

Cadastral Map Analysis: Simple Methods, Good Results

H. Shahbazkia, T. Candeias, R. Oliveira and F. Tomaz
Universidade do Algarve - FCT
BIF laboratory, Campus de Gambelas, Faro, Portugal
E-mail: {hshah, tcandeia, rsolivei, ftomaz}@ualg.pt

Abstract – This paper introduces the project ACID (Automatic Cadastral Information Digitalization), which started in February 2000 and which is financed by the Portuguese FCT (Fundação para a Ciência e Tecnologia). The main goal is to develop a system for automatic digitalization of Portuguese cadastral maps by using simple image processing algorithms. This paper presents the goals of the project as well as first results obtained.

Keywords – Cadastral Information System, Map Analysis, Image Processing

I. INTRODUCTION

ACID is the acronym of the project *Automatic Cadastral Information Digitalization*. The word *Digitalization* refers to a transfer of the visual information from a paper support to a GIS (*Geographical Information System*).

Every city council in Portugal has to manage its cadastral information. This information is usually stored on paper support, like the one showed in Fig. 1.

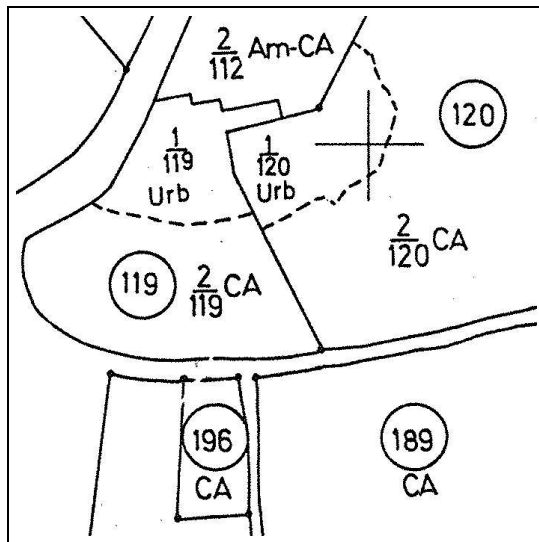


Figure 1 - Sample of a cadastral map (690x690, 300 dpi).

In the past years many administrative entities have decided to transfer the cadastral information to a numeric format and started using electronic management systems. Only 15% of the cost of a digital CIS (Cadastral Information System) goes to the hardware, 5% to the software and 80% to the manual data acquisition from the paper maps. Beside, the acquisition phase is not only expensive, but also

time consuming (8 hours for an urban cadastral sheet) and the quantity of data to be transferred is huge (more than 100 000 sheets in Portugal).

Some work was already done analyzing cadastral maps in other countries [1], but due to the formal aspects of representation different approaches must be considered.

As a strategy the authors have opted for a heavy use of basic and well-known algorithms [2] with some eventual modifications. The first results are encouraging and about 80% of the information can be already extracted from the map.

II. PROCESS ORGANIZATION

Any Portuguese cadastral entity is composed by a closed contour, a numeric identification inserted in each parcel circle, possible existence of depended plots (small parcels), separation lines between parcels and finally the description of each parcel (Fig. 1). A single map contains many of these entities together with a meta-knowledge sector in which extra information about the map, normally modifications, are presented.

The authors have used all these information from the start to establish the global strategy of the analysis. For example, circles can provide an excellent departure point for analysis an entity from inside. Figure 2 explains how the process can be guided by general knowledge of the domain.

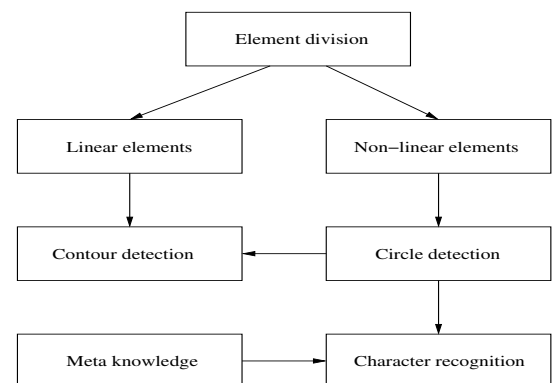


Figure 2 - Analysis process flow

The division between linear and non linear elements is made analyzing the occupied area. This knowledge permits the separated application of recognition algorithms. Recognized elements are marked to avoid interfering with another layer of analysis, and to resolve possible conflicts, making the overall process more reliable.

III. EXAMPLES OF ANALYSIS

Some important work was done for the analysis of the cadastral information. This include circle, semi-circle, character recognition and contour detection. The following sections include some important remarks for each solution implemented.

The use of the methods for this analysis are mainly conditioned by the enormous computational effort necessary to complete the task, due to the quantity of information¹.

The legal status of cadastral administration imposes full robustness therefore the applied methods are largely known and tested, but are also necessarily fast and accurate.

A. Circles and semi-circles

Some problems are associated with circle detection in a common cadastral map: the existence of semi-circles (fig. 3), multi scaling (fig. 4) and connection with elements defined as linear (see section II).

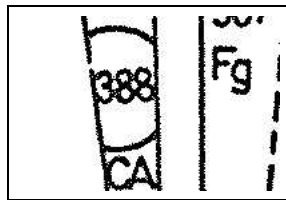


Figure 3 - Semi-circle, connection with linear object

Hough transform [3], [4] is a know process for extraction of parametric defined shapes, but the required parametric space is very large, making it's use unwise. We use a simple method based on a modified progressive probabilist HT [5], to cover the recognition of semi-circles and with local decision. This prevents the necessity of large storing and searching, but results in a strong circle mismatch recognition near a "true" circle. This problem is easily resolved with a post-treatment searching and confirmation algorithm based on known characteristics.

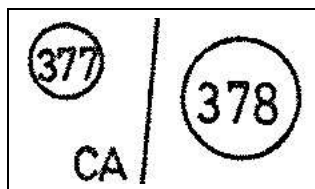


Figure 4 - Multi scaling

From the division of elements presented in section II, the detection of circular elements over the non-linear class is very reliable only requiring a few points of analysis². For the recognition of circular objects over the linear elements (connected circles and semi-circles), more points were used (35 points for 300 dpi), and in case of a a possible semi-circle, a validation algorithm is used, checking it's interior and boundary. Table I presents some results for this

¹Each map have around 500 entity's, 800 parcels, 2500 characters and a significant number of miscellaneous information.

²high performance with accurate results were obtain for 15 points for a 300 dpi image

methodology, with post-treatment analysis, that we consider very encouraging.

Shape	n in image	correct hits	incorrect hits
Circle	294	89 %	0.013 %
Semi-Circle	84	81 %	0.038 %

Table I

RESULTS OBTAINED ANALYZING A REAL MAP WITH 300 DPI

B. Optical Character recognition

Optical character recognition (OCR) is one of the most widely used application of automatic pattern recognition and it's a very active research field since the 50's. Today we are able to recognize high quality text documents or especially written hand printed text.

Efforts are being made to achieve better recognition rate of degraded printed text and unconstrained handwritten text, albeit the recognition speed become slower.

OCR has, independent of the source of the characters, some steps that are common to most projects of investigation like preprocessing the text to separate touching characters, a method to extract features from the char to be classified, and in the end there is a post-processing for the validation of the resulting recognition.

Additionally optical character recognition has problems with characters that are incomplete due to noise.

The segmentation problem of touching chars were solved by using projection histograms that try to detect minima in the middle from the horizontal histogram and maxima from the vertical histogram. This is done because most of the touching chars are the parcels fraction number that could be observed in Fig. 5

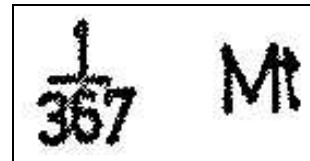


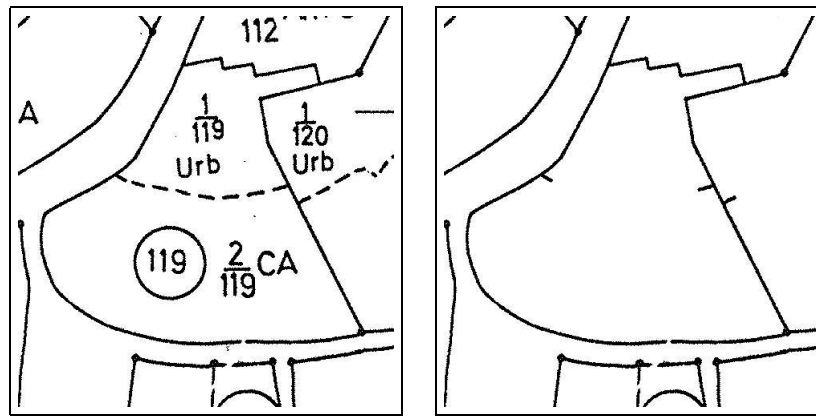
Figure 5 - Connected character's problems

After analyzing different feature extraction methods [6] like: template matching, deformable templates, unitary image transforms, graph description, projection histograms, zoning, geometric moment invariants, spline curve approximation, Fourier descriptors, we chosen the template matching because of it's simplicity.

Instead of using all the template points, only important points were selected. Later weights (+,-) were attributed to these points acting as confirmation or rejection points [7].

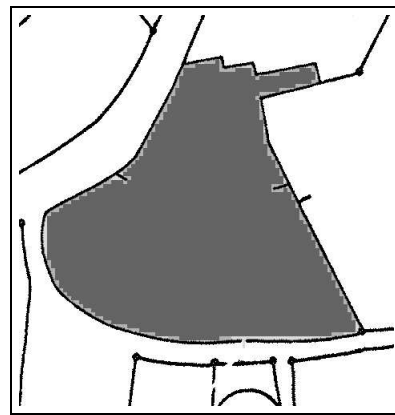
This information was stored in a list with the relative point positions and the associated weights, Fig. 7 shows two examples for the list point's position. From this two examples, it can be seen that the weights should be carefully chosen, because important differences should be highlighted.

For the positive weights a skeletonization of the character was done, and equidistant points were chosen. For the negative weights two different dilations were done, and the

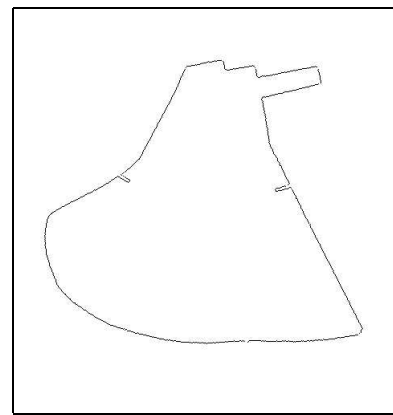


(a) Original image

(b) Processed image



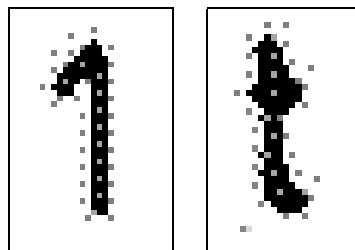
(c) Brush algorithm



(d) Final result

Figure 6 - Contour detected using Brush algorithm

difference between the bigger and the smaller were stored. Similarly we selected equidistant points and stored them in a list with the respective negative weights.



(a)

(b)

Figure 7 - Example from the determination of weight position.

The initial method used to attribute the weights was a heuristic knowledge of representative points from each character. To solve this cumbersome manual method a genetic algorithm was introduced that gives the optimal weight for each character. This algorithm uses a fitness function that count the number of true or false recognitions in a database with multiple examples of every possible char-

acters.

This method, although satisfactory, have some problems related with the segmentation of connected characters with histograms.

Although a list is being used, we are planning to use a neural network to compare the results and speed with our current implementation.

C. Contour detection

Contour detection is an important task in automatic recognition of maps, mainly because it gives the coordinates of points that constitute a parcel in a GIS.

This detection is based on the knowledge provided by the circle position already detected, as refereed in section II. As every parcel contains a circle inside this information is reliable.

Before applying the contour detection, the map should be pre-processed. This mean that non-linear elements should be eliminated.

The main problem in contour detection is the existence of discontinuities. Two strategies are possible: restore discontinuities or use algorithms that aren't sensible to it. The last one was chosen because is a more robust method.

```

DetectContours (Image, Circles, N) {
  for i=1:size(Circles),
    CritPoints=Fill (Image, Circles(i).x, Circles(i).y, N)
    Contours(i)=BrushAlgorithm (Image, N, CritPoints)
  return (Contours)
}

Fill (Image, Circle, N) {
  if (EmptySquare (Image, x, y, N))
    FillSquare (Image, x, y, N, GrayColorSquare)
    Fill (Image, x-N, y, N)
    Fill (Image, x+N, y, N)
    Fill (Image, x, y-N, N)
    Fill (Image, x, y+N, N)
  else
    CriticalPoints.add(x, y)
  return (CriticalPoints)
}

BrushAlgorithm (Image, N, CritPoints) {
  if (GraySquareLeft)
    for y=1:N,
      for x=1:2N, // Scroll
        if (!LockX)
          if (Black)
            Contour.add(x, y)
            Paint (x, y, LockColor)
            LockX=True
          else
            if (LockColor)
              LockX=True
            LockX=False // Restart Lock locally
          LockX=False // Restart Lock globally
          // Continue to bottom, right and top ...
        return (Contour)
      }
}

```

Figure 8 - Contour detection pseudo-code

Initially it was pretended to do this analysis using active contour model also known as Snakes [8], but due to a lack of different energies fields, it wasn't appropriated to apply.

It was chosen, due to this specific problems, the creation of a innovative algorithm (see figure 8), with no sensibility to discontinuities and with the initial knowledge of a point inside the contour. Fill algorithm is a valid process but it's sensible to discontinuities in contour, so it was reformulated.

Instead of using a normal fill it was used a fill with a variable block size. In this way, using recursivity blocks that are part of contour are identified and then processed locally, this areas are called *critical squares*. To locally process this critical zone it's considered the squares neighbourhood, so it should be denied fill's recursivity because squares aren't filled sequentially. To do that a list of critical squares is made and later, all filled neighbors are processed.

When the neighbors have filled squares, the process of contour detection starts. This method is called *Brush* method because it's like a paint brush. After a detection with a contour point, a match happen, and it's axis is locked.

Effectively contour edges are obtained, so the detection of edges should be made from all four directions.

Figure 6, is a representative example of the presented algorithm, in (a) the original image is presented. After processed figure (b) was obtained, in figure (c) the Brush algorithm was applied. The result of contour detection, is represented in (d).

To perform a automatic contour detection, the size of an generic block (N) must be determinated. This can be done applying a fill in each circle and then a search is made for

the smallest block size that satisfy the non existence of any empty contour. This happens because if it's made a fill in one parcel, that goes outside it's contour, then other parcel gets a empty contour.

IV. CONCLUSIONS

Most of the problems presented, and their handling are known. Our general methodology is based on the application of these known methods, with some modifications based on the meta-knowledge of the map construction rules. This provides not only a robust but also a fast analysis due to the known element's characteristics. An exception using an innovative algorithm is presented in detection of contours, being both fast and efficient.

Results are encouraging, and correct detection reaches about 80%. The remaning information must be treated using other methods like artificial intelligence and specific algorithms to this task, should be create.

There are heuristics in this process that could be used to enhance the results even further. All the map construction rules could be used to an expert system, implemented in a forward, backward or a mixed chain that could confirm results. A Black Board system or multi-agent could also be used.

Some examples of construction rules are: inside each circle there can be only numbers, there is a fixed number of different words occurrency like: Am, Ca, Fg with different meanings. So a dicionary database can be created with these rules.

V. ACKNOWLEDGMENTS

This work is funded by the FCT (Fundação para a Ciência e Tecnologia), project ACID, contract SRI/34257/99-00 - Automatic Cadastral Information Digitalization.

REFERENCES

- [1] H. Shahbazkia, *Reconnaissance invariante et acquisition de connaissance: application au traitement automatique des plans de cadastre français*, PhD thesis, Universit Louis Pasteur de Strasbourg, 1998.
- [2] K. Tomber, C. Ah-Soon, Ph. Dosch, A. Habed and G. Masini, "Stable, Robust and Off-the-Shelf Methods for Graphics Recognition", in *Proceedings of the 14th International Conference on Pattern Recognition, Brisbane (Australia)*, pp. 406-408, 1998.
- [3] P. V. C. Hough, "Method and means for recognizing complex patterns", *US Patents 3069654*, 1962.
- [4] D. H. Ballard, "Generalizing the hough transform to detect arbitrary shapes", vol. 13, no. 2, pp. 111-122, 1981.
- [5] C. Galambos J. Matas and J. Kittler, "Progressive probabilistic hough transform", Tech. Rep., University of Surrey/Czech Technical University, 1998.
- [6] O. Trier, A. Jain, and T. Taxt, "Feature extraction methods for character recognition - a survey", 1996.
- [7] P. Gader, B. Forester, M. Ganzberger, A. Gillies, B. Mitchell, M. Whalen, and T. Yocum, "Recognition of handwritten digits using template and model matching", *Pattern Recognition*, vol. 24, pp. 421-431, 1991.
- [8] M. Kass, A. Witkin, and D. Terzopoulos, "Snakes: Active contour models", in *Proc. of IEEE Conference on Computer Vision*, London, England, 8-11 1987, pp. 259-268.