AUTOMATIC OBJECT RECOGNITION IN LINE MAPS

Tiina Kilpeläinen
Department of Photogrammetry
The Royal Institute of Photogrammetry
S-100 44 Stockholm, SWEDEN
Commission III/IV

0. ABSTRACT

Raster scanning is an automatic digitising method used in data extraction from line maps in large scales. This paper deals with the problems in automatic recognition of complicated objects based on vectorized map data.

In order to treat the problem of automatic object recognition systematically, the cartographic language must be strictly formalized. The first part of this paper describes a theoretical model for large scale maps. The model is tested with objects "building". A manual test is made to recognize all the buildings in map sheets in scale 1:2000 with the rules, composed according to the model. The result gave 100% of buildings classified with 1-9% errors of second order depending on the type of map information.

The second part of this paper discusses the problems when trying to implement the presented rules in Turbo Prolog running on IBM PC. The main problem lies on - not the difficulty on programming the rules- but the difficulty to update all the possible situations and to organize input data.

1. INTRODUCTION

A lot of efforts have been involved in automatic recognititon of symbols. The classical way to perfome symbol recognition has been training the system for the classification task under consideration and finding the relevant probability density function for each symbol class. Norwegian Computing Center reports results with up to 99% correct classified handwritten letters, (Holbaek-Hanssen et al.,1986). Weber points out that there are several companies which offer software for symbol recognition without giving details about the underlying method and results (Weber, 1987). It seems quite clear anyhow that recognition of numbers, letters and point symbols have a rather long and successful tradition, while recognition of map objects based on vectorized data still lies on its early ages.

De Simone reports a success rate of 90% with 1% error in automatic recognition of railways, roads and landparcels, (De Simone, 1986). He uses the combination of the geometrical/ statistical and the relational method for recognition, (Weber, 1987). The check of neighborhood relation of the reclassified candidates decides if these are refused or accepted into the class in question (e.g., the adjacent polygons of a preclassified railway have to be narrow and long). The prior results are taken into account in classification of the next object type by executing the recognition process in a fixed sequence (first railways, then roads and finally land parcels).

The experiments made on object recognition show the need of strict problem specifications. To make use of the knowledge of the neighborhood relations in the best way, the relations must be specified. Before the relations can be specified, the object specifications are required. In the next section a model for descriptions of the objects in map is presented. The methods used for object recognition had their basis on computer science. Sooner or later observations made by mapreaders must be included in the process to make recognition consistent (e.g. relationships). My approach is to include perceptual considerations in the model at the beginning and try to imitate mapreading in the model as far as possible. Then the computer abilities and capacities come into play - the problems are pointed out.

2. THE THEORETICAL MODELL

2.1 Terminology

In this study a symbol describes a geometric figure; the symbol's size is not related to the figure's size and shape in reality. Some examples of symbols are the symbols for trees and boundary marks.

An object represents a geographical unit, the size of which in a map is related to the extension of the object in reality and depends on the mapscale. Some examples of objects are buildings, streets and real estate boundaries. The problems treated in this work refer to object recognition.

2.2 Perceptual background

A map shows a simplified and abstract picture of reality. A mapreader interpretates the map with the aid of visual variables and combinations of these. Some of the most important visual variables are dimension, colour, form, type of lines and raster (Baudouin et al., 1984). Reading large scale maps the visual signals received consist on one hand of visual variables like dimension, form and type of line, and on the other hand of differences between successive observations, e.g. contrasts, discontinuities, irregularities in object's outlines and object's relationships to the surroundings. Objects and their relationships to other objects in the map are transformed to corresponding relationships and objects in reality. Therefore the object's surroundings and relationships to other objects (e.g. dimension, distance, direction) have to be taken into account when trying to imitate the object classification a human being does.

The received visual signals are broken down into smaller and smaller elements, and then categorized and classified by the brain (Keates, 1982). Perception of large scale objects presumes finding the characteristic elements of the object and categorizing them.

There are different levels of perception. To find differences between objects or to discover objects doesn't demand understanding or earlier experiences of the meaning of the object. Identifying objects in contrast is a learning process which assumes understanding of the meaning of the object.

How the interpretation of maps takes place depends on the level of perception. Detail observations are mainly concentrated on the object searched for and it's imediate surroundings, while overview observations often use partitioning of the map into parts with the aid of streets, for instance.

2.3 Definitions

The concepts used here are based on Goldkuhls work (Goldkuhl et al, 1984) with some modifications. He uses the means of information theories to identify information quantities and information processes in an organization.

A map can be considered as an _information quantity_ consisting of _messages_ in several levels. The objects in maps (e.g. parks, streets and buildings) represent messages. The messages have certain relationship to each other, e.g. a park lies 500 m from a house. A _relationship_ is thus a condition which connects the different messages. Some examples of relationships are distance, angle, form and dimension. We have the equation for information quantity:

$$\text{INFORMATION QUANTITY} = \Sigma\ \text{MESSAGE} + \Sigma\ \text{RELATIONSHIP} \quad \text{(eq 1)}$$

Messages of different levels can be identified in a map. E.g. a map can include several blocks, which in turn consist of several real estates. A real estate consists of a certain line style for the boundary and an identifier for the real estate in question. The messages of first level are _elementary messages_. An elementary message is a message which can not be broken down into smaller elements. Some examples of elementary messages are lines with certain linestyle and - width, symbols, strings of letters and numbers. We have an equation for a message:

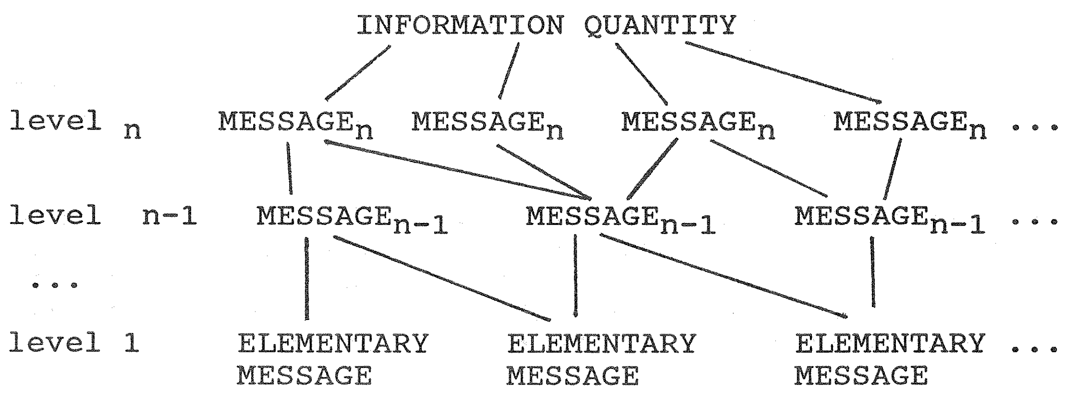
$$\text{MESSAGE}_n = \Sigma\ \text{MESSAGE}_{n-1} + \Sigma\ \text{RELATIONSHIP}_n \quad \text{(eq2)}$$

where n is the level of the message.

An equation for the messages of second level can be written as:

$$\text{MESSAGE}_2 = \Sigma\ \text{ELEMENTARY MESSAGE} + \Sigma\ \text{RELATIONSHIP} \quad \text{(eq3)}$$

A theoretical model for mapinformation is illustrated in figure 1.



Figur 1. ——— = relationship
An information quantity is built up by messages and relationships in different levels.

Both the relationships and the elementary messages can have different values. E.g. a line can have values linewidth 0.25 mm and linestyle dashed.

3. PROCEDURE FOR THE MANUAL EXPERIMENT

3.1 Assumptions

The natural way to build up the hierarchy for the model is from elementary messages up to messages of higher level. Therefore I assume that the mapreading takes place from detailed observations to an overview. I work only on level 1 and 2 and thus do not consider overview observations presently.

Here I limit the experiment on vectorized large scale mapinformation. I assume the following information is already recognized based on section 1.:

- vectors(start- and endpoints, linestyle(width,style))
- characters and strings of characters
- symbols
- arcs

The objects on the edges of the mapsheets are not considered in this experiment, neither contour lines.

3.2 Method description

In the experiment rules were made with different degrees of consistency to describe the object "building" according to the presented model. The basis for building up the rules is formed by perceptual observations. Some detailed conditions have their origin in (the Handbook for Detailsurveying from the city of Stockholm, 1978), e.g. the linewidth for the outlines of houses. The rules were then matched manually with all the buildings on three different base mapsheets in scale 1:2000.

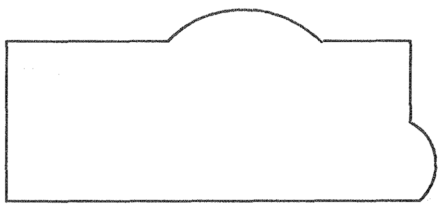
The success rate was determined for the different rules. I tried to match only and only according to the given rule. In the next step the rule was improved with additional conditions, which took into account perceptual matters.

Logical operators (IF,AND,OR,NOT), aritmetical operators (<,>,=,..) and aritmetical expressions are not separated in the rules.

3.3 The rules

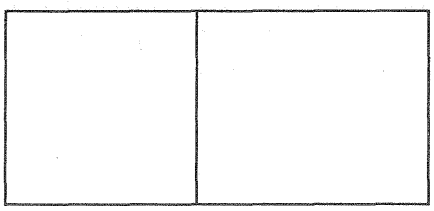
I will here describe only the rule of the highest complexity, because it includes all the rules of lower complexity. I will first write down the rule in English and then separate it in table 1 into elementary messages and relations according to the model.

A building is a closed polygon with solid lines with linewidth 0.25mm. It is not required that it has rectangular corners as shown in figur 2. Besides straight lines, a building can consist of arcs.



Figur 2.
A building can consist of straight lines and arcs as well.

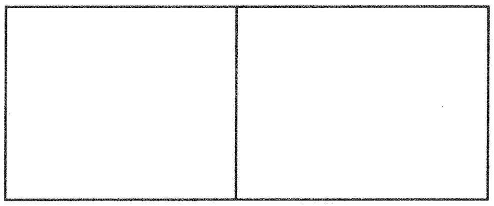
It is not possible to devide the polygon that forms the outline of the building into smaller parts with other lines. Figur 3 can be read in different ways without this restriction. It can be read as one, two or three separate polygons.



Figur 3.
Is this one, two or three polygons?

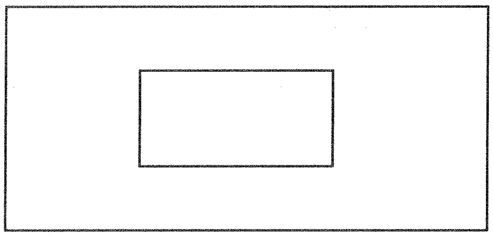
It may occur that polygons are connected at their corners to the symbols for boundary marks. This is the case when buildings are lying on the boundaries of real estates in connection to the streets.

None of the outlines of a building should be connected with the symbol for wall, fence etc. This condition avoids symbols for fences and walls to be mixed with a building, figur 4.



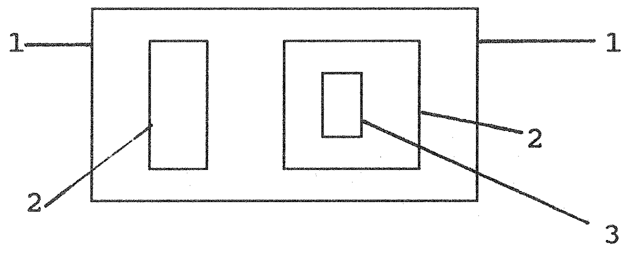
Figur 4.
Symbols for fences can be mixed with the outlines of buildings.

Figur 5 illustrates a yard enclosed by the body of the building.



Figur 5.
A yard surrouded by a building.

In this case we have to define what's a building and what's a yard. Figur 6 illustrates a system for numbering the polygons in such a way that it is possible to separate buildings from yards.



Figur 6.
Numbering the polygons illustrating buildings and yards.

Now we can say that if there are some polygons inside the building, the polygon which forms the outlines for the building has to have an odd number of order.

The reasoning above is summarized in table 1. Separation of the rule into elementary messages and relations is made according to the presented model.

Object	Elementary messages	Relations
building	e1=line (solid)	r1=polygon(closed)
	e2=symbol for boundary marks	r2=possible occur
	e3=symbol for fence	r3=(e2) in connection to (e1)
	e4=symbol for wall	r3=(not) (e3)/(e4) in connection to (r4)
	e5=number of order	r4=(e1) of (r1)

Table 1. Description for a building according to the model. e=elementary message, r=relationship, ()=value.

It is possible to separate between buildings of different size. E.g. if it is not of interest to classify garages as separate buildings we can take them away and say that a building must have an area (r) greater than 25 m^2, for instance. A transformer station is one special case of building. If we want to separate them, an additional condition can be given: the symbol for transformer (e) inside (r) a building. However, these examples are not considered in this test.

3.4 The results

Table 2 shows the result from the test above.

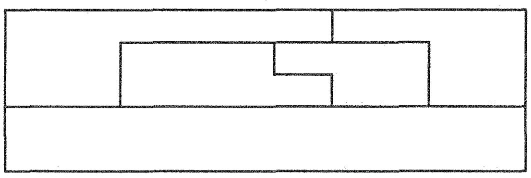
Errors of the first order mean buildings which have not been classified. Errors of the second order mean the objects which have been classified as buildings though they are not buildings.

mapsheet number	correct classified buildings	number of objects classified as buildings	errors of 1. order	errors of 2. order	tot number of buildings
37	302	305	-	3	302
47	460	506	-	46	460
107	1087	1087	-	-	1087

Table 2. The test of the rule for a building.

All the real buildings were classified as buildings independent of the degree of complexity of the rule. There are still objects, which are not buildings and despite of that have been classified as buildings. The results vary depending on the type of map. One result is, which is not shown here, that the result is better when including more and more complex conditions in the rules.

The errors still remaining were firstly the case where a boundary of a block coincides with the outline of a building at a street corner. It can be possible to avoid this error after the object block has been defined. The main part of the errors depends on the case in which a yard was surrounded by several buildings, figur 7.


Figur 7.
A yard surrounded by several buildings.

In the case of a yard surrounded by several buildings it may be impossible even for a mapreader to decide whether it is a yard or a building. Some additional knowledge e.g. field recognition, have to be got before you can be sure of making the right conclusion. Anyway this case of a yard is not of great importance for line map applications. In raster applications where entitites of surfaces are considered, it is necessary to separate even these objects, e.g. setting different colours on yards and buildings.

4 PROGRAMMING THE RULES

4.1 Considerations on programming language

The test shows that it is possible to make strict object descriptions on map objects according to the presented model. One can find it time consuming and quite difficult in some cases when a lot of special cases have to be taken into account. It is necessary anyhow to get specific object descriptions before the problem of automatic object recognition can be treated systematically. How could a machine solve problems when it is not told WHAT the problems are?

It is just this declarative meaning that results in requirements for the programming language to be used when programming this type of rules. The traditional programming languages work on procedural level. They are told HOW to solve the problem. The feature of relational data provides the basis for solving the problem. Forbes (Forbes, 1985) examines Prolog, used in artificial intelligence applications, as a programming language for relational models for high level manipulations in the development of geographical information systems. He raises some problems for which Prolog seems to offer a valuable tool which is worth investigating. One of the questions he considers for those who are interested in experimenting is: "What is a minimal set of relations necessary to completely specify the topology and attribute structure of a map? What clauses ("virtual relations") could be additionally defined in terms of these?"

This discussion serves as the basis for the second part of this paper, which decribes some preliminary results on programming the rules in Turbo Prolog, running on IBM PC.

4.2 Input data and scope

Vectorized raster scanned data was obtained from the Swedish company VBB. The data consists of a base mapsheet in scale 1:400 and was preprocessed by VBB. Symbols have been recognised using software from SysScan. That means in practice that almost all the elementary messages are already recognised. The objects "buildings" are not so complicated on this mapsheet as they are in the manual test.

The most important relationship to be programmed is a "polygon" with the values (closed) and (open). Then relations "in connection to" and "inside" come. The system for numbering the polygons of different order requires investigations of methods which can treat this problem.

4.3 Programming in Turbo Prolog

Input data was transformed to predicates, facts that build up a sort of database. The first program finds all linepairs having a common point together. Only lines within a given widthinterval are considered. In the following programs it is only the linepairs which are operated upon and the facts about lines are retouched when necessary. A sorting program orders the linecombinations into a sorted list. The program "allpoly" finds first a cluster of polygons, i.e. all the lines having some common point in a network. Among these clusters all the separate polygons, both closed and open, are picked up in recursive predicates operating on lists. The program "close" finds all the closed polygons among these polygons in a temporary database, where it is checked that each polygonside is listed twice.

4.4 Problems in implementing

The decision to transform input data to predicates results in a restriction that allows only 400 lines of a certain linewidth to be processed at a time due to limitations of the Turbo Prolog compiler. It may be possible to find some other way for this particular processing, but at the same time, the linepredicates offer the ability to be operated as a database which has several advantages. Restrictions of program sizes is a well known problem even in other PC applications. One way to attack the problem is to find the areas of interest. This is a similar problem to the problem of mapsheet edges. The second way is to point out the problem and say more powerful computer and computer program capabilities have to be used.

One problem is due to the structure that uses several levels of recursion. Prolog uses backtracking in problem solving, i.e. all the alternative proofs of a given clause may be obtained. There are techniques, called "cut" to avoid backtracking when only one answer is of interest. I have encountered problems with the placement of "cut" when working in high level of recursive predicates. The work is still going on when writing this paper. The procedure of operating lists in tail recursion seems to be a very suitable and "natural" way of finding all the possible polygons, but it requires a large amount of stack space. However it is likely that the existing Prolog versions for PC are not yet powerful enough for this kind of problem solving.

One problem encountered is due to the scanning techniques used. The rules presented in this paper use linewidth as a descriptive factor. The input data consists of remarkably varying linewidth within one polygon. On the other hand this is more a problem of scanning techniques and not a problem of the here presented method.

5 SUMMARY AND CONCLUSIONS

If the problem of automatic object recognition is treated systematically it is necessary to find models which can be used to describe map objects strictly. In this paper one alternative is presented, which seems quite suitable for the task.

It seems that Prolog-programming is well suited in programming relational connectivities presented in this paper despite of the encountered implementation problems. However, the compiler limitations seem to be so severe that this kind of task can not be handled on PC computers for real applications. The technical experiments will continue.

6 REFERENCES

Baudouin, A. and Anker, P., (1984), Kartpersepsjon og EDB-assistert kartografi, publ.nr.744, Norwegian Computing Center, Oslo, Norway, 1984.

De Simone, M., (1986), Automatic structuring and feature recognition for large scale digital mapping: Proceedings Auto Carto London, vol. 1, 1986, pp. 86-95.

Forbes, R., (1985), Very high level relational languages and GIS: Toward the 5th generation: Technical papers 51st annual meeting ASP, vol.1,1985, pp.414-420.

Goldkuhl, G., Nilsson, A. and Röstlinger, A., Att specificera informationssystem på ett användarorienterat och systematiskt sätt, Department of adm. data processing, The University of Stockholm and the Royal Institute of Technology, 1981.

Handbook for Detailsurveying, cap. 4, Stockholms Stadsbyggnadskontor, Stadsmätningsavdelningen, Stockholm, 1978.

Holbaek-Hanssen, E., Bråten, K. and Taxt, T., (1986), A general software system for supervised statistical classification of symbols: Technical Report, Norwegian Computing Center, Oslo, Norway, 1986.

Keates, J. S., Understanding maps, Longman Group Limited, Essex, UK, 1982.

Weber, W., (1987), Cartographic pattern recognition, (1987), Report on International Research and Development by the Commission on Advanced Technology of ICA, 13th Conference in Morelia, Mexico, 1987.