.

VII. Directions for users

Program have at all 5 parameters:

a, Name of the input file.
b, Name of the output block.
c, Minimal size of the area.
d, Deviation vector line from raster.
e, Switch of the border areas.

a, Input image may by in any type 2,16,256 brightness levels (with or without palette), DMT or true color. But must contain continuous domains with equal value.

b, The name of output block specifies the path (of block directory), where output data will be stored. Destination device must have enough free space for storing temporary data.

c, Minimal size of the area is measured in pixels (if somebody wish may be converted into Ha). If it is too small <1,10>, then the size of output vector image extremely grow.

d, Deviation of vector line indicate even of vector line and this parameter schedule a number of removed redundant vertexes. Unit of its value is point/point of line. The values from range 0.1 to 0.9 are recommended. If this value is too small, then output lines are not smooth. If this value is too big, then the shape of areas are deformed.

e, If the switch of border areas is setted on, then in output image will be areas which touches border of the image. In another care are areas mentioned above ignored and removed from output file.

VIII. Why is output file such long?

Let us have chessboard its size of domain 8x8 (it means 64 points). This case is near worst.

Each domain will be on output one area. Each domain have 4 lines. Lines are shared with neighbor cells.

Length of line = 103 bytes + 2 outer points 24 bytes (4x6)
Length of area = 71 bytes

$$lpl = 4*(103+24)/2 + 71 = 325 \text{ bytes}$$

lpl count of bytes needed for one domain of chessboard in vector format. Original size of each domain in raster representation was 64 bytes (for 256 colors), or 32 byte for 16 colors. It mens that length of data increases 5 (or 10) time.

Low-costing of vector format (output data size) is bolded from minimal size of area

325 points, it is 18x18. It the size of area is 1 then the size of vector image may be **300x** bigger, than that of original raster. User must carefully weigh the price of details which he need have in output vector data (or have sufficient space on destination disk).

IX. Conclusion

In this paper described method of automatic vectorisation is able save much work to peoples, whom later must do manually the vectorisation of aerial images. This program is now able to process very large images about hundreds Mb. User must have enough free space on disk and spend some time with modification input parameters.

References:

Počítačové vidění Václav Hlaváč, Milan Šoňka Grada 1992