copy 1

THE APPLICATION OF INTERACTIVE GRAPHICS AND PATTERN RECOGNITION TO THE REDUCTION OF MAP OUTLINES

by

Andrew Clement

B.Sc., University of British Columbia, 1968

A thesis submitted in partial fulfillment of

the requirements for the degree of

Master of Science

in the Department of

Computer Science

We accept this thesis as conforming to the

required standard

The University of British Columbia

March 1973

Abstract

Techniques from interactive graphics and pattern recognition are applied to the problem of reducing map outlines. Since the resulting generalized outlines are intended for use in interactive graphics systems their data content should be considerably less than that of the original lines. Also it is useful to have several levels of generalization for the same line and an extension of the X-Y coordinate encoding scheme is introduced to represent such hierarchically reduced lines. Experiments are conducted that suggest that people look at outlines in different ways. To accomodate these differences in taste and purpose the system is designed to adapt to the individual user's preferences. This is done by having the user reduce several outlines by hand. The system analyzes patterns in these lines and so learns to mimic the user's behaviour. Once enough has been learned the system is given new lines to generalize on its own. Experiments are performed to measure the learning ability and the generalization performance. Other experiments are performed to show the potential feasibility of this approach. There is a review of work done in related fields.

Table of Contents

List of Tables

List of Figures

Acknowledgements

Many people have helped me in my work and in the preparation of this thesis. To all these people I give my thanks. I am especially indebted to Frieder Nake, my advisor, whose patience and insight made it a pleasure to work with him and continued to sustain me for the last seven months that I was on my own. Thanks go also to Jim Kennedy for his careful reading and helpful comments on the rough draft and to Doug Seeley and Tom Peucker for their suggestions. I am also grateful to Inger Nisson for her typing of Chapter 3. There are many others, people who cheerfully submitted to be subjects in the various experiments, the people who wrote the software that I depended on heavily, and the taxpayers of Canada who through their institutions, the Department of Computer Science at U.B.C., and the Canada Council supported my work financially. I especially wish to thank my wife, not only for the typing and help with the diagrams, but also for her support throughout.

INTRODUCTION

The growing use of interactive graphic facilities for the display, manipulation, and interrogation of geographic information has created a demand for a more flexible and compact representation of map outline data. Whereas in traditional maps the accuracy and detail of outlines were considered important, in the case of interactive graphics the primary concerns often are the reduction of storage requirements over a range of scale dependent on levels of resolution, while enough detail is maintained for visual identification. The concern with minimizing the storage requirements stems from the usually severely limited memory capacities of CRT[1] display devices. In addition the processing times for typical operations such as shading and intersecting regions often vary as the square of the number of points along outlines. Due to the limited screen sizes available at the present it is also necessary to change radically the scale of the display of a map and at the same time correspondingly change the level of detail. This is to enable a person to view a large region at small scale and relatively low level of detail and then "zoom" in on an area of interest and observe more detail as the scale became larger.

While interactive graphics has created a demand for new methods of representing map outlines it has at the same time provided a potentially powerful tool to aid in the conversion of outlines to a more compact form. This arises from the link that can be established between the computer's speed and accuracy at

---

[1] CRT = Cathode Ray Tube

arithmetic operations and the ability of people tc recognize easily shapes and patterns in two-dimensional information.

The work described in this thesis represents one attempt to take advantage of this link offered by interactive graphics to build a system for reducing the data content of map outlines. The way in which this has been done is to display a series of outlines on the screen of the CRT and allow the user to train the system by manually reducing their data content according to his own particular tastes and requirements. The system learns by recording and analysing the actions of the user until it can satisfactorily mimic the person's behaviour. Once this point has been reached the system is then given new lines to reduce on its own. The results can then be checked by the user and corrected. The user can also re-teach the system if necessary.

Chapter 1 describes in general terms the processes involved in the system and how they are related to work done by other people.

Chapter 2 goes into the detailed workings of the system.

Chapter 3 describes and analyses the results of experiments with the system.

Chapter 4 evaluates the performance of the system as well as the work as a whole.

CHAPTER I RELATED WORK

1.0 Introduction

The work described here draws on work done in many other fields. In a sense it properly belongs in cartography, but contributions come from fields as diverse as perceptual psychology, pattern recognition and learning, linguistics, computer graphics, interactive systems, and numerical analysis. Some of the influences from these areas are discussed in this chapter.

1.1 Cartographic Generalization

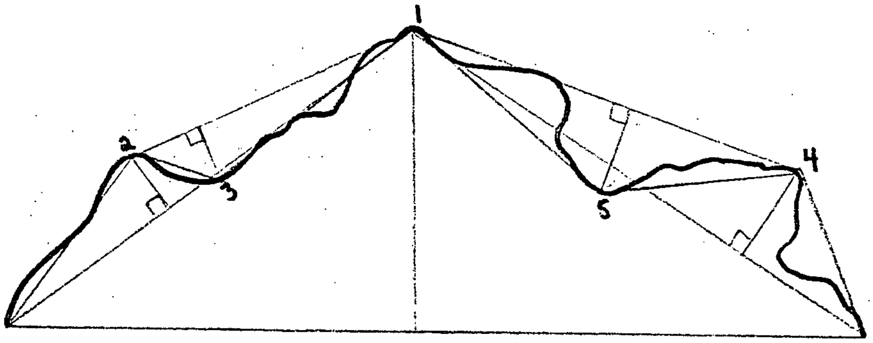
Reducing the information content of an outline is just one aspect of a process that cartographers refer to as "automatic generalization". Generalization is necessary whenever a map of reduced scale or special purpose is to be produced from other maps. The aim is "the expression of detailed by less detailed information by selection, and simplification" (Keates(1972)). Others think of it in terms of "simplification, selection, and emphasis." (Robinson and Sale(1969)) This is done so that the important spatial relationships are conveyed simply and clearly without interference from extraneous detail. For example, out of a potentially large number of possible choices certain towns, rivers, roads, islands, and so on must be selected for inclusion in a map while others are omitted or combined. Lines (e.g., boundaries, coastlines) must be simplified while maintaining their character (e.g., a rocky coastline should usually remain rough). This is in general a very complicated process demanding much knowledge and skill of the cartographer. The eventual purpose and scale of the map, special knowledge of the region, aesthetics, graphic limitations must all be considered by a cartographer in this work. It is thus highly subjective and therefore difficult to automate since any automatic scheme must include provision for these factors.

A major contribution to the automation of generalization has been the work of Topfer(1966). He has derived an expression that relates the density of map items to the scale of the map. This provides a quantitative criterion for judging the results of generalization. However this gives only an estimate of the number of items to be selected without any direct indications of the particular items to be selected. Work by Sukhov(1970), Srnka(1970), and others in Soviet circles have employed statistical and information theoretic principles to aid in this selection process. A combination of these approaches promises to be fruitful for automating generalization. However the factors of map purpose, aesthetics, and special regional knowledge will continue to demand the influence of experienced cartographers.

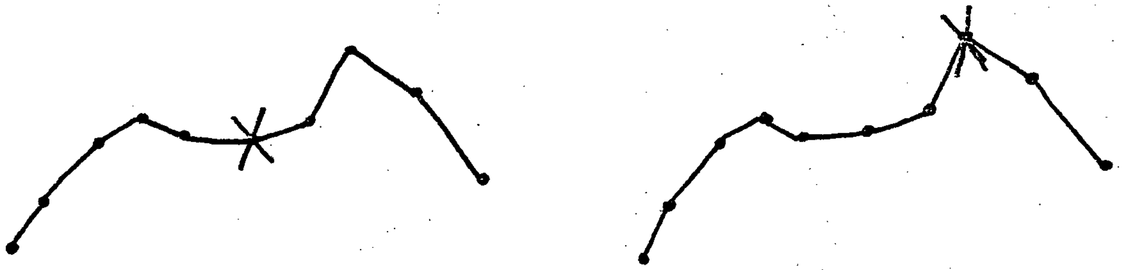
Much of the work in the automation of generalization has centred around the processing of line data. The work in this area can be roughly divided into two classes depending on how the points along a line are treated. The first class is characterized by "point filtering" schemes. What this means is that the points defining lines in the new map are simply a subset of the points from the original map. No points have been added or moved. Hershey (1963) removed points if they were closer than some amount depending on the display device's dot size. This technique ran into trouble when lines became very near each other and when there was a drastic scale change. Lang (1969) describes a scheme that removes points if they do not deviate too much from straight line approximations. In a similar vein is recent work by Douglas (1972). The process starts by considering a straight line joining the end points of the line. If the point on the line that is furthest from this straight line is farther than a specified tolerance then that point is selected. The process is repeated recursively on each of the two sections formed by the newly selected point until nowhere is the deviation greater than a specified amount. Figure 1.1 illustrates this and indicates the order in which points are selected. Thus, in a way, the selection of points depends on all of the rest of the line.

The work described in this thesis also belongs to this class. Points are selected based on the recognition of patterns in the lines that have been taught to the system by the user.

In the second group is work by Koeman and van der Weiden (1970). They use an averaging process over a sequence of points to alter the line, thus simplifying it. Recent work by Brophy (1972) combines approaches of both groups. Points are first selected based on the desired scale and line width and then moved according to the degree of generalization to either exaggerate or simplify the line. Certain features are eliminated



Douglas' Generalization Technique   Figure 1.1



Outlines with points to be removed   Figure 1.2

Outline with points removed   Figure 1.3

if adjacent lines begin to merge.

This work in cartography, although directed towards reducing the information content of a line, is only of limited applicability. The reason for this is that the maps that result are quite different from the sort of maps we are interested in. The cartographer's maps are made of paper, they are "hard" and static. They will be hung on walls, stared at, and measured carefully. Accuracy is important and the points are usually spaced according to the best resolution possible (i.e., spacing on the order of the line width). The degree of information reduction required is governed only by the eye's ability to perceive and distinguish images from coloured ink on paper. On the other hand, the maps that we are interested in are quite different. Our maps will be dynamic, "soft", and formed by the glow of phosphor dots of the screen of a CRT. Most of these maps will live only for a few seconds or minutes to convey some relationship before being replaced by another. Accuracy will not be so important since measurements will probably not be made from these maps. This means that points can be fairly far apart - the important criterion being that the objects are clearly identifiable. The information content of lines will be reduced to diminish the storage requirements, transmission time, and processing time. At CRT devices the amount of information that can be displayed is often restricted by the memory size of the device. Transmitting large amounts of information to a display terminal is often time-consuming and expensive. The processing times for many operations that can be performed on maps grow very quickly with the number of points involved (e.g., finding the intersection of two regions can be proportional to the square of the number of points).

These considerations encourage us to be "bold and intrepid" (Miller and Voskuil(1964)) in the elimination of points. Even though we are forced to reduce drastically the number of points things are not so bad since the constraints of accuracy have been loosened somewhat.

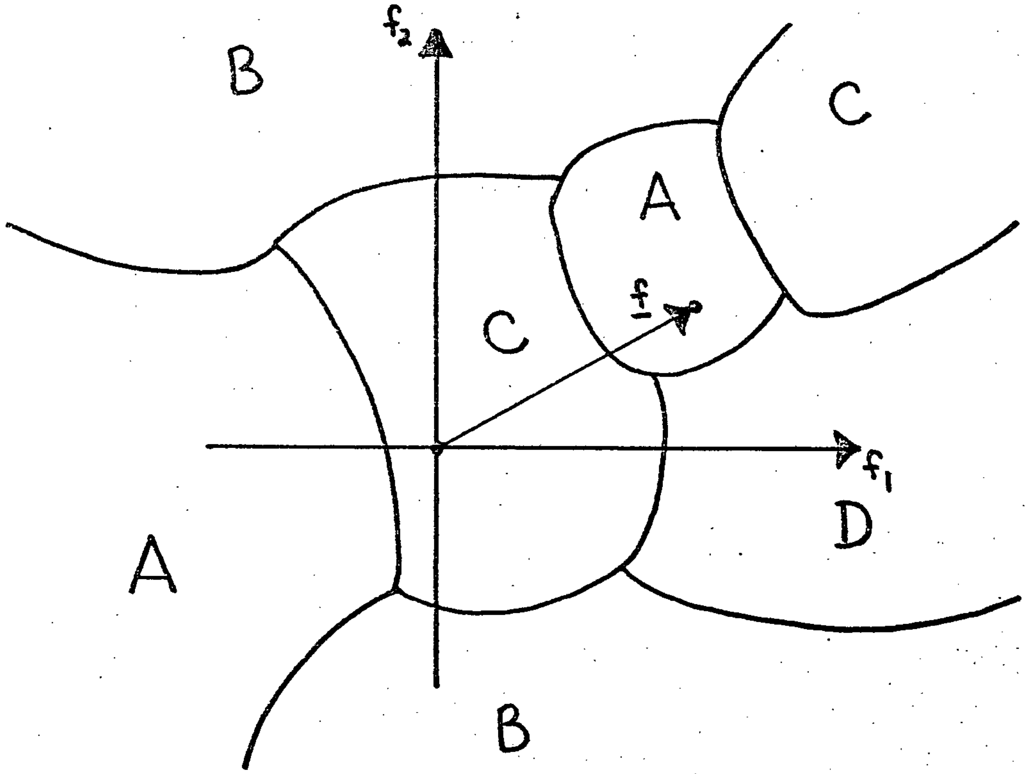
1.2 Psychology Of Perception

We receive some additional indication that we shall be successful from the field of perceptual psychology. Studies by Attneave (1954) show that most of the information for the recognition of a figure comes from the regions of maximum curvature. Related indications come from Ryan and Shwartz(1956) who report that caricatures, though distorted, are often more readily recognized than photographs of the corresponding subjects. This is because there is too much detail and redundant information in the original. Hopefully this will be true for the identification of map outlines as well. If it is true then it should only be necessary to recognize and select points where the curvature is quite large in order to produce outlines that are easily recognizable. Some contrary indications come from Gestalt Psychology since it is maintained that the recognition of a figure is dependent on the figure taken as a whole. In other words, figure recognition is a global rather than a local process. Which of these views is the more important in our situation will have to be decided by experiment.

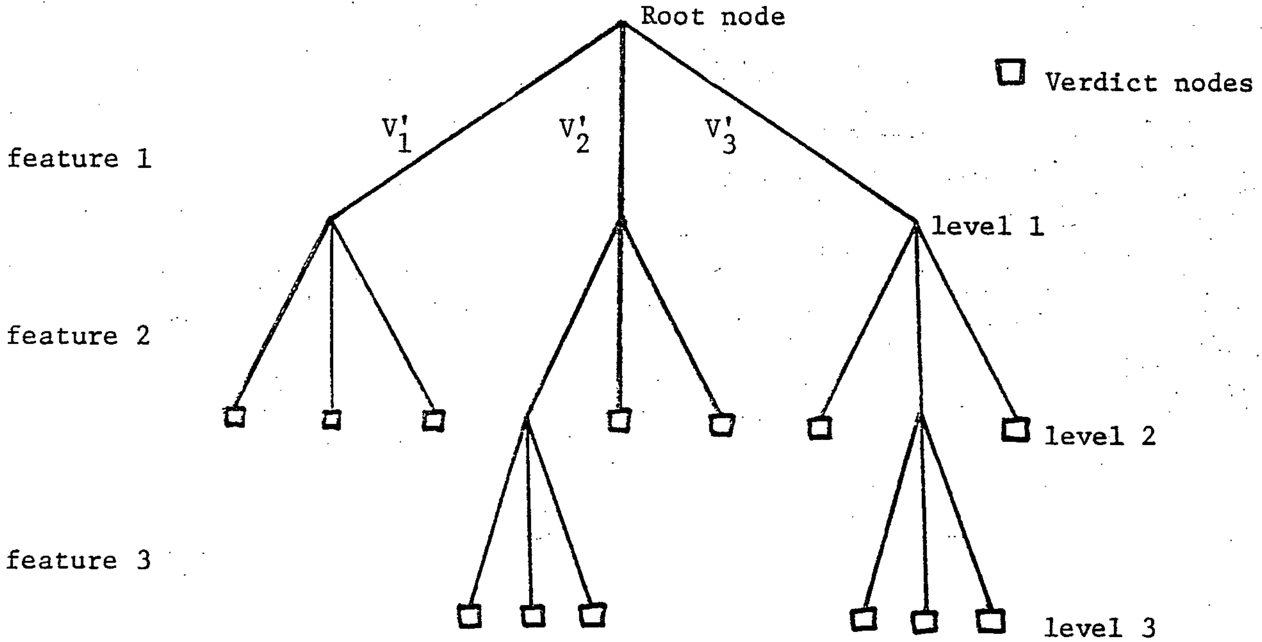
1.3 Pattern Recognition And Learning

In deciding which points are to be removed from an outline in order to reduce its data content the neighbouring portions of the line must be considered. For example, in the two cases shown in Figure 1.2 the point on the left can safely be removed while the one on the right cannot. The result of removing these points is shown in Figure 1.3. In order for the system to automatically decide on the fate of points in a host of less clear cut cases it must somehow "look" at sections of the line and classify them appropriately. This process is an example of what is commonly known as "Pattern Recognition". A pattern is described by an n-tuple of features $\underline{f}$. i.e., $\underline{f} = (f^1, f^2, \ldots, f^n)$. Pattern Recognition consists of assigning these patterns to classes, c , out of a set of m classes $\underline{c} = \{c^1, c^2, \ldots, c^m\}$. A standard way of looking at this problem is to consider the space of patterns to be divided into a number of disjoint regions each with a unique label chosen from $\underline{c}$. Classifying a pattern now means finding the label of the region in which the pattern vector lies. For example figure 1.4 illustrates a case where n=2 and m=4. The pattern vector $\underline{f}$ lies in a region labelled "A". The difficulty arises in defining the boundaries of these regions.

Since we can rarely know *a priori* what the region boundaries are a pattern recognizing machine must be trained to determine them. This is usually done by specifying initial approximate boundaries and then adjusting them based on externally classified patterns. This is done by giving the device a series of patterns to classify. These patterns should



Two Feature Pattern Classification Figure 1.4

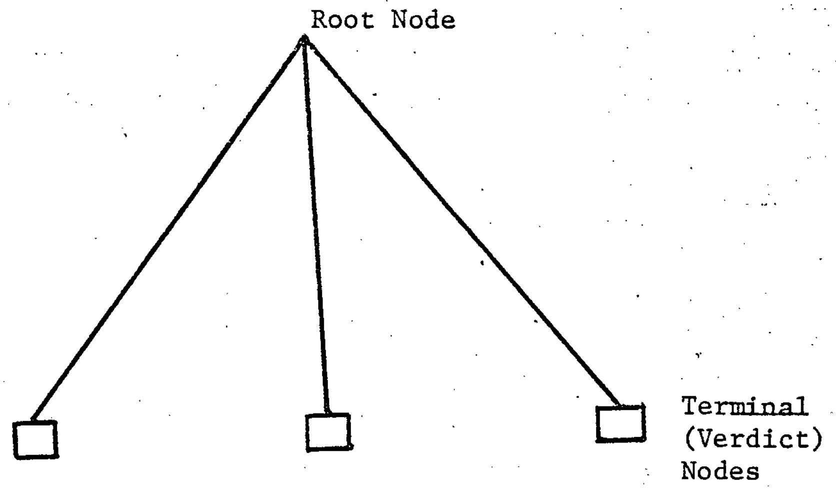


Decision Tree for Pattern Classification Figure 1.5

be fairly representative of all the pattern classes. If a pattern is classified correctly then boundaries are often left unchanged. However, if the classification is incorrect the boundary of the appropriate region is moved closer to the point so as to either give the correct classification or else come "nearer" to it. With successive patterns and with repetitions of the same patterns the classification should become more and more reliable and the boundaries move less and less. If the pattern classes are well separated by the measured features , the training patterns are reasonably representative, and the initial boundaries are not too far out then this process should converge and result in a satisfactory pattern classifier.

A standard way of specifying region boundaries is in a piece-wise fashion using hyperplanes. This means that only linear polynomials are needed to evaluate pattern class membership. The learning process using this approach consists of adjusting polynomial coefficients. This is a fairly well understood technique and would be appropriate for us except that it demands that all n features be measured in order to arrive at a verdict for a pattern. In our case, in order to consistently guarantee that enough of the line surrounding a point is considered, n would have to be quite large, say 10 or more (using angles and lengths of straight line segments). However, it is often possible to make the right decision by considering only a single feature. This occurs when the line is virtually straight at the point. Since there is a certain expense associated with measuring a feature and since learning is generally slower and less reliable with more parameters a method which considers only the minimum number of relevant features would seem desirable. This is the approach taken in Sequential Pattern Recognition (SPR) (Slagle and Lee(1971)). With SPR features are considered one at a time as needed. When a new feature is measured a test is made to determine whether the pattern classification can be made reliably. If it can then we can move on to the next pattern. If not then a decision is made as to whether further features would be likely to increase the reliability of the verdict and if so , which one. Only if this is affirmative is another feature selected. Although the work for each feature considered this way is more than in the "parallel" case, fewer features will be involved so there can be a considerable overall saving.

A simplified version of this SPR technique has been used in this project. With each new feature the only decision made is whether a verdict can be made. If it can, it is, otherwise the next feature is considered. The order in which the features are taken is fixed ahead of time. This decision process is conveniently represented as a tree (see Fig. 1.5). The root is the initial node and the features as they are measured in turn cause transitions down the appropriate arcs until a terminal node is reached. At the beginning of the learning process the tree consists only of the top level (root plus terminal nodes,see Figure 1.6). As patterns are presented, if the verdict agrees with the classification given then that verdict is reinforced. If there is disagreement then the tree either sprouts new terminal nodes from the previous terminal node or else the verdict is weakened or changed depending on the depth



Decision Tree Before Learning Figure 1.6

and past history of that verdict. This learning and classification scheme is very similar to that of EPAM (Feigenbaum(1963)) and the work of Sherman and Ernst (1969).

Which features to measure is often a serious problem in Pattern Recognition. They must contain the essential distinguishing characteristics of the original, otherwise all subsequent work will be futile. Since we are dealing with graphic or two-dimensional information there might be an initial temptation to measure two-dimensional features (as in the case of Uhr and Vossler's pattern recognition machines(1963)). However, since we are interested in the boundaries of regions and not the regions themselves, we can conveniently exploit the essentially one-dimensional nature of our line drawings by converting them into strings of characters. I have chosen to do this by quantizing the lengths and the angles between individual segments along the line since this appears to offer the best way of capturing the essence of a line. The classification recognition problem can now be looked at as recognizing whether strings belong to a particular language over a finite alphabet. Since the data is "naturally" occuring the language is very messy and ill-defined, but this crudely linguistic approach could be fruitful in the future. Some people who have tried this approach in other fields are Miller and Shaw(1968), Feder(1968), Pfaltz(1970), and Seeley(1970).

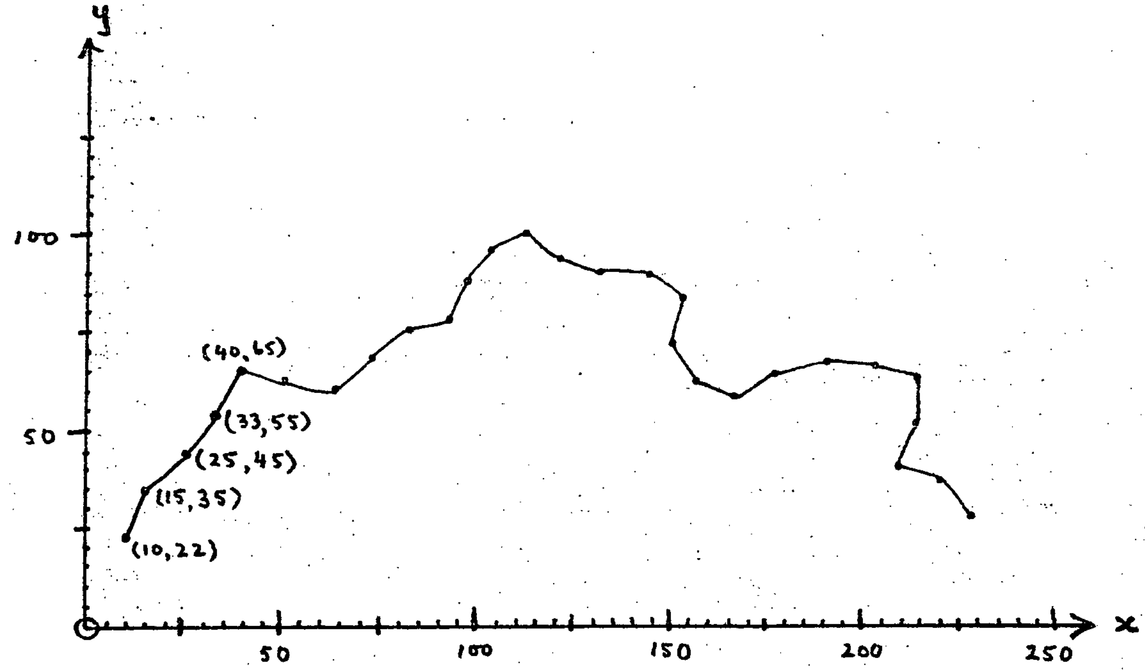
1.4 Line Encoding Schemes

Whenever one comes to representing line data in the computer one is faced with making a choice of which of three encoding schemes to use. The most traditional scheme is known as the vector approximation or X-Y coordinate method. A curve is approximated by a series of contiguous straight line segments and the absolute coordinates of the endpoints of these segments (or vectors) is recorded relative to a fixed coordinate system. For example, the curve shown in Figure 1.7 would be encoded numerically as:

X	Y
10	22
15	35
25	45
33	55
40	65
.	.
.	.
.	.

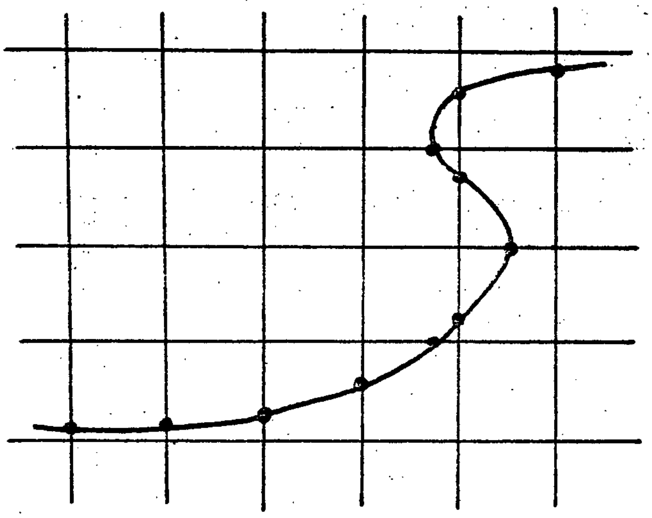
Another way to encode lines is with the use of Freeman chains (Freeman(1961)). This is done by overlaying a rectangular grid of fixed mesh size and finding the intersection points of the curve with the grid. (see Fig. 1.8) The grid points nearest these intersection points are determined and connected in order with line segments whose length is either 1 or sqrt(2) times the mesh size (see Fig. 1.9). These short segments are encoded with the digits 0 through 7 according to the diagram shown in Fig 1.10. Thus the final encoding of the original curve would be ...00101320... .

A third encoding scheme that is mainly suitable for closed



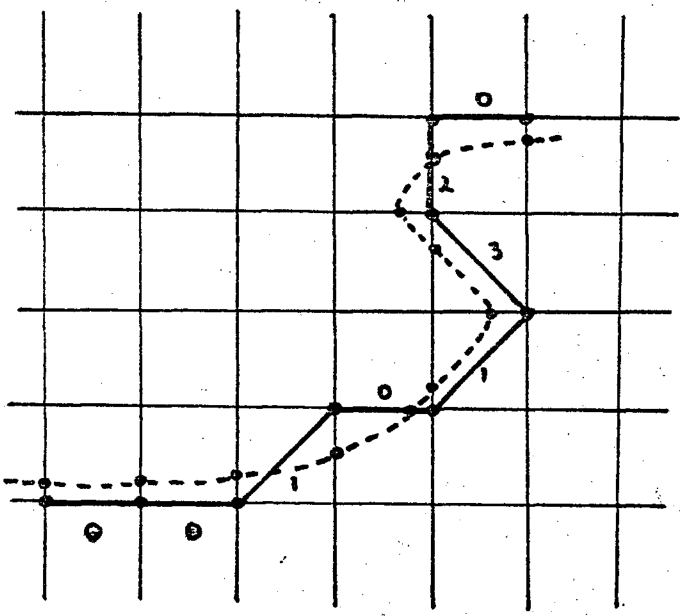
X-Y Coordinate Encoding

Figure 1.7

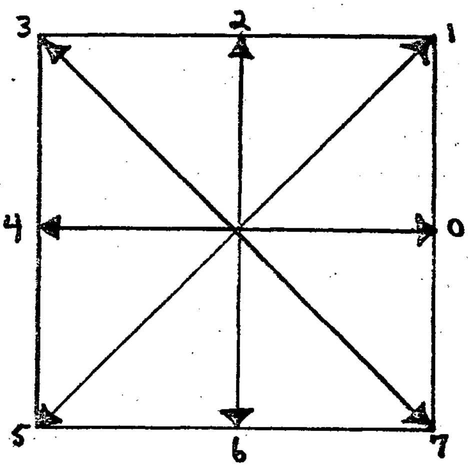


Chain-encoding (grid overlay)

Figure 1.8



Chain-encoding Figure 1.9



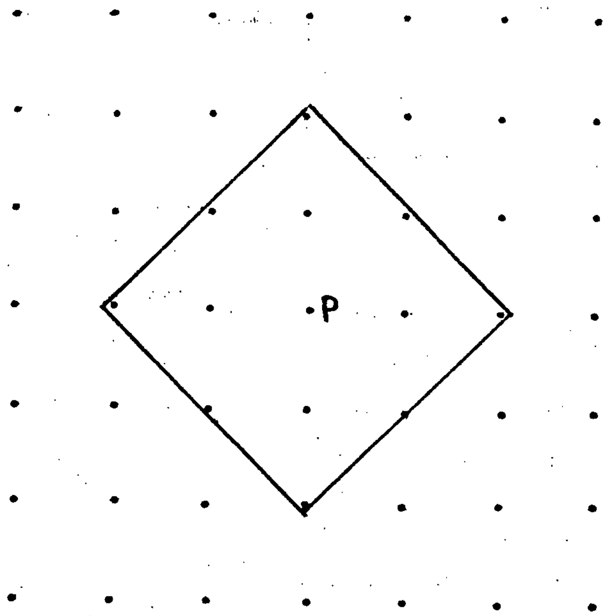
Chain-encoding (basic increments) Figure 1.10

curves is one first proposed by Blum (1964). This method, referred to as skeleton encoding, is based on finding the maximal neighbourhoods of planar regions. In a particular metric the neighbourhood of a point is the set of all points within some distance from that point. With the "city-block" metric in Fig. 1.11 (from Pfaltz and Rosenfeld (1967)) the 2-neighbourhood of the point P is indicated by the rhombus. For a closed region in the plane every point in the interior of the region has a neighbourhood completely within the region. The maximal neighbourhoods of the region are the set of such neighbourhoods that are not completely enclosed within some other neighbourhood. The centres of these maximal neighbourhoods form stick-like skeletons which together with the corresponding radii give an adequate description of the original outline (Figure 1.12). (see Pfaltz and Rosenfeld (1967), Montanari (1968)).

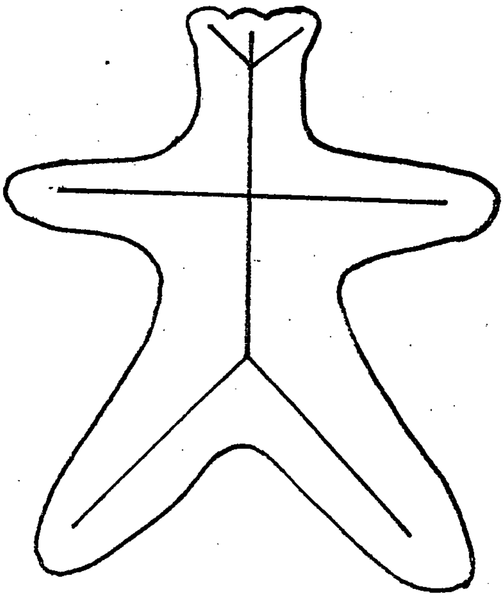
Each of these methods has its own advantages and disadvantages, depending on the situation in which it is used.[1] Generally chain-encoding gives the most compact representation for very detailed outlines and is well suited to finding lengths and areas. Also, chain encoding lends itself to considerable theoretical analysis (see Freeman (1961) and Feder (1968)). Skeleton-encoding, while poor for processes associated with the boundary, such as finding the perimeter of the region, is superior when it comes to "areal" operations such as shading and intersection of regions. Vector approximation is generally intermediate. It enjoys an advantage in economy of

---

[1] see Deeker(1970) for a much more complete comparison of these three encoding schemes.



The 2-neighbourhood of point P Figure 1.11



Skeleton of an Outline Figure 1.12

representation when the cutline is fairly simple and permits easy rotation. It is better suited to our particular purposes for reasons that will be discussed in the next chapter.

1.5 Line Reduction

The problem of reducing and simplifying line data is met in many areas other than cartography. Some of the major sources of this data are photomicrographs of chromosomes and cells in the biomedical field, bubble chamber photographs in high energy physics, and aerial photos in remote sensing. Such applications as these produce enormous quantities of line data that must be reduced in some way before they are manageable. Jarvis (1971) has done work on fitting low-order polygons to the chain-encoded boundaries of muscle cells. The approach he took was to pick a point in the interior of the cell and plot the distance to the boundary as a function of angle. The peaks and troughs of this function can be associated with the vertices of the original shape. These vertices are added one by one to the polygon description until a least squares deviation error falls below a specified threshold. Zahn (1969) has also attempted to reduce the data content of chain-encoded boundaries but by not nearly as drastic an amount. He essentially re-codes the description by recognizing regularities in the chain. No information is lost in the process and the original line can be reconstructed precisely. More recently (Zahn and Roskies(1972)) he has used Fourier Descriptors to encode lines. The coefficients in the Fourier expansion of a line are sufficient to specify it adequately. Since most of the information is usually stored in the lower terms the remainder can be dropped in order to reduce the data content of the line. There have been several attempts to obtain the minimum perimeter polygon (MPP) from the chain-encoding of shape. An MPP of a shape is the polygon of minimum perimeter that produces the same chain-encoding as the original shape. Montanari(1970) was one of the first and his approach was to start with a chain-encoding and then move the vertices around within small neighbourhoods of their original positions until the resulting boundary was of minimal length. In a more theoretical vein is the work of Sklansky et al(1972).

All these attempts have in common the fact that the resulting line must satisfy some strict mathematical relationship to the original line. This makes the lines more suitable for comparision and analysis but does not necessarily give the best reductions from the point of view of visual recognition. It may well turn out that there is some well-defined mathematical criterion that produces the most appropriate reduction visually, but one is not known yet. It therefore seems reasonable to allow the prospective user to define empirically what he thinks is most appropriate by reducing lines manually. The system could learn to mimic the user and thus the criteria are established internally according to the individual's needs and preferences.

1.6 Interactive Graphics

This tailoring of the system to suit a particular user's tastes is one of the main attractions offered by interactive computing. An interactive system is one in which a person has immediate access to the computation process and plays a direct role in guiding its course. This is done by having terminal devices attached to a computer that allow output from a program to be displayed to the user and permits the user to enter information for the program. The earliest computing systems were of this nature. A person had the machine to himself and was able to directly monitor its performance and make changes to it and the program as desired. As computers became faster and more powerful it became necessary to submit programs in batches with consequent separation of user and process. More recently, with the advent of time-sharing systems, it again became possible to put people and process back together again. This development promised many great things. It was thought that by coupling man and machine it would be possible to exploit the diverse skill of the two parties simultaneously. Computers are well-suited to performing simple repetitive operations quickly and accurately. People, while poor at this are good at recognizing patterns in information from diverse sources and identifying goals: things that computers are not good at.

While many see this linking of people and computer with their vastly different skills and speeds of operation as offering a much improved method of tackling many problems, Norbert Wiener did not share this optimism. He wrote in 1963:

> Disastrous results are to be expected not merely in the world of fairy tales but in the real world wherever two agencies essentially foreign to each other are coupled in the attempt to achieve a common purpose. If the communication between these two agencies as to the nature of this purpose is incomplete, it must only be expected that the results of this cooperation will be unsatisfactory. ... One of the chief causes of the danger of disastrous consequences in the use of the learning machine is that man and machine operate on two distinct time scales, so that the machine is much faster than man and the two do not gear together without serious difficulties.

Although this was written before interactive systems were widely available his comments are still relevant and should be considered seriously. An attitude of caution and scepticism is especially important when confronting proponents of Man-Machine Symbiosis such as Licklider (1960) lest it be forgotten that the machine is to be our tool.

On a more concrete level interactive systems promise to make computing more efficient from the user's point of view. Not only would results generally be available faster than with batch systems, but less work would be required on his part to obtain them. This is partly because the context of any operation could be narrowed considerably. Since a program can prompt for input and show the results immediately, the user does not have to anticipate ahead of time precisely what will be needed. This is especially applicable to debugging programs, where the behaviour is often unpredictable, and in editting programs and text, where the context is limited to the recent output and changes can be verified immediately.

Generally, applications most suitable for interactive systems can be characterized according to the following modes of operation:

- Relatively few, straight-forward operations with immediate response, e.g., inquiry systems (airline reservation systems, customer record systems)
- Many sequential decisions with fast response, each action often depending on the result of earlier actions; e.g., editting, program debugging, computer-aided learning (CAL), computer-aided design (CAD)
- Complex sequential operations, each action heavily dependent on results of previous ones; e.g., on-line problem-solving, interactive simulations.

While these different categories overlap and in fact really belong on a continuum they are intended to highlight some of the basic distinctions. It is in the second class that the work described here belongs. The sequential manual selection and rejection of points along an outline affects the shape of the resulting outline and hence influences future actions. Also, during learning it is important to know how the system is behaving since this has a bearing on the order in which sample patterns are presented to it.

Computer Graphics, which is the computer manipulation and display of 2-(or more) dimensional information (as opposed to the processing of one-dimensional string and numerical information) goes naturally with interactive computing. This is true for two distinct reasons. In the first case, with many interactive situations a considerable amount of data is produced upon which the next action by the user is to be based. This information is often best represented graphically because this is a form in which people are good at seeing patterns and relationships. This is true even when the information itself is not inherently graphic such as in numerical or statistical applications. Secondly, when dealing with information that is basically graphic , interactive methods are often highly preferred to non-interactive methods. This is because it obviously makes sense to specify in graphic terms the operations that are to be performed. The alternative is through numerical or linguistic descriptions. These are in general necessarily long and messy since so much context must be supplied that was inherent in the original graphic form. This is a consequence of the fact that pictures, diagrams, maps and so on are informationally very rich. This fact, however, makes it difficult to get raw graphic data into the machine. Because present computers are not "graphic machines" such information must be digitized somehow - usually a lengthy and expensive affair. The way to get around the problem of specifying operations graphically, once the basic information has been entered the hard way, is to display this information on a graphic terminal and use the available interactive devices to indicate the desired manipulations. These devices (such as light-pens, joy-sticks, function keys, "mice", tablets etc.) allow one to enter graphic information directly. They can be used to draw figures, select picture components to be operated on, control the form of the display and so on.

The information that we are dealing with, since it comes from maps, is basically graphic. In addition the selection/rejection of points demands the visual inspection of outlines and specification of operations on a point by point basis. For these reasons the facilities of interactive graphics were used in the approach described here to the reduction of map outlines.

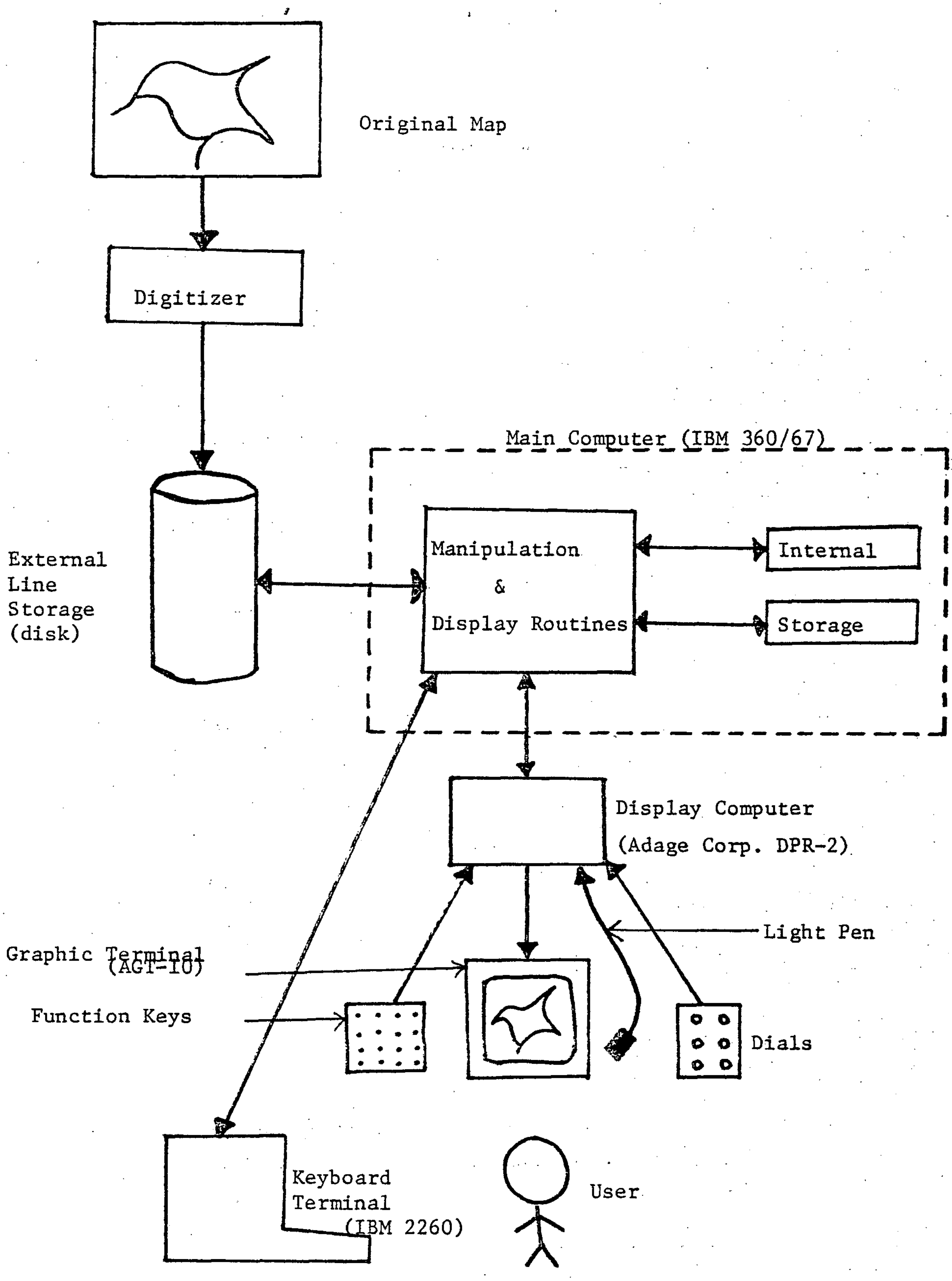
CHAPTER II METHODS AND TECHNIQUES

2.0 Introduction

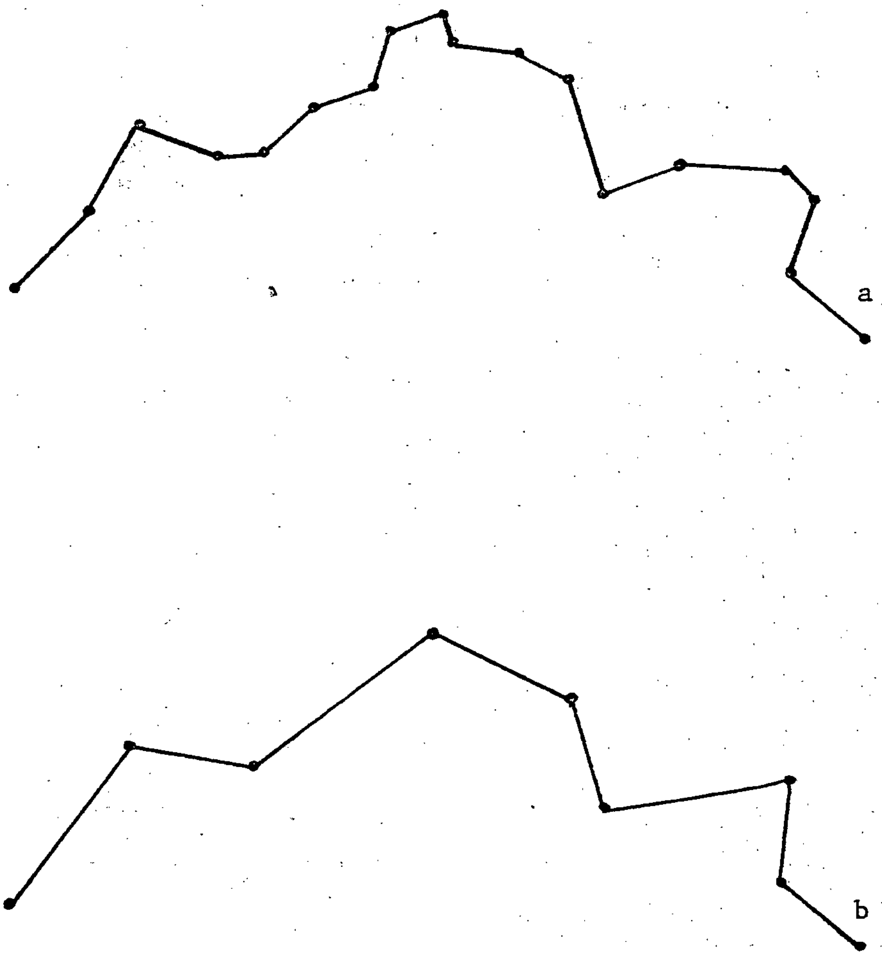
The learning and subsequent automatic generalization of lines is embedded in a considerably larger system for manipulating lines and the levels that are attached to the individual points on these lines. The basic components of this system and their interrelations are shown in Figure 2.1. This diagram shows the flow of information as lines on maps are digitized and the data stored on a disk. A person using the various devices around him can specify that these lines are to be brought into the computer's internal memory, manipulated in various ways, and displayed on the screen of the graphics terminal.

2.1 Line Representation (external)

I chose to represent lines externally by means of the standard X-Y coordinate method as described in the previous chapter. The line used in the example then (see Figure 1.7) might be generalized several times to give a number of levels of detail as shown in Figure 2.2 (a and b). The additional information to be stored in the line is easily handled by attaching to each point of the vector approximation a number that indicates the relative importance of that point in conveying the shape of the line as a whole. In the case of the previous example these values (or levels) of the points would be as depicted in Figure 2.3. When it comes to displaying this line at a particular level of detail, these levels tell us which

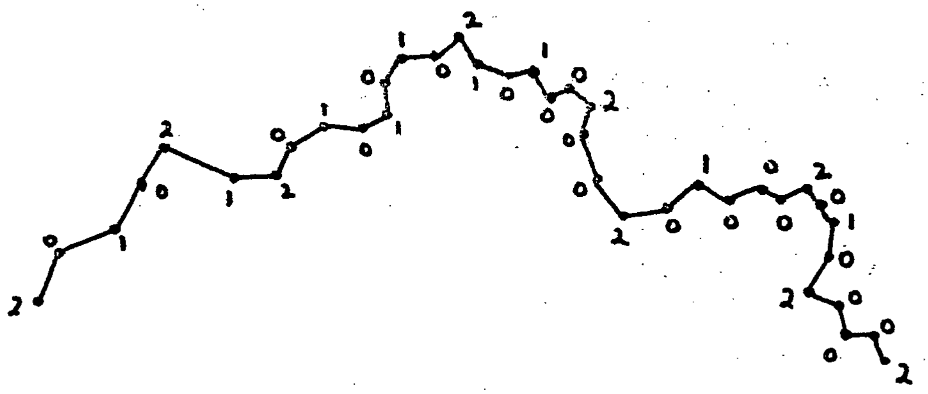


The Interactive Line Processing System Figure 2.1



Levels of Detail

Figure 2.2



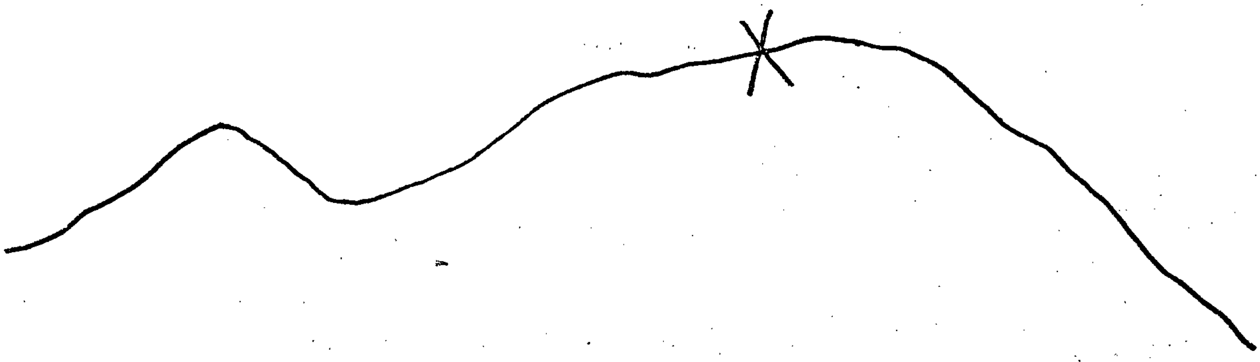
Level Values Attached to Points

Figure 2.3

points to include and which to leave out. A line displayed at a specified level will contain only those points whose levels are greater than or equal to the specified level. Thus at level 0 we get Figure 2.3 (without the numbers), at level 1 we get Figure 2.2a, and at level 2, Figure 2.2b. The numerical representation of the line with all these levels would be:

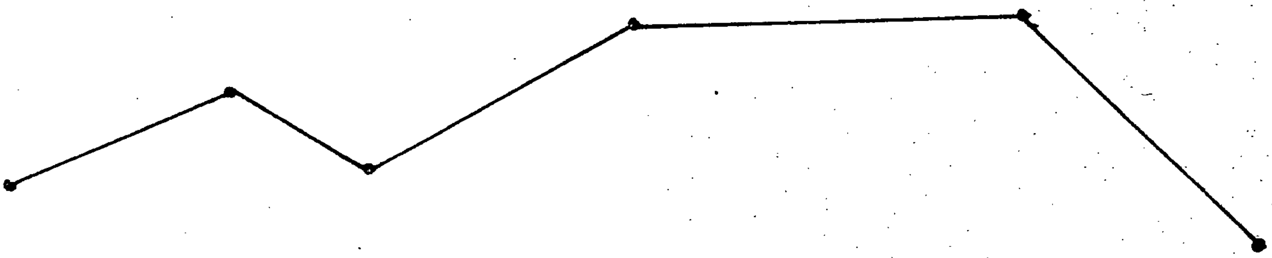
X	Y	V
10	22	2
15	35	0
25	45	1
33	55	0
.	.	.
.	.	.
.	.	.

The method described above for encoding lines at different levels by attaching a unique level to each point is quite adequate as long as there is a strict hierarchical ordering of the 'importance' of each point. For example the line shown in Figure 2.4 might appear at one level as in Figure 2.5 while at a lower level of detail (i.e., higher level of display) appear as in Figure 2.6. The point indicated by the 'X' cannot have a unique level but needs rather ranges of levels associated with it. A similar problem arises when handling the representations of objects such as rivers. If the original map contained a river as in Figure 2.7 and at a particular level of generalization it was to appear as shown in Figure 2.8 then single levels are again not adequate since the two edges come together. This last case can be taken care of by associating with each point two values that specify the upper and lower limits of the levels at which the point can be displayed. In the case of the river, several different lines would have to be associated with the



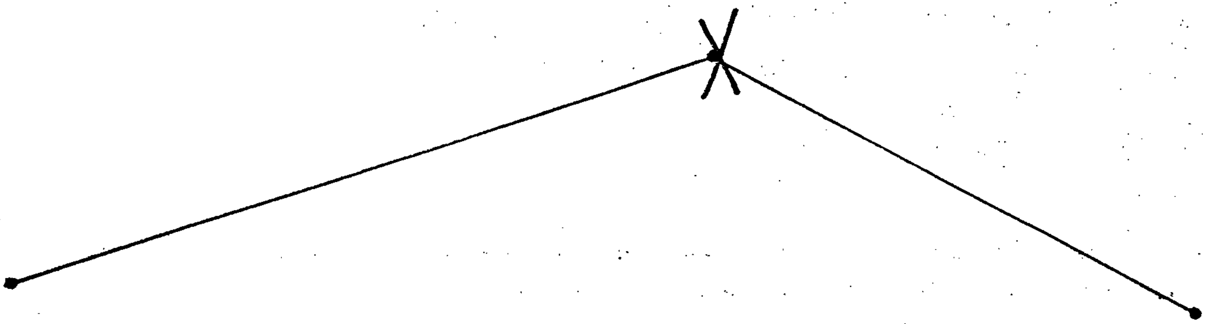
A Line to be Generalized

Figure 2.4



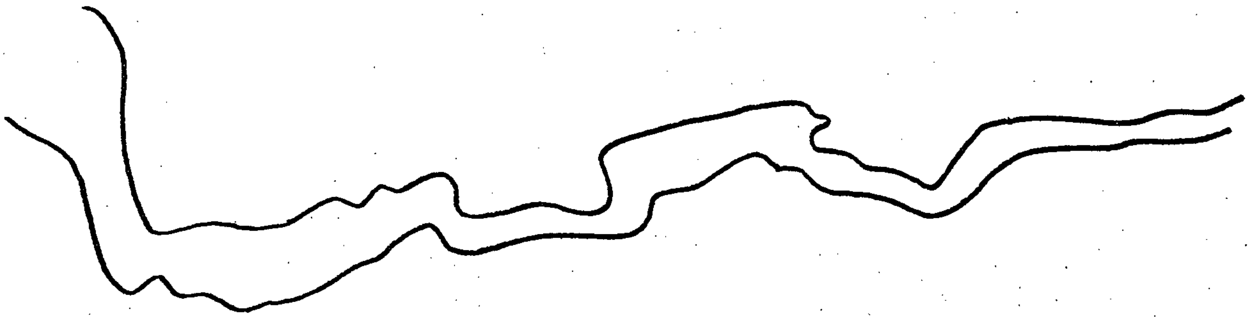
Generalized Line

Figure 2.5



Generalized to a Greater Degree

Figure 2.6



A River to be Generalized

Figure 2.7

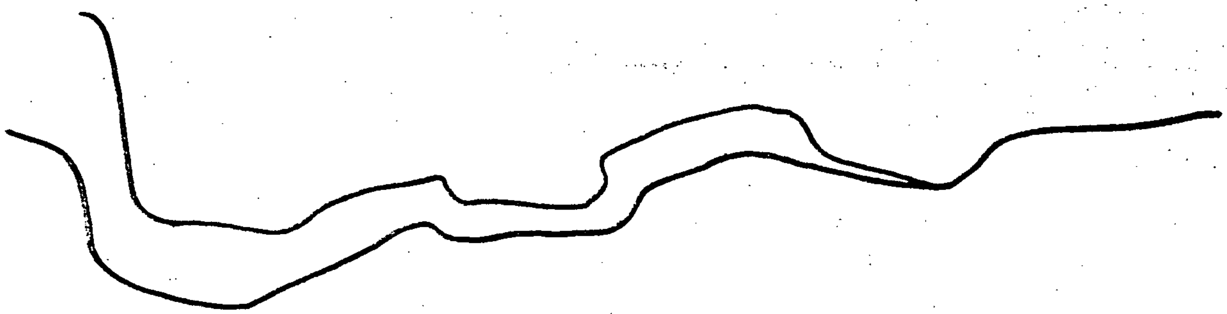


Figure 2.8

Generalized River

river since the two river banks merge. By appropriate adjustment of the values only the proper lines would appear when they were all 'displayed' at some particular level of generalization.

The implemented system makes provision for the inclusion of two values associated with each point, although the system is not really suitable for handling these more complicated cases in a convenient way. For the remainder of this discussion only the first of the two values will be referred to. No confusion should arise from this exclusion.

The vector approximation method for encoding curves was chosen because it enjoys several considerable advantages over the alternative methods. Chain encoding using Freeman chains, while an excellent way to represent static lines, gives rise to serious problems when drastic generalization is attempted. The reason for this is that data reduction can only really be accomplished by making the base grid coarser. Chain encoding's reliance on a regular grid means that it cannot take advantage of long, relatively featureless sections of outlines to reduce storage requirements. These relatively straight sections of outline occur quite frequently in the sort of data that we are interested in (urban, political boundaries) and especially as the lines become less detailed through generalization. The vector approximation method can take full advantage of these sections, concentrating vectors only in the areas of important features. Another important objection to chain encoding is that one must maintain as many copies of a line as there are levels of generalization. Although the space per copy decreases as the level of generalization increases, because of the larger step sizes, it still results in being very clumsy to handle . If one is not happy with sections of the encoding at a particular level then that whole line must be regenerated from the next lowest level. The changes could also affect successively higher levels of lines. The vector approximation with levels attached to the points allows the changing of levels to be done much more easily. A further point in favour of vector approximation is that most of the data that was available was digitized by this method. Also, the graphic display device (an AGT-10) is a vector driven machine.

Skeleton-encoding is better suited to representing different levels of generalization than chain-encoding because the radii of the maximal neighbourhoods already indicate to a considerable extent the relative importance of that particular neighbourhood. In this way generalization could be seen naturally in terms of simply choosing those neighbourhoods whose radii were greater than some particular value. This would probably lead to fairly decent representations. However the expense of converting from the boundary encoding of the initial digitization into skeleton form and then back again for display would be considerable. Also there does not seem to be an obvious and convenient way for simplifications in the outline to be reflected in changes in the skeleton short of re-encoding the whole region.

2.2 Manual Interactive Alteration Of Levels

In the previous section we have seen that an outline can be conveniently represented on several levels of generalization by attaching an extra value to each point along the outline. In this section I will describe the way in which these values can be altered either en masse or singly, under the direct control of the user.

In order to alter the levels of one or more points several stages of selection must be made. The first stage is to select the subset of lines containing the desired points. This is done by specifying the identifiers of these particular lines. In addition the levels of the lines in this subset are also specified. If the set of line identifiers is I={i1,i2,...in} and the level of selection is VAL then the set of points selected so far is:

$$P = \bigcup_{\substack{i \in I \\ 1 \le k \le |Li|}} \{ (Xik,\ Yik,\ Vik) \in Li \mid Vik \geq VAL \}$$

(where |Li| is the number of points in line Li)

If a number of values are to be altered simultaneously then the set P can be further refined by specifying an upper bound (UBV) and lower bound (LBV) for the values of the points to be altered. The resulting set of points is:

$$P' = \{ (Xj,\ Yj,\ Vj) \in P \mid LBV \leq Vj \leq UBV \}$$

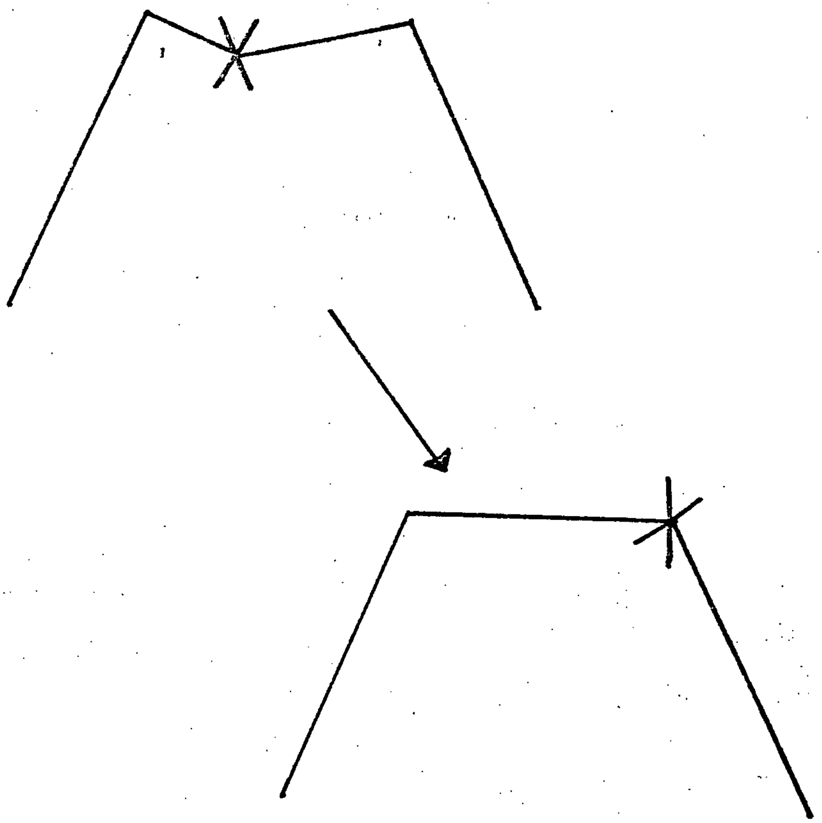
If, instead, the values are to be altered one at a time then a specified fragment of P or one of the Li can be displayed on the screen of the display device at another specified level (VAL). The fragment to be displayed is described by giving a starting position (START) and a length (LEN) so that what appears on the screen is the set of points:

$$PS = \{ (Xj, Yj, Vj) \in P \mid Vj \geq \underline{VAL}, \ START \leq j < START + LEN \}$$

If one of the lines (Li) was given then the expression for PS is the same but with $P \cap Li$ replacing P. i.e.,

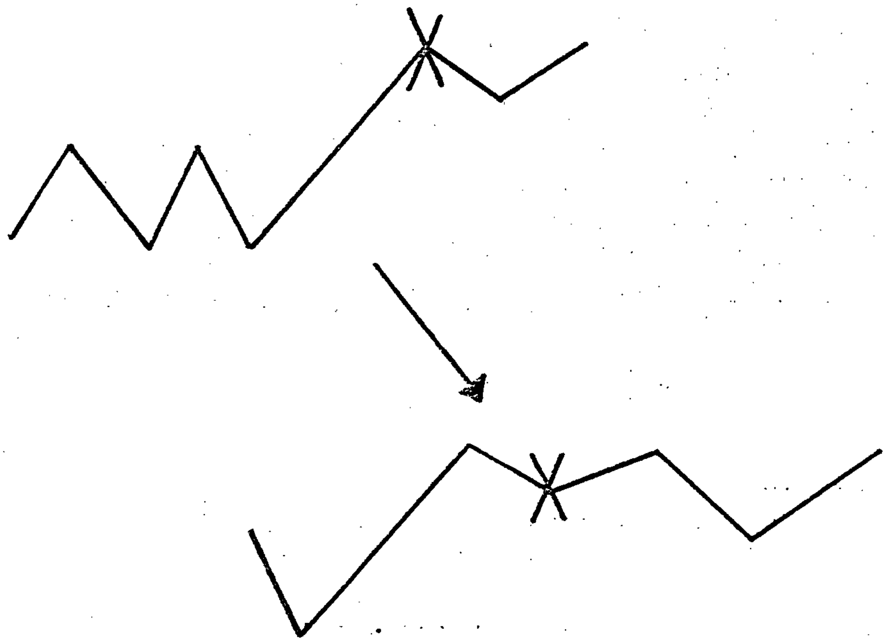
$$PS = \{ (Xj, Yj, Vj) \in P \cap Li \mid Vj \geq \underline{VAL}, \ START \leq j < START + LEN \}$$

Consecutive points are joined by straight lines if the points belong to the same line and they are all scaled to fill the screen as much as possible. Superimposed on the first point is a small 'X' and at the bottom of the screen is some text indicating the level of display and the name of the line if one was specified. If the value of the point indicated by the 'X' is to be altered then a particular function key attached to the graphics computer is pressed, otherwise a different function key is pressed. In either case the 'X' moves on to the next point. If the new value of a point falls below the display level then that point and its adjoining segments disappear, and if it was not an end point the adjacent points are reconnected directly (see Fig 2.9). Before the 'X' gets to the end of the section of the line displayed on the screen, and if there are points remaining to be displayed then the next portion of this set of points is displayed together with the last few of the former portion (see Fig. 2.10). The purpose of this is so that there will always be a reasonable amount of context for making the decision to alter the value of a point. In this way the 'X' steps along the line and the value of each point is either



Point Alteration

Figure 2.9



The Changing View of the Line

Figure 2.10

transformed or left unchanged. Regardless of which method has been used to select the points to be altered, the method of changing the values is the same. The values of the points that have been selected undergo a linear transformation. i.e.:

$$Vnew = C1 + C2 * Vold$$

The parameters of this transformation (C1 and C2) are specified by the user at the time he initiates the second step of the selection process.

2.3 Learning

In the last section we saw how lines could be generalized by hand, so to speak. Although doing it this way is fairly fast, it is still fairly expensive and very time consuming if many lines are to be processed. For this reason a component was added to the system that allows it to learn to mimic the selection behaviour of the user. Once the program's performance reasonably approximates that of the user then the job of selecting points for alteration can be left up to the program.

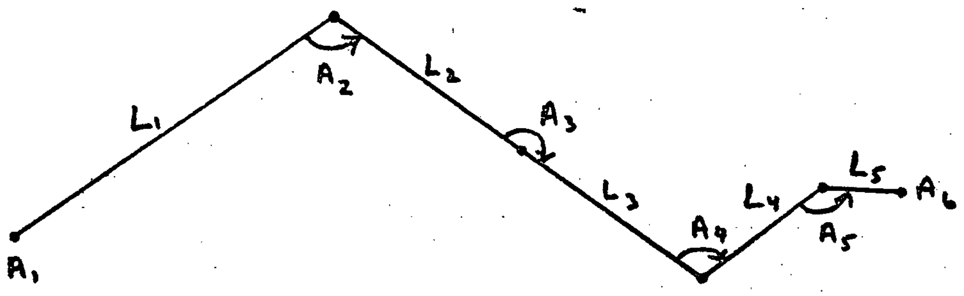
It is my hypothesis that if a person could do a satisfactory job of selecting points, when all he could see at one time was a small section of the line (10 points, say), then so could a program. (The correctness of the basic assumption in this hypothesis will be examined in the next chapter.) The first step in getting the machine to recognize points for alteration is to represent the lines in a more convenient form. The main criterion that a new encoding scheme must satisfy is that it must represent lines in a much more general way without sacrificing the essential features. It should also consist of the elements that are most important for visual discrimination. I suspect these are the lengths of the vectors and the angles between them. For these reasons I chose to adopt a variant of chain-encoding to represent lines to be processed by the learning and automatic alteration components of the system. Using this scheme a line is represented by an alternating sequence of lengths (Li) and angles (Ai). For example, the line in Figure 2.11 would be stored as:

$$A_1 \; L_1 \; A_2 \; L_2 \; A_3 \; L_3 \ldots$$

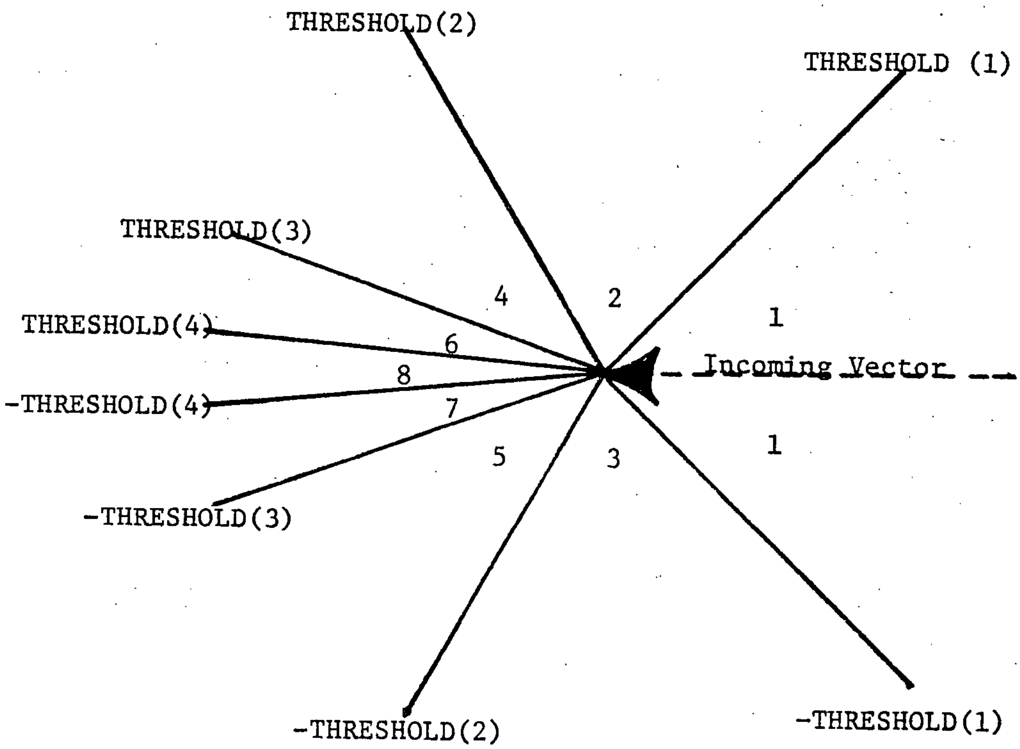
So far there has been no information lost in the sense that the original digitized line could be reconstructed exactly. However we have to go further than this since there is still too much distinction between essentially similar lines. The next step is to quantize these angles and lengths. I.e., they are transformed to take on only discrete values. (In the current implementation the number of these discrete values for both angles and lengths is 8.) The quantized length depends mainly on the logarithm of the original length since it provides a good way to compress the great range over which the lengths can vary and also because it seems to correspond to visual importance. (Another good potential candidate would be the Arctangent function since it too compresses a great range of lengths.) The exact relationship between the quantized and original length that the system currently uses is given by the following expression:

$$QLEN = \left\lfloor \log\left(\frac{LEN}{MINLEN}\right) * \frac{8}{\log\left(\frac{MAXLEN}{MINLEN}\right)} + 1 \right\rfloor$$

The parameters MINLEN and MAXLEN can be specified by the user



Angles and Lengths for Line Quantization Figure 2.11



Thresholds for Angle Quantiztion Figure 2.12

based on prior knowledge of the lines being processed and should approximate the actual bounds on the lengths to be encountered. The "8" corresponds to the number of levels of quantization. The net effect is to transform the lengths onto the integers 1 through 8 such that the quantization levels are closer together with shorter lengths.

The angle between the two vectors at a point is calculated by rigidly rotating the two vectors in order that the incoming vector lies along the positive x-axis. The angle made by the outgoing vector is then just the usual one with the restriction that the angle must be between $+\pi$ and $-\pi$. The correspondence between this angle and the quantized angle is given by Fig. 2.12. Thus if the original angle was between THRESHOLD(2) and THRESHOLD(3) the corresponding quantized angle would be 4, and so on. The standard values for these thresholds are approximately those depicted in Fig. 2.12, but can be changed at will by the user. The thresholds are bunched around 180° because most angles will be in this range and this is the critical region for point elimination.

This quantization process is applied alternately to each angle and length along the entire line so that a line that originally appeared as in Figure 2.11 might result in:

0 4 2 3 8 3 3 2 4 1 0

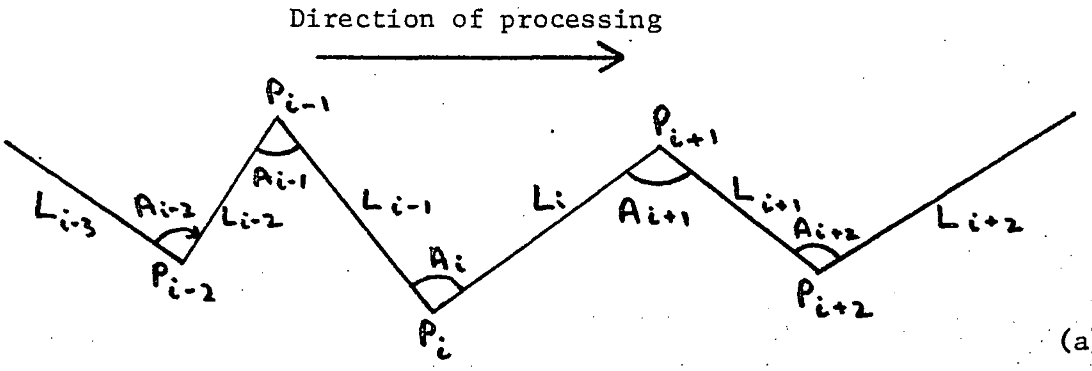
where the 0's indicate undefined angles at the ends. When a particular point is under consideration during the learning process this quantized version of the line is reordered to correspond to the 'view' of the line as seen from that point.

There are a number of ways that this could be done and the way I have chosen is as follows: If the line in the vicinity of a point Pi is as depicted in Figure 2.13a and the direction of processing is to the right then the line is transformed into the chain shown in Figure 2.13b. This gives a bias to looking ahead along the line.

Effectively we progressively look farther in alternating directions along the line. If one end of the line is encountered then that direction of view stops and is continued in the other direction until that end is reached too. A bias is given to angles in this scheme because it seems that they are more important in visual discrimination than are lengths.

The learning process at a point begins by feeding the line in this converted form into a decision tree (see Fig 2.14a) in order to come up with a verdict on whether or not the point should be altered. This string of symbols determines a path through the decision tree until a terminal node containing the verdict is reached. For example if the line was represented by the sequence "3545..." then it would reach the indicated node where the verdict is that the point should be altered. In addition to the verdict being located at the terminal nodes there is also stored a measure of how 'strong' or 'reliable' the verdict is and also how 'old' it is (i.e., the number of times it has been referenced).

Once the expected verdict has been determined in this way it is compared with the verdict of the user. If they agree then the verdict at that terminal node can simply be made 'stronger'



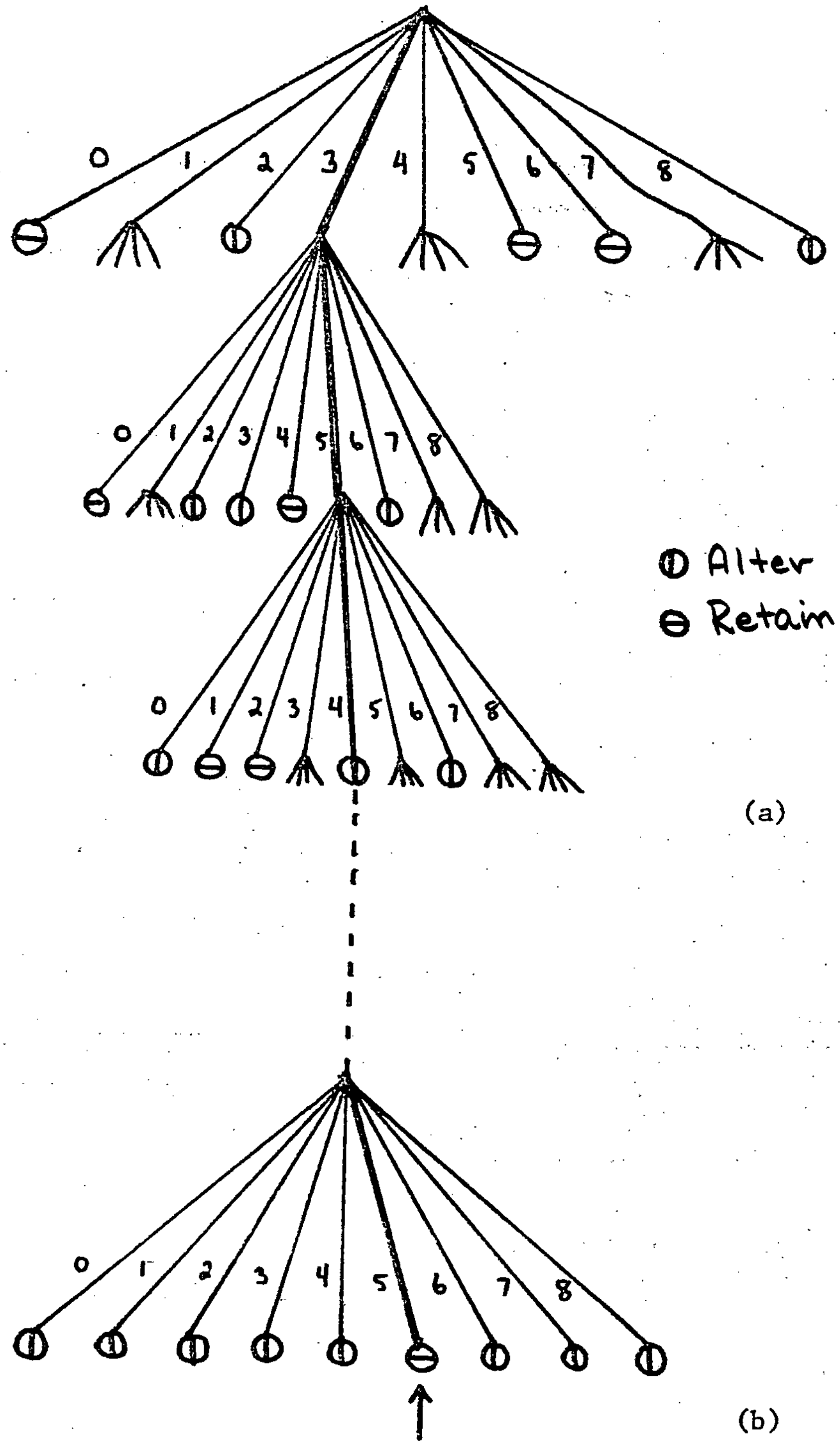
(a)

$$A_i\, L_i\, A_{i+1}\, L_{i+1}\, A_{i-1}\, L_{i-1}\, A_{i+2}\, L_{i+2}\, A_{i-2}\, L_{i-2} \ldots$$

(b)

Transformation of a Line
with respect to the point P_i

Figure 2.13



The Decision Tree and an Addition to it Figure 2.14

and 'older' by a unit value. However if they disagree then there are several things that can be done: the verdict can be left the same but made weaker, or the verdict can be changed, or the tree can sprout new terminal nodes from that node, each with its own verdict. Which of these three alternatives is chosen depends on the depth of the node, the 'age' and 'strength' of the verdict and whether there are any more symbols left in the line. These factors are contained in the following ALGOL-like expression:

$$\text{if } \left(4 * \left(\frac{1}{\text{DEPTH}} - \frac{1}{\text{MAXDEPTH}}\right) - \left(\frac{\text{STRENGTH}+1}{\text{AGE}+1}\right)\right) > 0$$

```
      then  "SPROUT"
      else  if  STRENGTH > 0
                  then  "WEAKEN"
                  else  "CHANGE";
```

The effect of this expression is such that the deeper the node is, the less likely it is that the tree will be expanded further. This is done to avoid growing excessively large trees (e.g., the number of nodes in a complete tree of depth d is about 9**d which grows very quickly with d). Also if the verdict is nearly as 'strong' as it is 'old' and it is reasonably old then it has given good service and so should only be punished slightly. This is done by making it weaker by one unit. If, however, it is young or old and weak then it is much more likely to have the verdict changed. In this case the age is incremented as usual but the strength is reduced by the unit amount. If, in fact, it has been decided to sprout more terminal nodes then the verdict of all these new nodes, except one, is set to agree with the verdict of the former terminal node. The opposite verdict is given to the node reached by considering the next symbol of the line. For example if the input line is the same as in the example above (i.e., "3545...") and it has been decided to expand the tree because the user did not concur with the expected verdict, then the additions to the tree will appear as in Figure 2.14b. In this manner the tree grows from being only of depth 1 at the start of learning.

There are two ways for the user to specify the true verdict. One way is to manually alter the levels at the same time. In other words the verdict comes directly from the person pushing one of the two function keys at the graphics terminal. An advantage of this method is that a preview of the decision of the program is available on the screen. This can help the user guide the training of the program. If the level to which a point is altered is lower than the level of display then the internal quantized version of the line is changed to reflect the deletion of this point. This allows the 'view' of the line to be effectively expanded at no extra cost.

The other way to indicate the true verdict is to alter the levels beforehand and then the verdict is determined by whether the level of a point is below a specified level. This has the advantage that the same line can be used many times to reinforce the message. To help evaluate the behaviour of the program learning statistics are available with both methods. These statistics give a breakdown for each line processed in terms of the number of verdicts made stronger and weaker and "changed" and the number of times the tree was enlarged. In general, the program will have learned to capacity when continued learning results mainly in verdicts getting stronger with relatively few verdicts being made weaker or changed and only slcw enlargement of the tree.

2.4 Automatic Generalization

Once the program appears to have learned satisfactorily it can be turned loose on new lines that are similar to the lines it learned on. This is done by specifying the particular lines, a level, and the transformation to be performed on the levels of the points to be selected. In this case, once a line has been quantized, converted, and fed into the decision tree, then the verdict returned determines whether the point is to be altered. If the new value for the point's level falls below the given level then the quantized representation of the line is updated.

This automatic generalization can be helped out by raising the levels of some potentially "borderline" points. This is done so that even if these points are later altered the resulting levels will still be sufficiently high to ensure that they will always be part of the context of nearby points. After the lines have been altered the results can be displayed for inspection and correction. Those sections that have been generalized poorly can be redone manually. If there is some similarity between these sections then the program can be taught some more as these corrections are being made. This will hopefully diminish the chances of the same mistakes being made in the future.

Chapter III EXPERIMENTAL RESULTS

3.0 Introduction

There are a great many questions that can be asked regarding the approach to the generalization that has been described in the previous chapter. Three of the most important and wide ranging of these questions are:

- What is the nature of people's perceptual responses to generalized outlines?
- Is it possible to achieve satisfactory generalization of outlines with purely local considerations?
- How well does the system perform with regard to the generalization of map outlines?

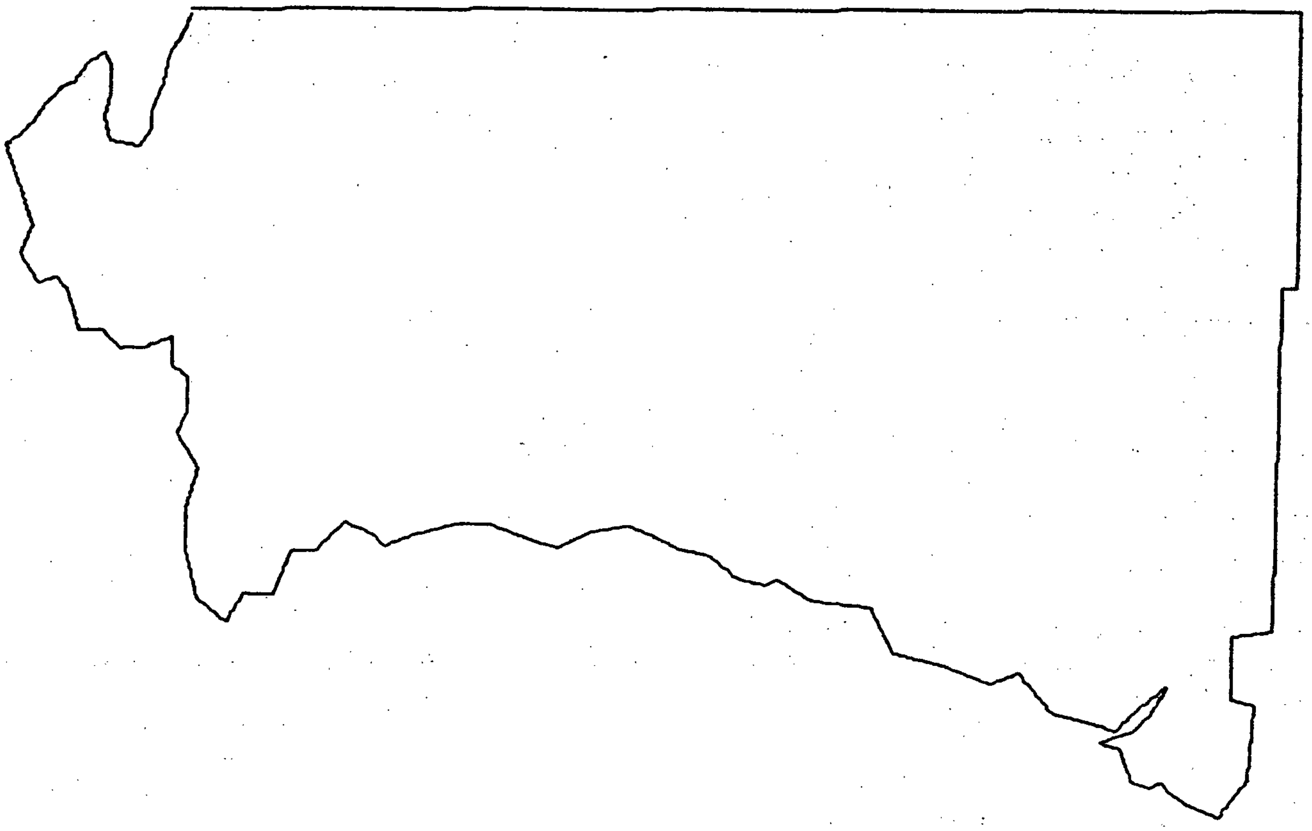
The remainder of this chapter is devoted to a discussion of experiments that were performed in attempts to answer these questions.

3.1 Perceptual Response to Generalization

A major purpose of this work is to provide a way of generalizing map outlines requiring a minimum of data storage while retaining enough of the essential components to be easily recognizable by people. It is therefore necessary to investigate how people respond to outlines that have to be generalized drastically. Is the rendition of prominent features the most important element of recognition or is it the maintenance of line character that is primary? What role does aesthetics play?

To help answer these questions two experiments were performed that involved people ranking a number of generalizations of a particular outline according to the degree to which they each resembled the original outline. Both experiments were conducted in the same way except that the outlines used were different. The original line, used in both cases, was the outline of the municipality of West Vancouver (see Figure 3.1) that was digitized at a scale of approximately 1:200000. A smaller scale version (see Figure 3.2) was plotted and mounted on a piece of opaque cardboard as were the various generalizations of this outline (see Figures 3.3 and 3.4). In each experiment the subject was given cards with the original outline (marked with a "1") and its generalizations (which were shuffled and marked with a letter) and asked to arrange these cards in order of the similarity to the one marked "1". Each subject was asked in the same way and were given no guidance on what "similarity" meant. The opaque mounting prevented overlaying the maps to make comparisons directly and instead forced the subjects to compare the different versions more by their individual appearance. The result of these rankings in the two experiments are recorded in Tables 3.1 and 3.2. Every trial within an experiment was performed by a different person although some individuals took part in both. These subjects hardly represented a cross-section of society or even potential interactive map users, coming, as they did, almost entirely from among my colleagues and friends. However, the diversity of opinion exhibited by this small sample is, I believe, indicative of the range that might be expected from a larger, less homogeneous group and is quite sufficient for my purposes.

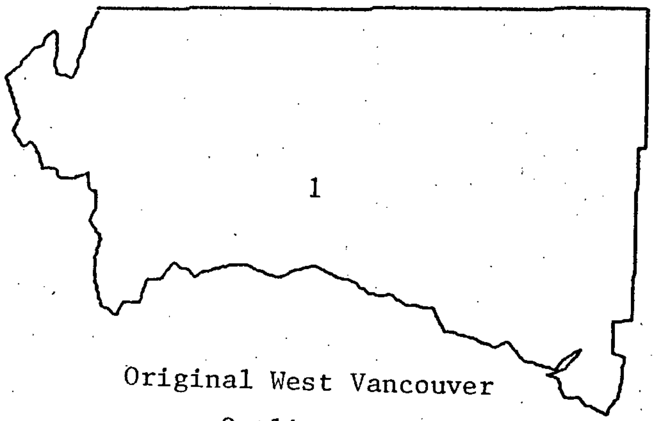
The outlines used in the first experiment were derived from



LEVEL=0 WEST_VAN

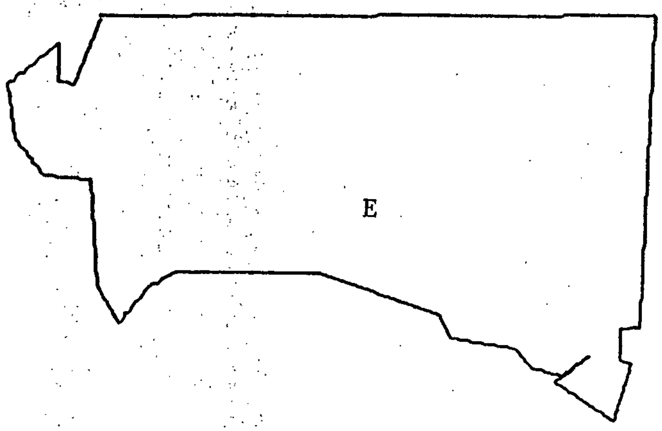
Outline of West Vancouver Municipality

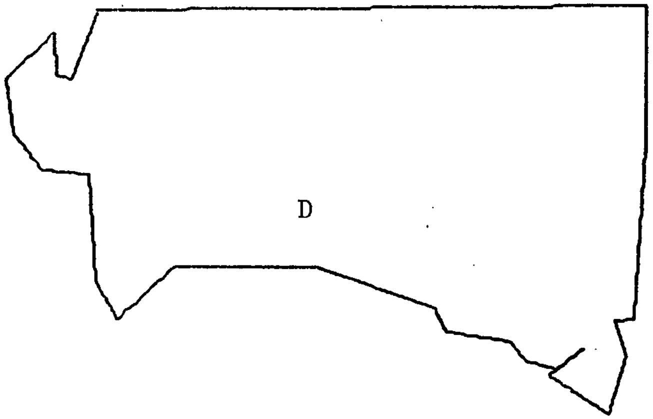
Figure 3.1

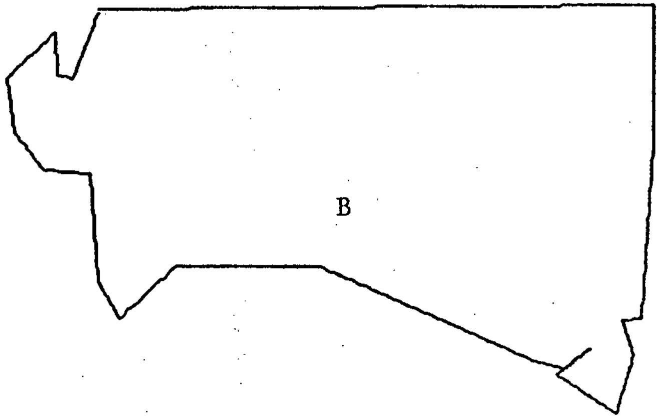


Original West Vancouver Outline

Figure 3.2

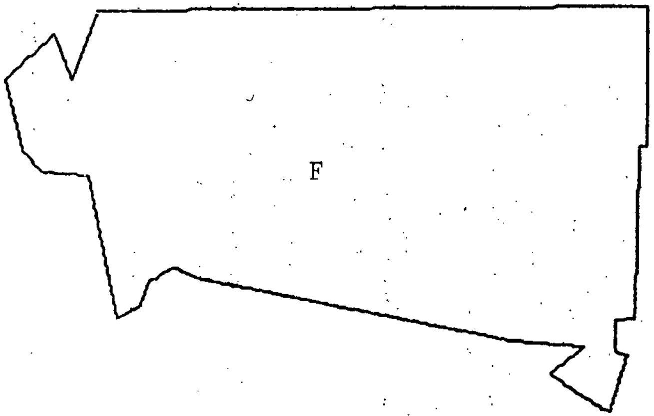


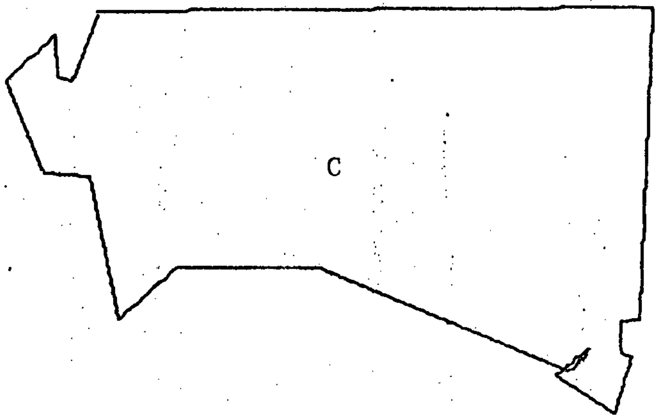


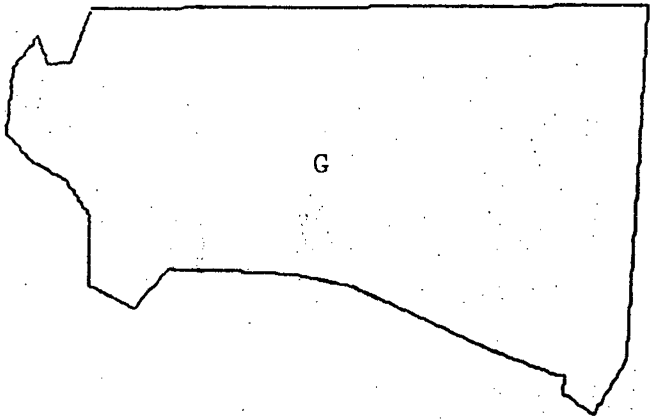


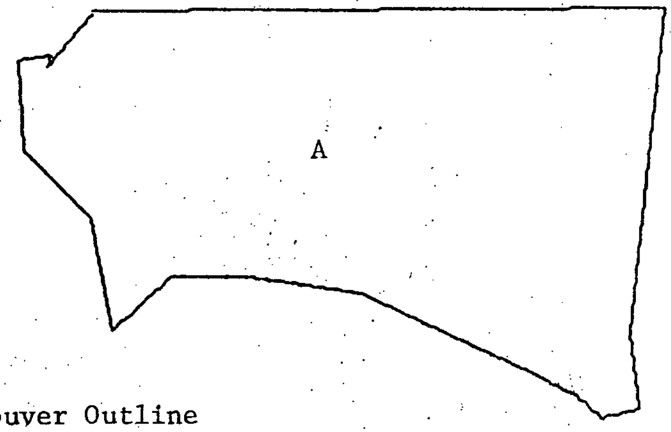
Versions of West Vancouver Outline for Experiment I

Figure 3.3



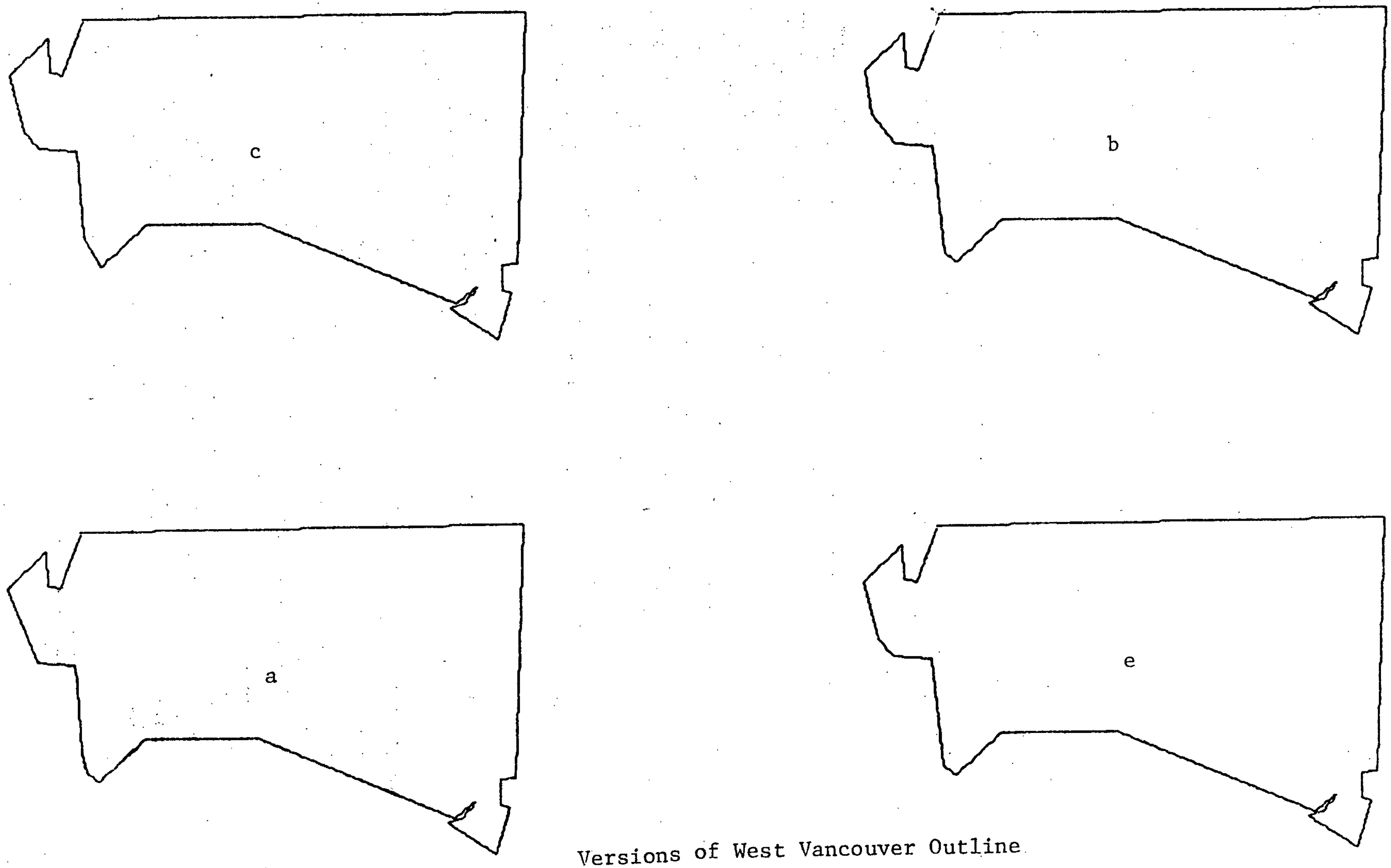






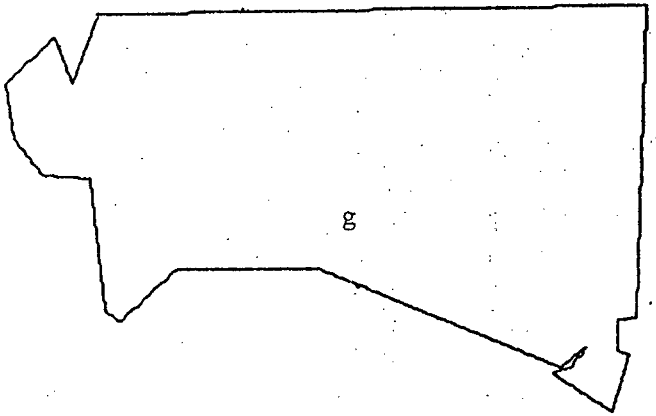
Versions of West Vancouver Outline
for Experiment I

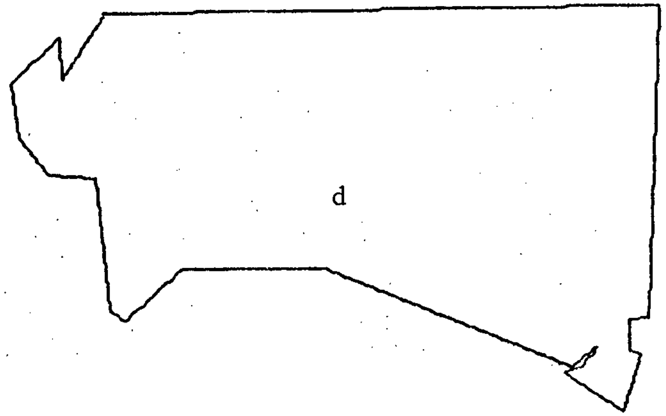
Figure 3.3(continued)

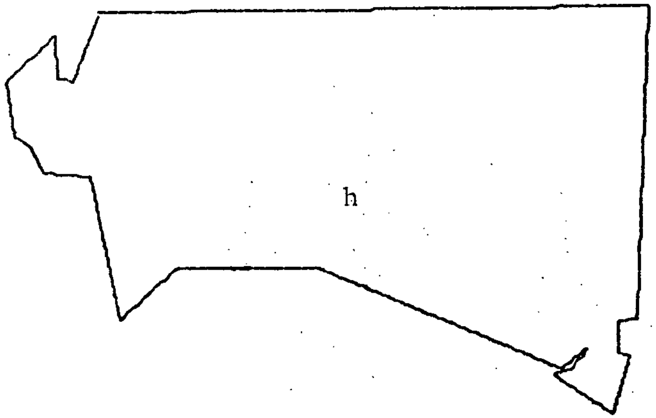


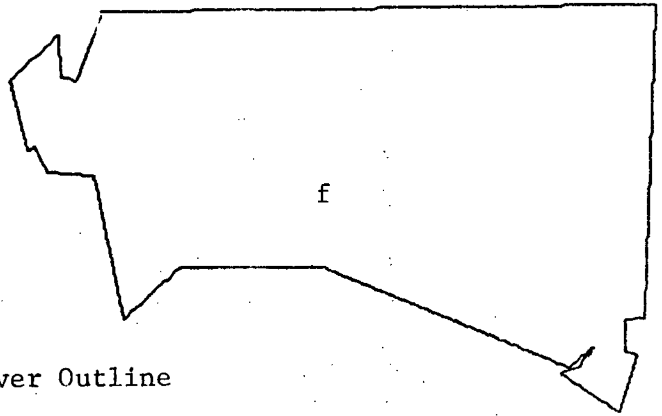
Versions of West Vancouver Outline
for Experiment II

Figure 3.4









Versions of West Vancouver Outline
for Experiment II

Figure 3.4(continued)

RESULTS FOR EXPERIMENT 1

Ranking of similarity to outline 1

Trial Number	High 1	2	3	4	5	6	Low 7
1	D	E	B	C	F	G	A
2	C	D	E	B	F	G	A
3	E	D	B	C	F	G	A
4	E	D	B	C	F	G	A
5	C	D	B	E	F	G	A
6	E	D	C	B	F	G	A
7	E	D	B	C	F	G	A
8	E	D	B	C	F	G	A
9	E	D	B	C	F	G	A
10	E	D	C	B	F	G	A

TABLE 3.1

RESULTS FOR EXPERIMENT 2

Ranking of similarity to outline 1

Trial Number	High 1	2	3	4	5	6	7	Low 8
1	h	f	b	e	c	a	d	g
2	b	c	e	g	d	h	a	f
3	a	c	e	b	g	d	f	h
4	h	b	e	c	f	d	g	a
5	e	f	b	a	c	g	d	h
6	d	b	e	g	c	a	h	f
7	b	c	d	e	g	a	h	f
8	b	e	a	c	f	h	d	g
9	b	d	e	c	g	a	h	f
10	f	h	a	c	d	b	e	g
11	b	e	h	a	f	c	g	d
12	b	c	e	h	a	g	d	f
13	d	g	e	b	a	c	h	f
14	f	h	b	a	e	c	g	d

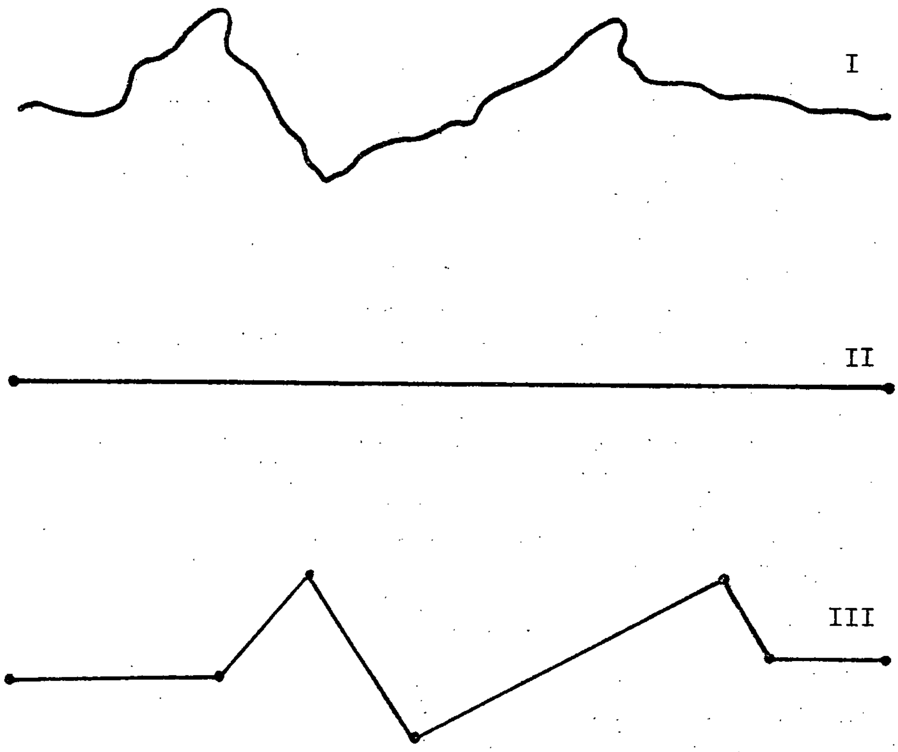
TABLE 3.2

the original in a variety of ways. The general aim was to reduce the number of points to about 23, one quarter of the number of points in the original (which is 92). Following are details of the ways in which the points were selected for the various outlines:

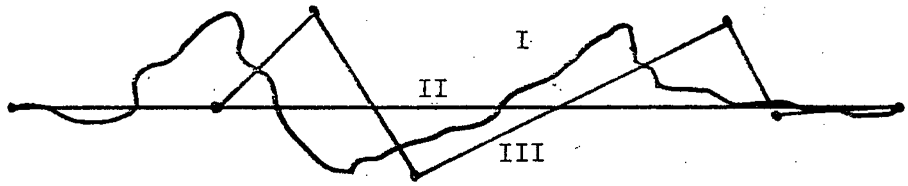
- Outline A - Randomly. Every point was considered in turn and was selected with a probability of 0.25. 22 points.
- Outline B. Douglas #1. The method of Douglas (see Chapter 1) was used with a threshold deviation of 0.08". 23 points.
- Outline C. Hand-picked. The points were selected by hand to ensure that the small inlet in the lower right-hand corner remained open. 21 points.
- Outline D. Douglas #2. The same as for outline B except that the threshold was reduced to 0.06". 26 points.
- Outline E. Lang. Points were selected using a slight variation of Lang's method (see Chapter 1) with a threshold distance of 0.07". 27 points.
- Outline F. Largest angles. The points that had the 23 largest angles (i.e. greater than 66°) of bend were selected. 25 points.
- Outline G. Every n'th point. Every fourth point was kept. 24 points.

Note: the points in the upper-left and upper-right hand corners of the map were automatically kept to avoid gross distortions in the cases where the method used to obtain the outline did not specify they be kept. This was required with outlines A and G. There are also small wiggles in these outlines that do not correspond to actual points. They arise from the incremental nature of the Calcomp plotter that was used.

By referring to Table 3.1 we can see that the average order of preference is EDCBFGA. This order was calculated by adding up the position numbers of each letter for each trial. These totals are 16, 19, 32, 33, 50, 60, 70 respectively. Generally people conformed to this sequence, but it is the deviations that are interesting. C and B are almost tied and in fact B was preferred by more people than C. However, when C is preferred it is by a greater margin. This is probably due to the fact that the small inlet in the lower right hand corner was left open while in B, D and E it closed off. This supposition is borne out by comments made by the subjects after they had finished ranking the outlines. Some people said that they regarded the accurate rendition of the lower right corner as the overriding factor while others said the details along the left side were more important. Some but not all regarded the maintenance of the bumpy character along the bottom center as an important consideration. The actual accuracy in terms of the total (or integrated) deviation from the original probably was not a crucial factor as some people's preference for B over D and D over E indicates. It is quite likely that people would prefer line III to line II as a representation of line I (see Figure 3.5) although superimposing them as in Figure 3.6 shows that



A Line and Two Possible Generalizations   Figure 3.5



All Three Lines Superimposed   Figure 3.6

line II obviously deviates much less.

Generally we can conclude from this experiment that different people see maps in different ways and that they respond to different aspects of them when making comparisons. There is little we can say about how to generalize outlines satisfactorily. This is partly because the outlines used vary in the number of points along their boundaries. Generally, of course, the more points one has available the better the rendition possible. What would be of interest is to see people's preferences among various versions of an original map that all have the same number of points. It was with this aim in mind that a second experiment was designed and performed.

The outlines used in the second experiment were all derived in roughly the same way. In all of the 8 generalized outlines there is the same basic set of 19 points together with four other points chosen from an additional fixed set of 7. [1] Thus all outlines have 23 points (except d and g) and are fairly similar.

The average ranking, calculated in the same way as before, is becahdfg and the totals are 33, 46, 57, 68, 69, 73, 76, 82 respectively. Certain lines are rated fairly consistently, such as b, e and c, while others such as h, f and d are more controversial. This can be seen by inspection of Table 3.2 or more rigorously by adding the deviations in each trial from their mean positions. This gives totals of 19, 19, 23, 22, 33,

---

[1] Except for d and g which have only 3 of the possible 7.

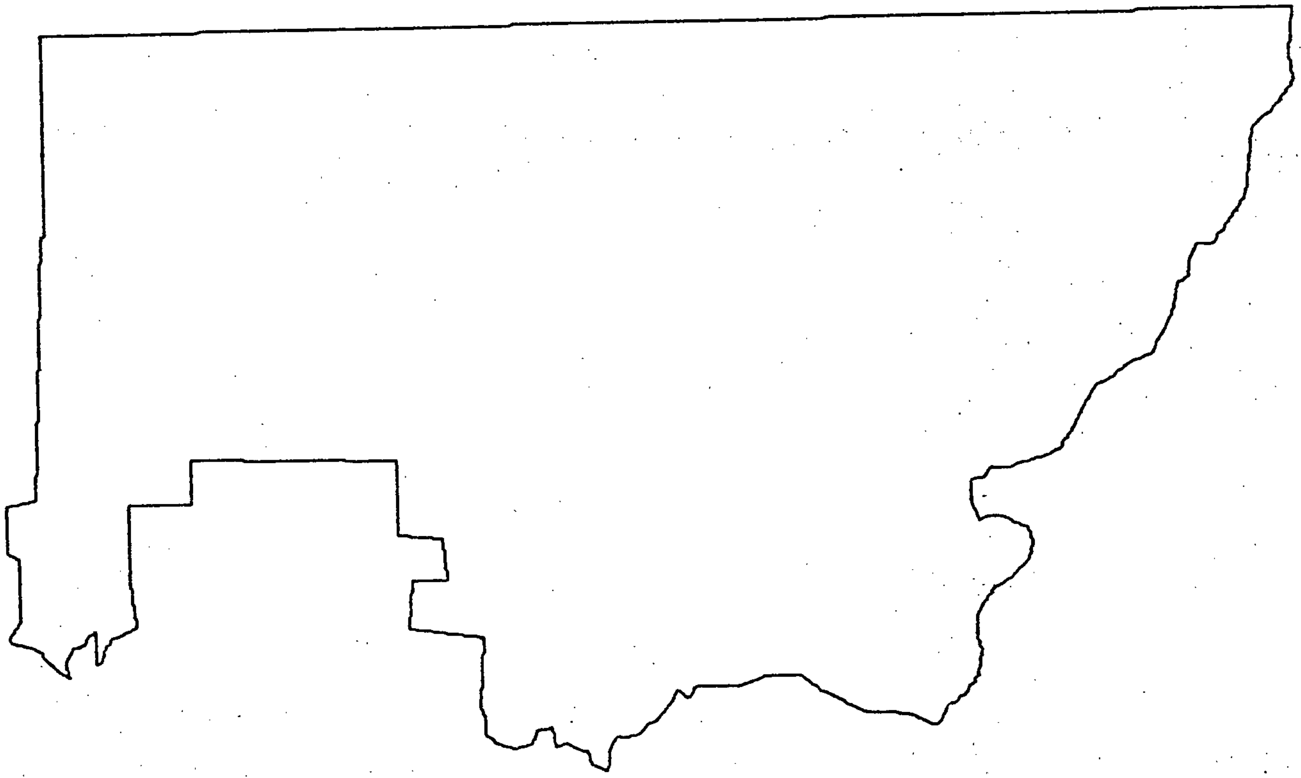
27, 34, 30 (where the order is the same as before). By looking at the individual outlines (see Figure 3.4) we can partially account for this. Most people do not like the sharp point at the lower left but there are some who do not mind. The character of the line in the middle of the left is important to some while for others it is the overall shape. The comments people made later to justify their ranking support these views. Every person saw some particular aspects of the shape to be most important while other aspects were relatively unimportant. One person said that he was more interested in bays than in peninsulas and attributed this to the fact that he looked at a coastline from the point of view of the land and not the sea. The tone and expressions used by people also indicated that their perception was quite subjective. Often they were "bothered" by the presence or absence of particular features as if they were reacting on aesthetic grounds as well.

I think that this experiment shows even more clearly than the first that people have quite different ways of looking at outlines and that a single technique for doing generalizations will not satisfy everybody. Some people will like to see character preserved, others particular features, and still others the overall shape. This suggests that a method that can be suited to an individual's tastes and preferences would enjoy an advantage over less flexible methods. It is the aim of the work described here to provide such a flexible system.

3.2 Local Reduction

It is generally the case that computer processing of information is more easily and conveniently performed on a local rather than global basis. Computer architecture and existing software discourage making decisions using information that is widely separated. It is usually much simpler and cheaper to make decisions based on information that is restricted spatially. In this sense this work is no exception. However, it is not clear *a priori* that local considerations will be adequate to perform generalization satisfactorily. Even if we do not need fully global considerations it is not obvious what point on the local-global continuum is most appropriate for our purposes. Clearly, making the decision to select or reject a point depending only on the angle at that point will in general be unsatisfactory, but how much more of a line does a program have to look at? We can get some estimate of a lower bound on the required view by observing how well people perform at manually reducing outlines when all that can be seen at one time is a small section. (If a person cannot do it, then probably neither can a machine.) An experiment to do just this was set up and performed.

Basically the experiment to investigate the adequacy of a local view involved a number of subjects manually reducing an outline of North Vancouver Municipality (see Figure 3.7) in the manner described in Chapter 2. A section of the outline containing at most 7 points was displayed on the screen with a small "X" at one of the points. The person would then press one of two buttons depending on whether that point was to be kept or



Outline of North Vancouver Municipality

Figure 3.7

LEVEL=0 NORTH_VAN_DISTRICT

removed. If the point was removed then the line was re-drawn without that point. In either case the "X" moved to the next point. If the "X" moved to second to the last point, then the line was re-drawn adding the next three points and keeping the last four. In this way every point along the original outline was considered in turn and with no fewer than 2 points on either side. The scale at which the individual fragments were displayed was the same for every fragment and was chosen so that if the whole outline were displayed it would just fill the screen. The individual fragments were also centered in the middle of the screen to avoid giving spatial clues.

The task of generalizing a line under such circumstances requires considerable skill and I suspected (based on my personal experience) that the performance would depend on practise. In order to give the subjects some opportunity to learn how to do it they were given the outline of West Vancouver to start with. They were shown first all of this outline on the screen and asked to aim for a reduction of three-quarters (i.e. leaving about 23 points). The fragments were then displayed in turn and points eliminated. The reduced version was then displayed and the number of points remaining counted. This gave the person an opportunity to see how much more ruthless they would have to be in pruning points from the North Vancouver outline. As can be seen from Table 3.3, people were not very successful in achieving the three-quarters reduction. The next step was the reduction of the North Vancouver outline using only the small fragments without seeing the whole thing first and without knowing what it was. The result of these reductions for

RESULTS FOR EXPERIMENT 3

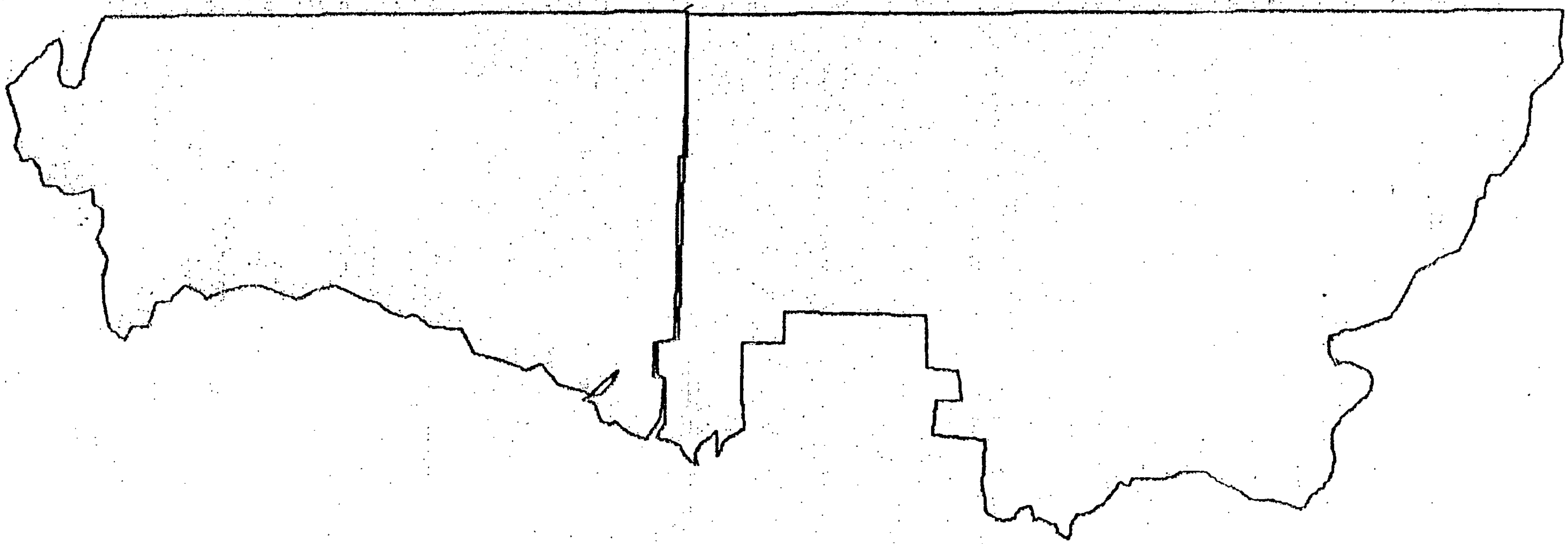
Trial Number	Number of Points Remaining West Vancouver (92 points originally)	North Vancouver (134 points originally)
1	44	45
2	47	47
3	47	47
4	38	44
5	56	55
6	46	33

TABLE 3.3

both maps are shown in Figures 3.9 to 3.14 with the originals shown at the same scale in Figure 3.8.

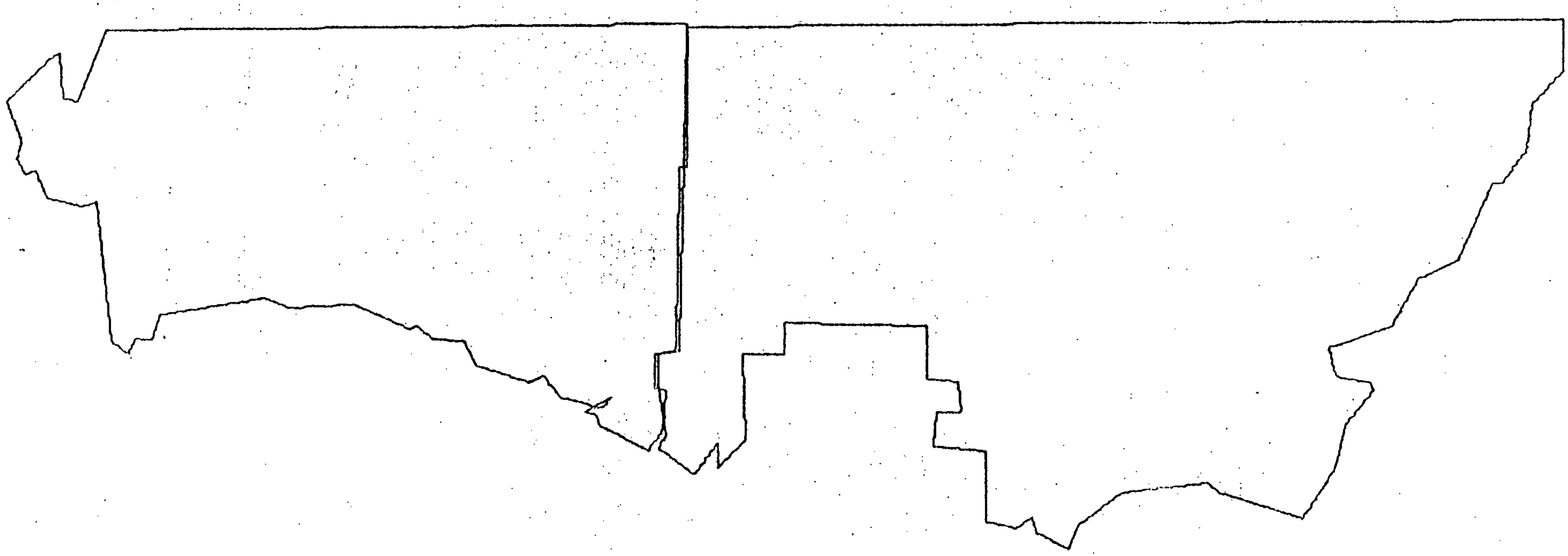
In general terms we can see that the generalization was done fairly well although there are some glaring mistakes. Almost all the main features are captured and the character of the coastline on the right-hand is maintained. Referring to this series of figures and to Table 3.3 it is clear that the subjects did learn to be more drastic in their reduction. The average reduction for West Vancouver was slightly over one-half of original 92 points while with North Vancouver the average reduction was slightly above one-third of the original 134 points. This was done in spite of the large number of long segments in the lower left corner. One person in particular (on Trial 6, Figure 3.14) became excessively concerned about the desired degree of reduction and removed a large portion from the lower middle although the remainder of the line was done quite satisfactorily. This person actually left less than one quarter of the points remaining.

In order to more fully answer the question posed at the beginning of this section much more experimentation would have to be done. It would be interesting to see how people's performance depended on the size of the view available, the desired degree of reduction, and on their experience doing generalization in this manner. The effect of the character of the outlines used is also an important factor that would have to be investigated. Relatively smooth lines will present different problems than lines in which there are large angles between



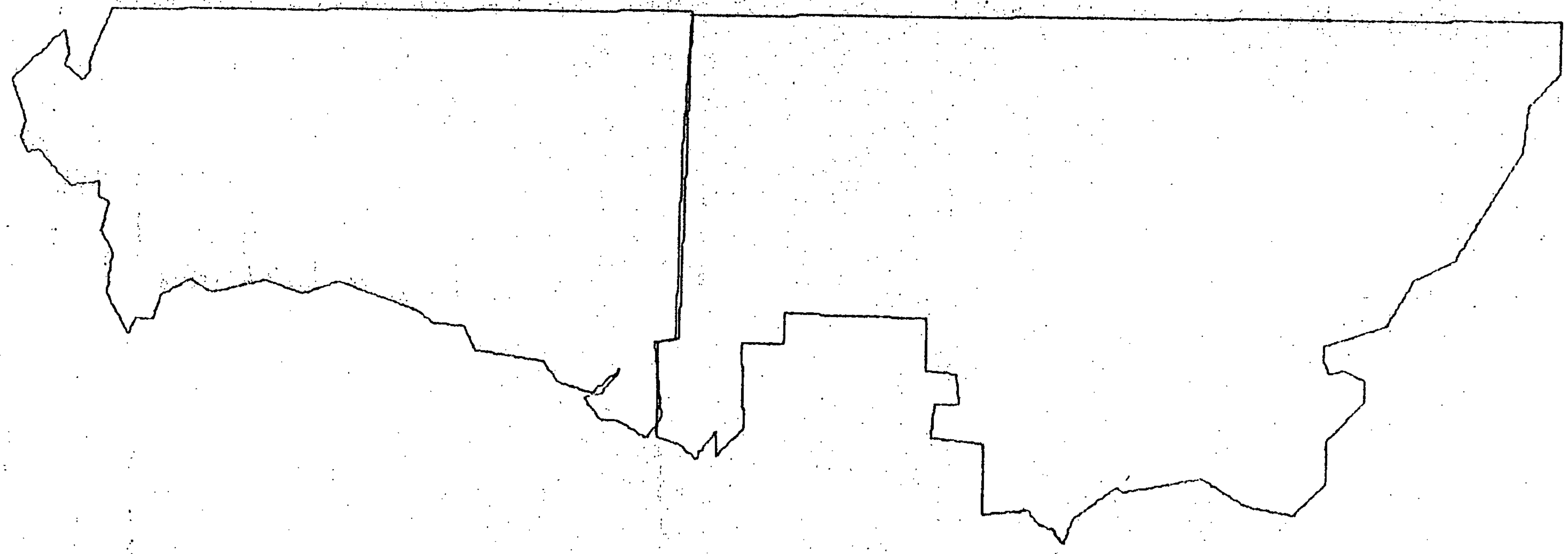
Original Outlines of West Vancouver and North Vancouver

Figure 3.8



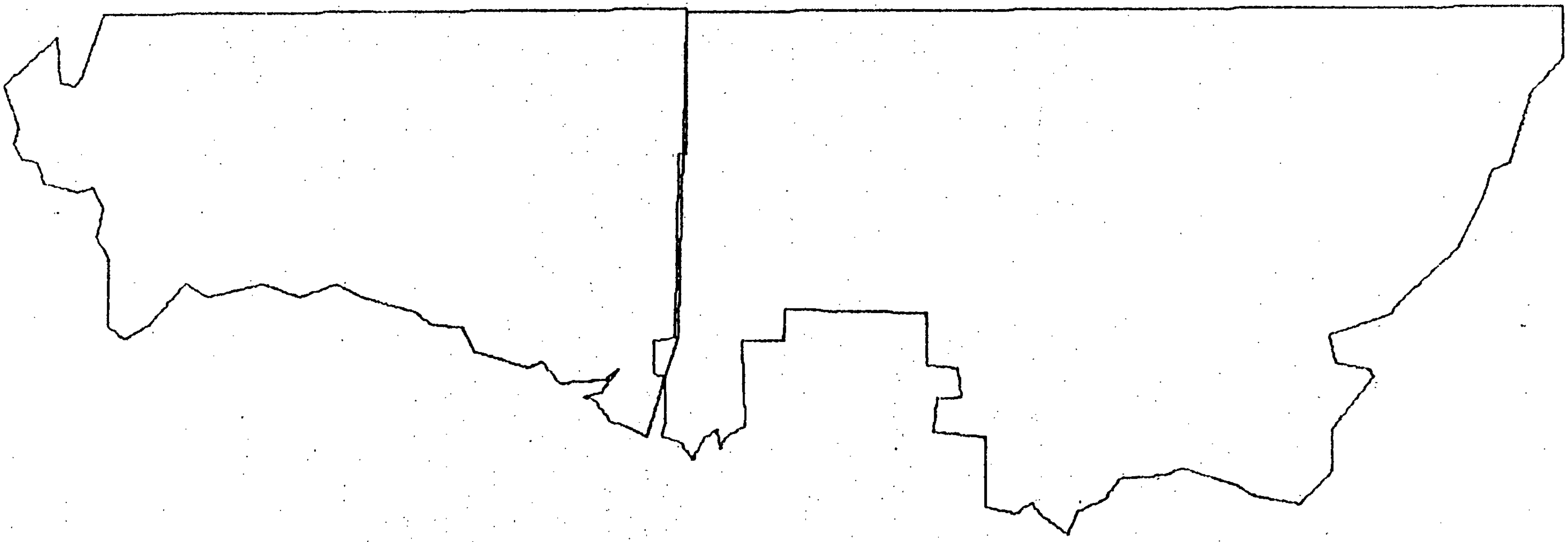
Reduced Outlines, Trial 1

Figure 3.9



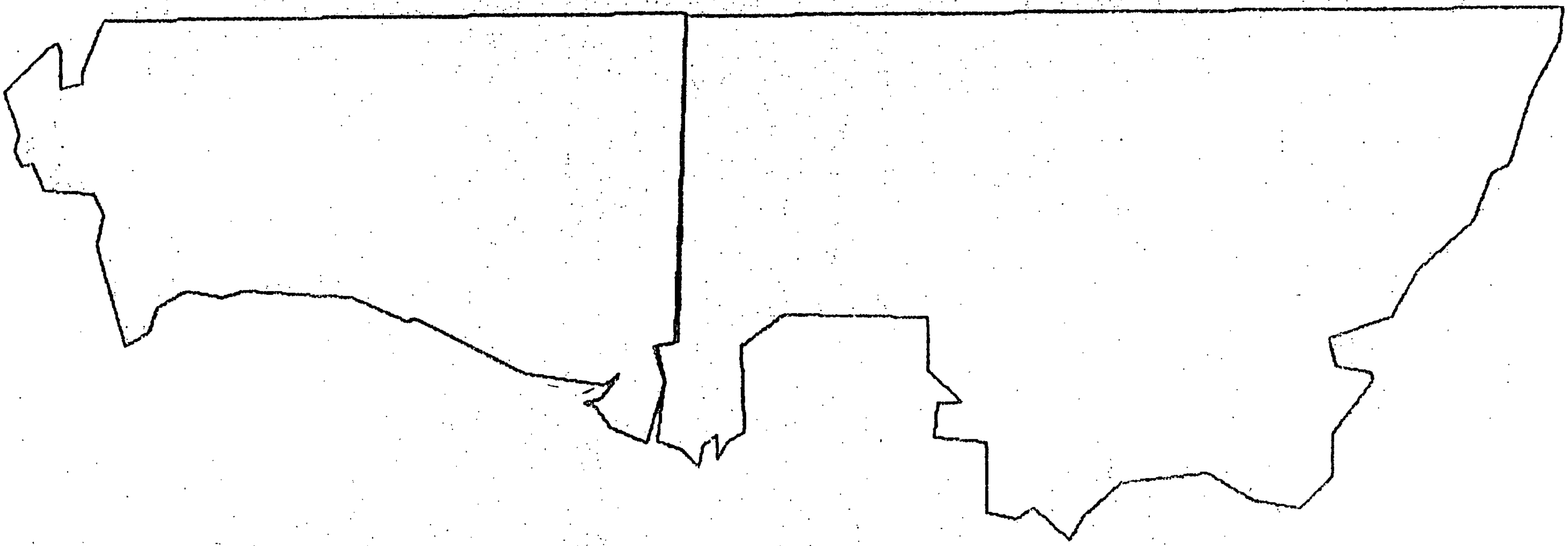
Reduced Outlines, Trial 2

Figure 3.10



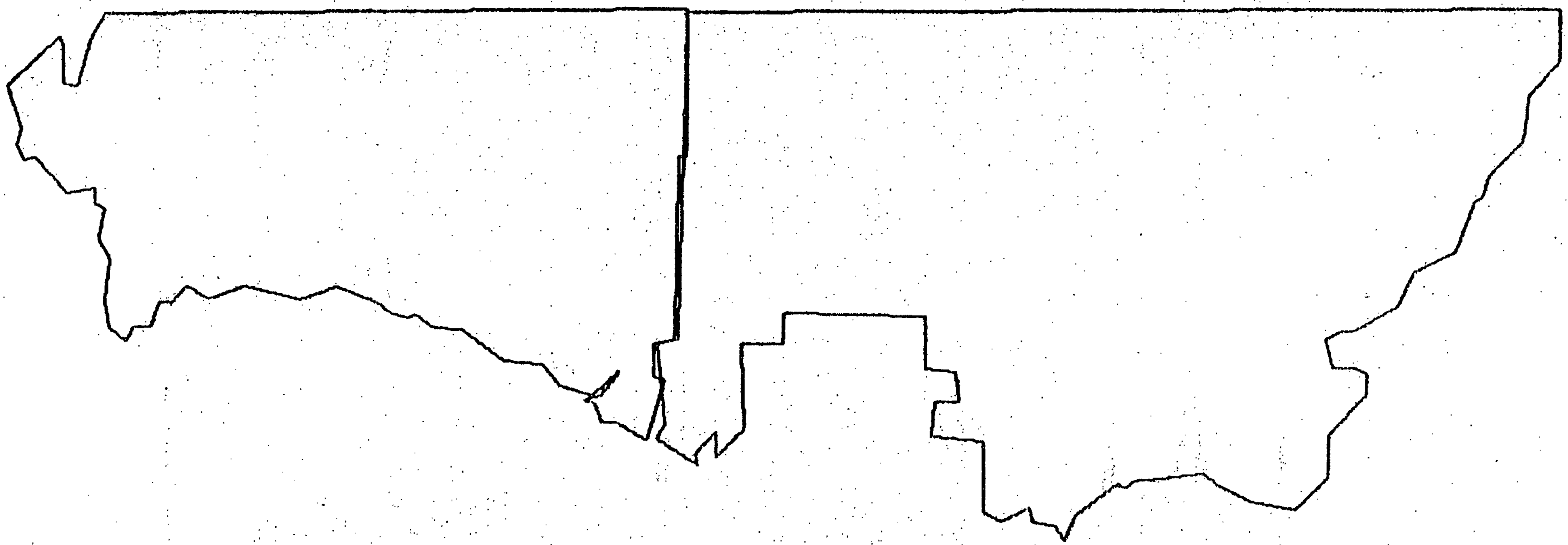
Reduced Outlines, Trial 3

Figure 3.11



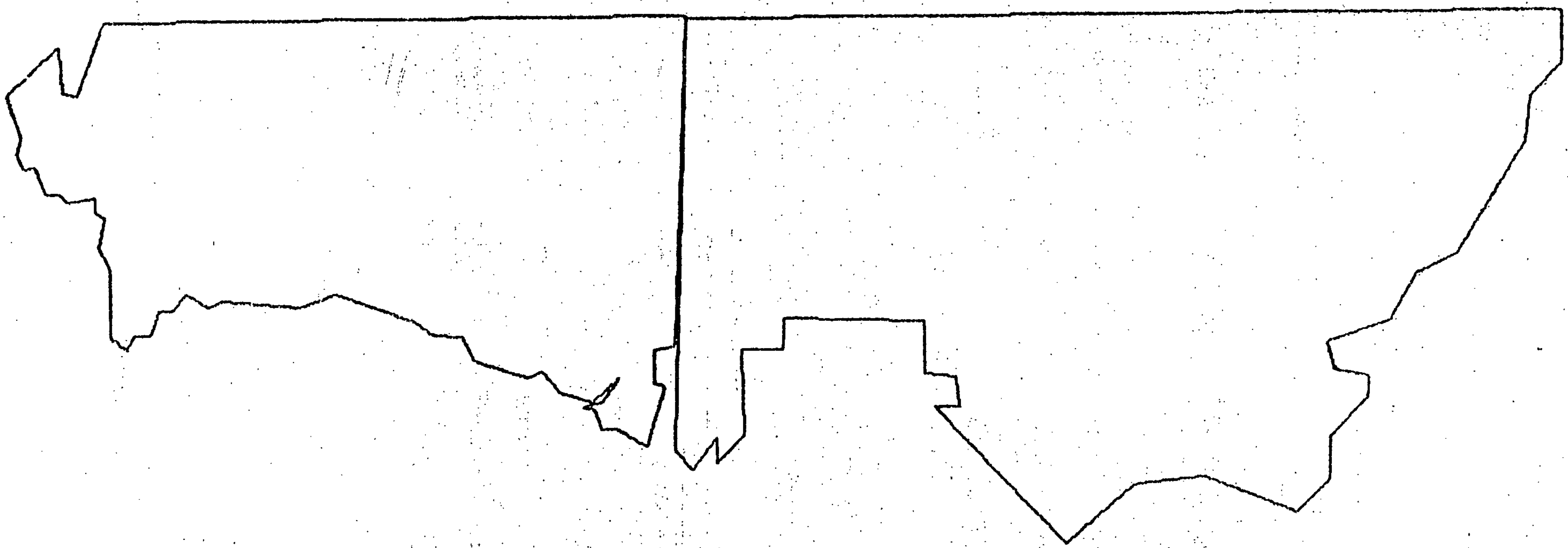
Reduced Outlines, Trial 4

Figure 3.12



Reduced Outlines, Trial 5

Figure 3.13



Reduced Outlines, Trial 6

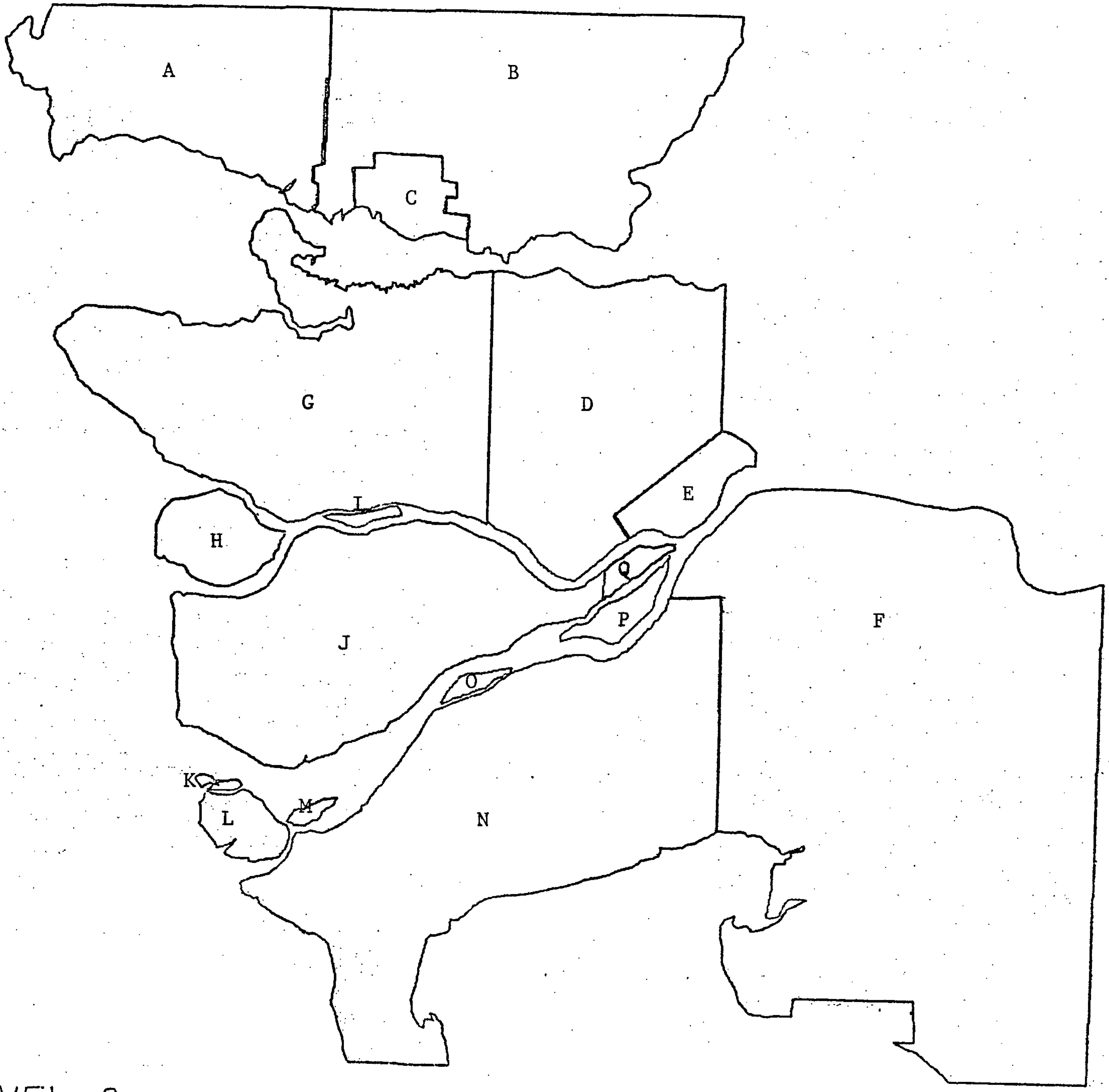
Figure 3.14

consecutive vectors. However, in spite of the limited extent of the testing that was done there is sufficient evidence to suggest that it is not hopeless to expect a program operating only on a local basis to perform generalizations satisfactorily.

3.3 The System's Generalization Performance

The third major question that was asked at the beginning of this chapter was how well can the system described here learn to generalize map outlines. Since the system generalizes by first learning to mimic a set of previously reduced outlines, the question can be broken down into two parts: How well does it mimic and how well does it do on outlines it has never met before?

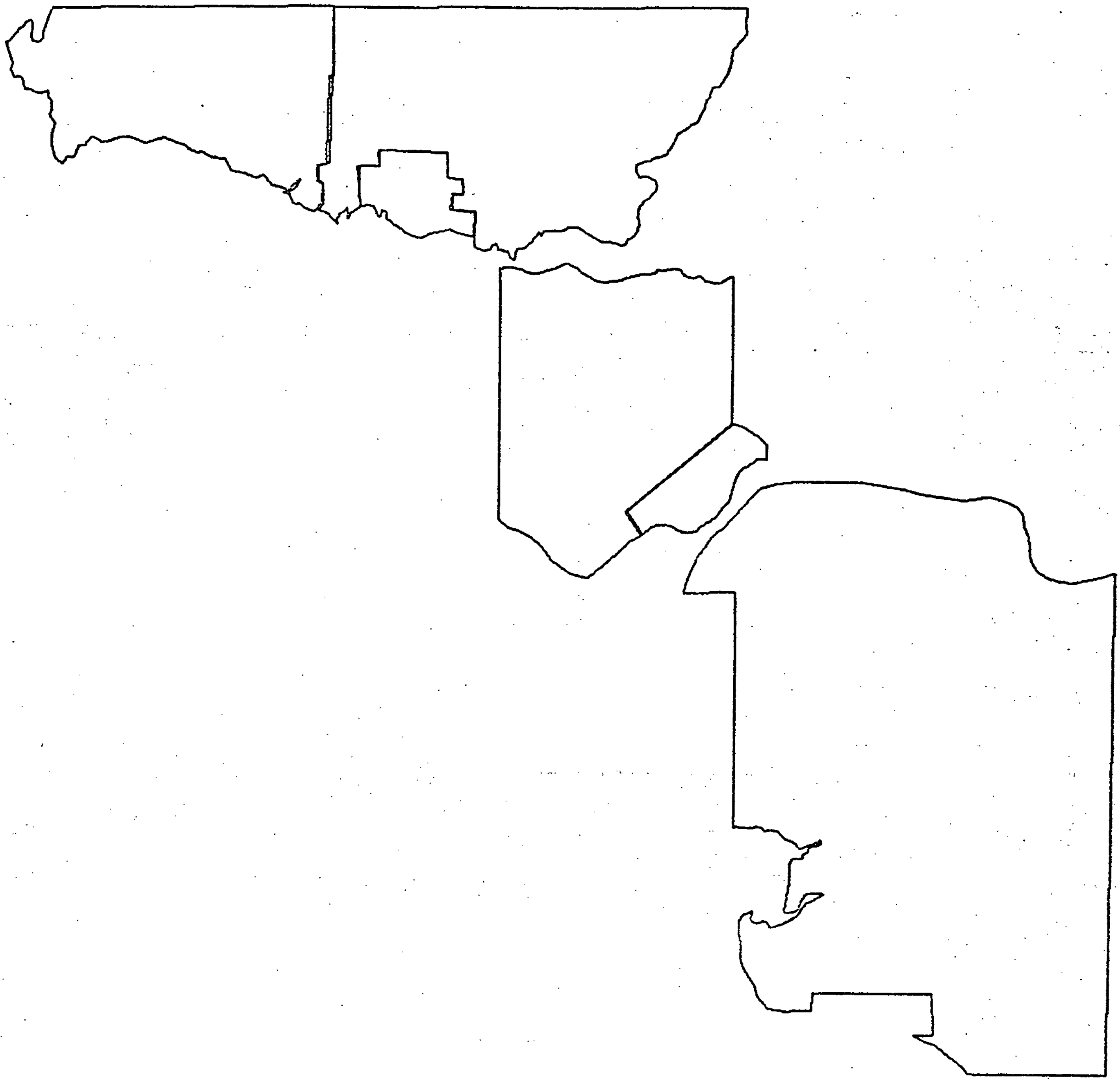
The first step in attempting to answer these questions is the selection and manual reduction of an initial set of outlines on which the system is to learn. In our case these lines were chosen from the digitized boundaries of some of B.C.'s Lower Mainland Municipalities (see Figure 3.15). The outlines labelled with the letters A through F were chosen to be the basic learning set. The reason that these particular lines were chosen rather than others is simply that they spanned the region of the map and that they appeared to contain a variety of features and line types. These lines were reduced by hand so that displayed at level 0 we have Figure 3.16 and at level 8 Figure 3.17. This reduction was aimed to leave as close to one quarter of the original number of points in each outline as possible without excessive distortion. The original and remaining number of points for each line is shown in Table 3.5. This basic learning



EVEL=0

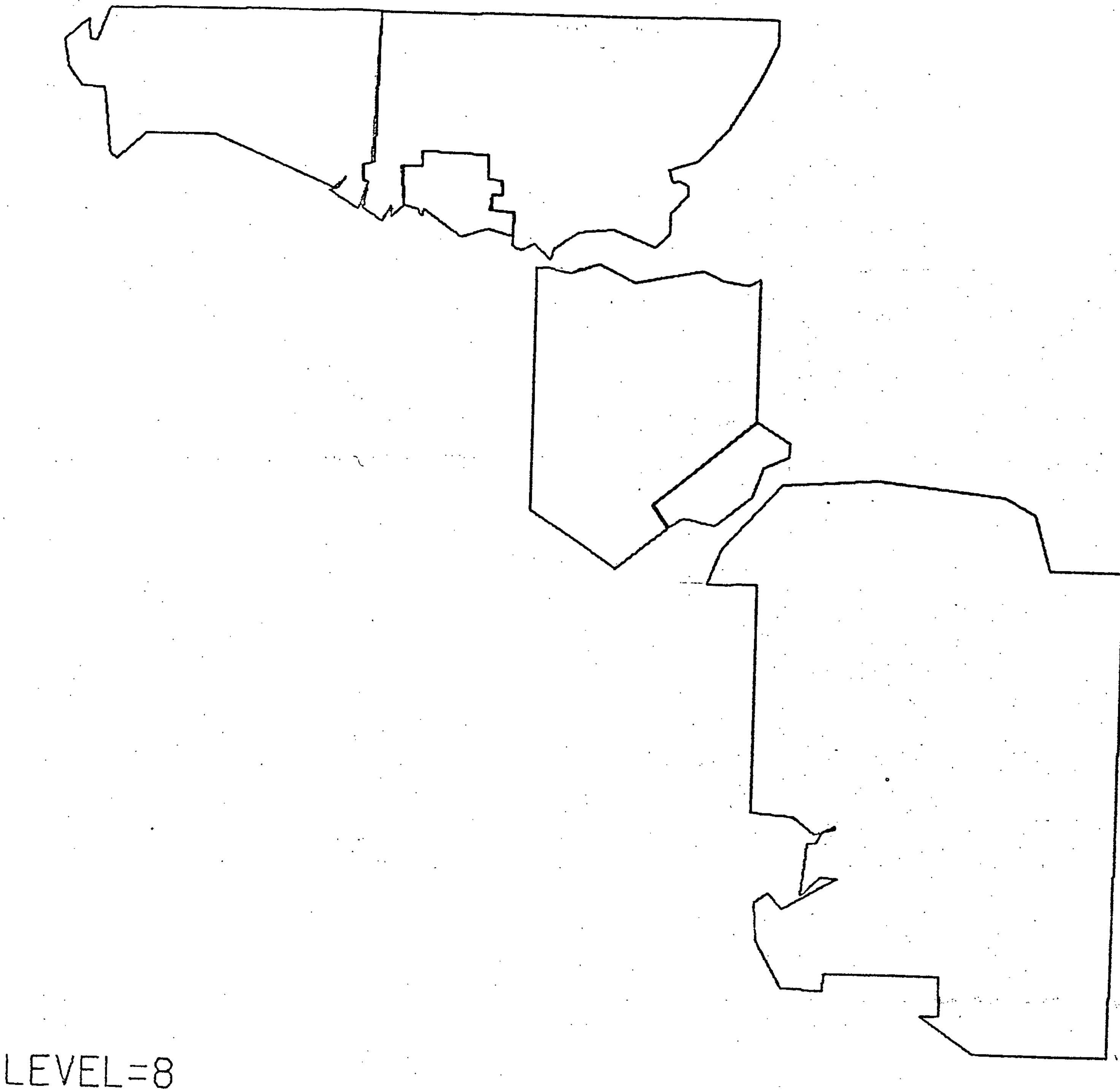
Outlines of LowerMainland Municipalities

Figure 3.15



LEVEL=0 Original Outlines for Learning

Figure 3.16



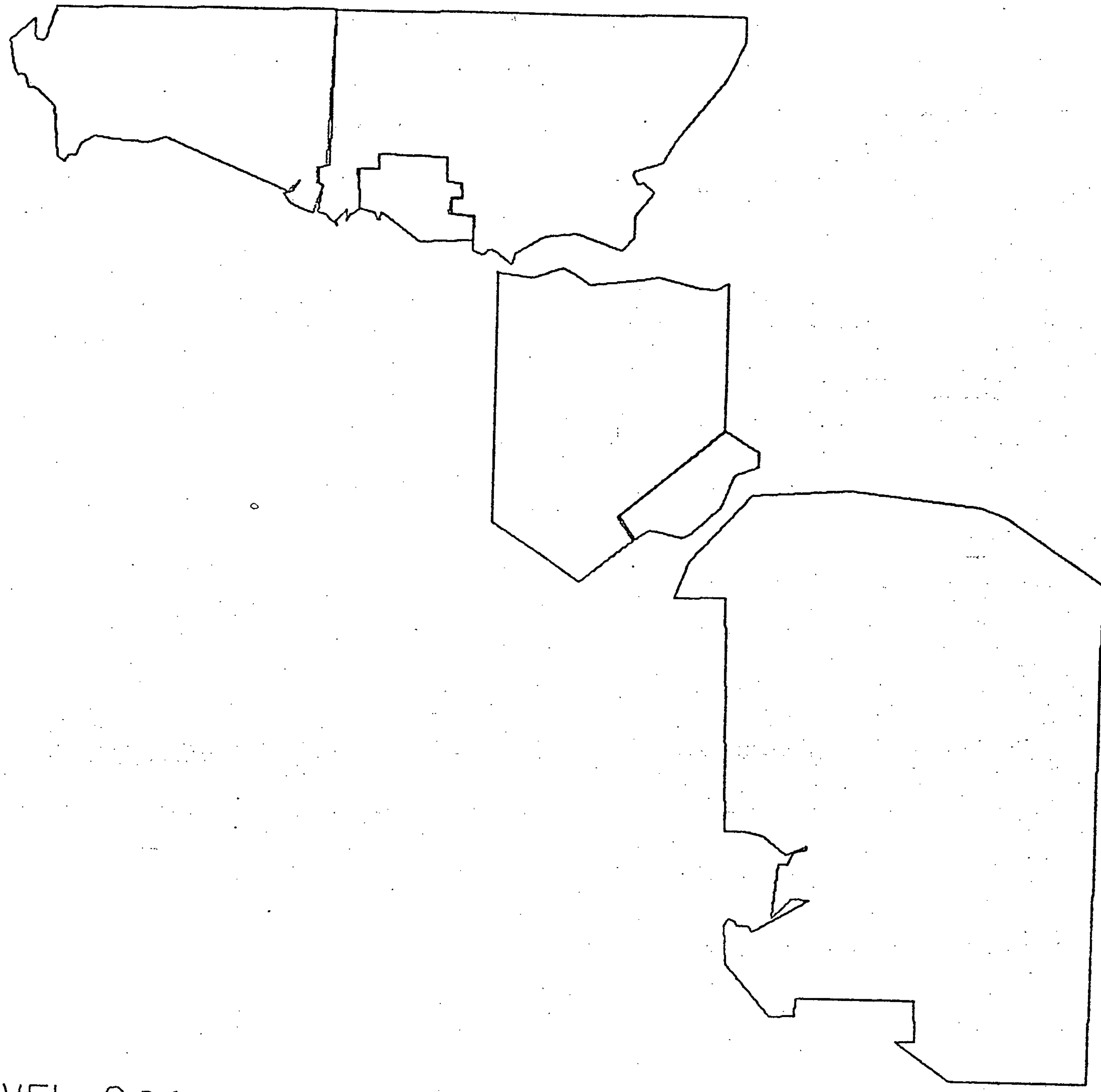
Reduced Outlines for Learning   Figure 3.17

set of outlines was then fed into the learning component of the system in the manner described in Chapter 2 with a point threshold level of eight. The angle and length parameters were standard [1] and the maximum allowable depth in the decision tree was twelve, thus permitting a view of the line about the same as was permitted for the experiment described in the previous section. This learning procedure was performed five times in succession to reinforce the message. After each iteration statistics were produced to indicate how many terminal node verdicts were made stronger, weaker or changed and how many times new nodes were added to the decision tree (see Table 3.4). In addition, after each iteration, the system was given the learning set at level zero to generalize using its current knowledge. The resulting outlines are shown in Figures 3.18 through 3.22 and the numbers of points in each of these outlines is given in Table 3.5.

There are several ways of using the results mentioned so far to answer the question of how well the system can mimic a person's generalization behavior. The first step is to ensure that the system is actually capable of duplicating the person's performance. This clearly is the case as can be seen by comparing the results after five iterations (Figure 3.22) with the model lines shown in Figure 3.17. The only really discernable difference occurs in the upper right corner of outline F and even then there can be doubt as to whether the

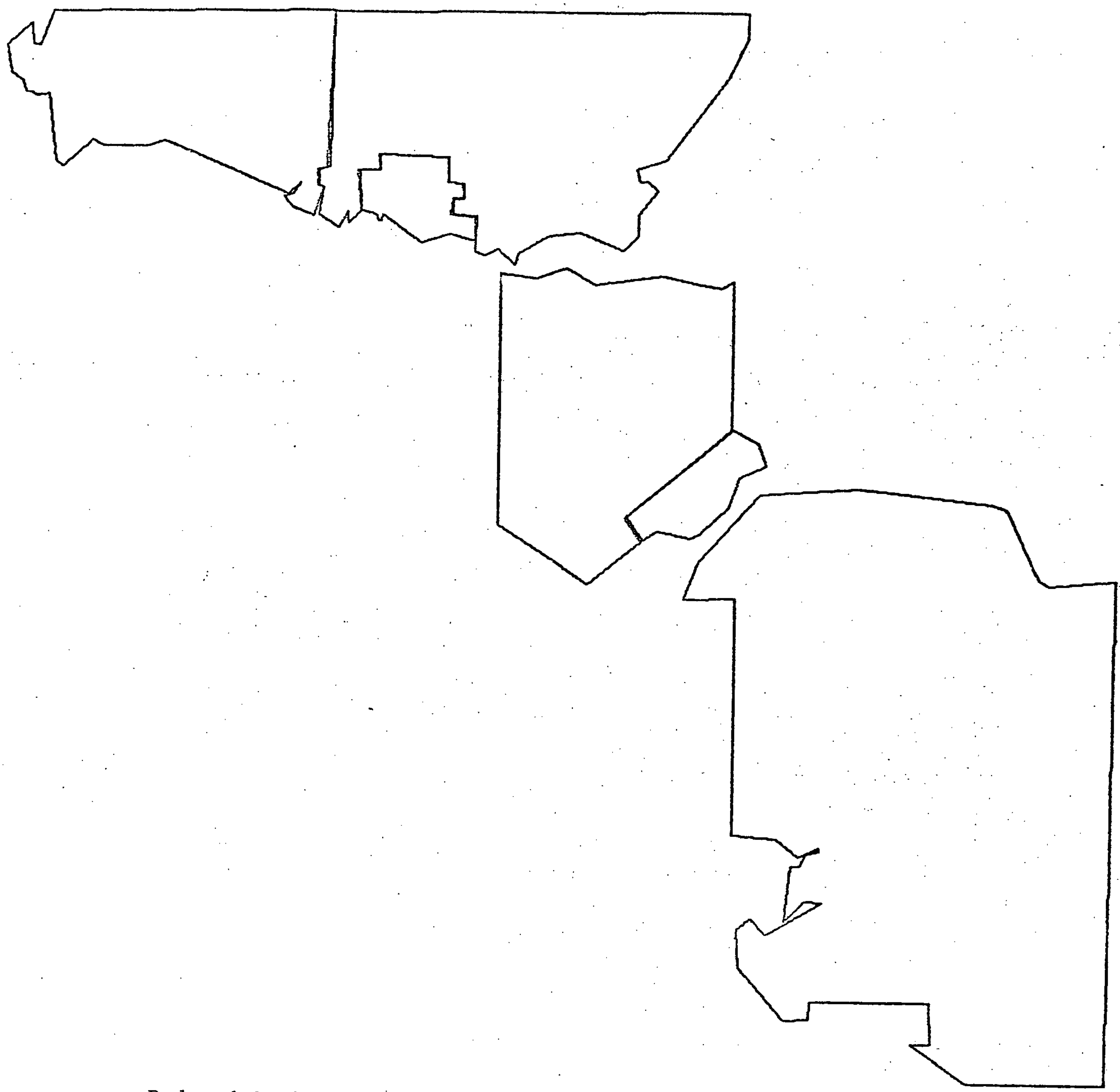
---

[1] ie. The angle thresholds (in radians) were 0.65, 2.15, 2.80 and 3.05 respectively and the minimum and maximum lengths were 0.01" and 0.5" at the scale of Figure 3.15



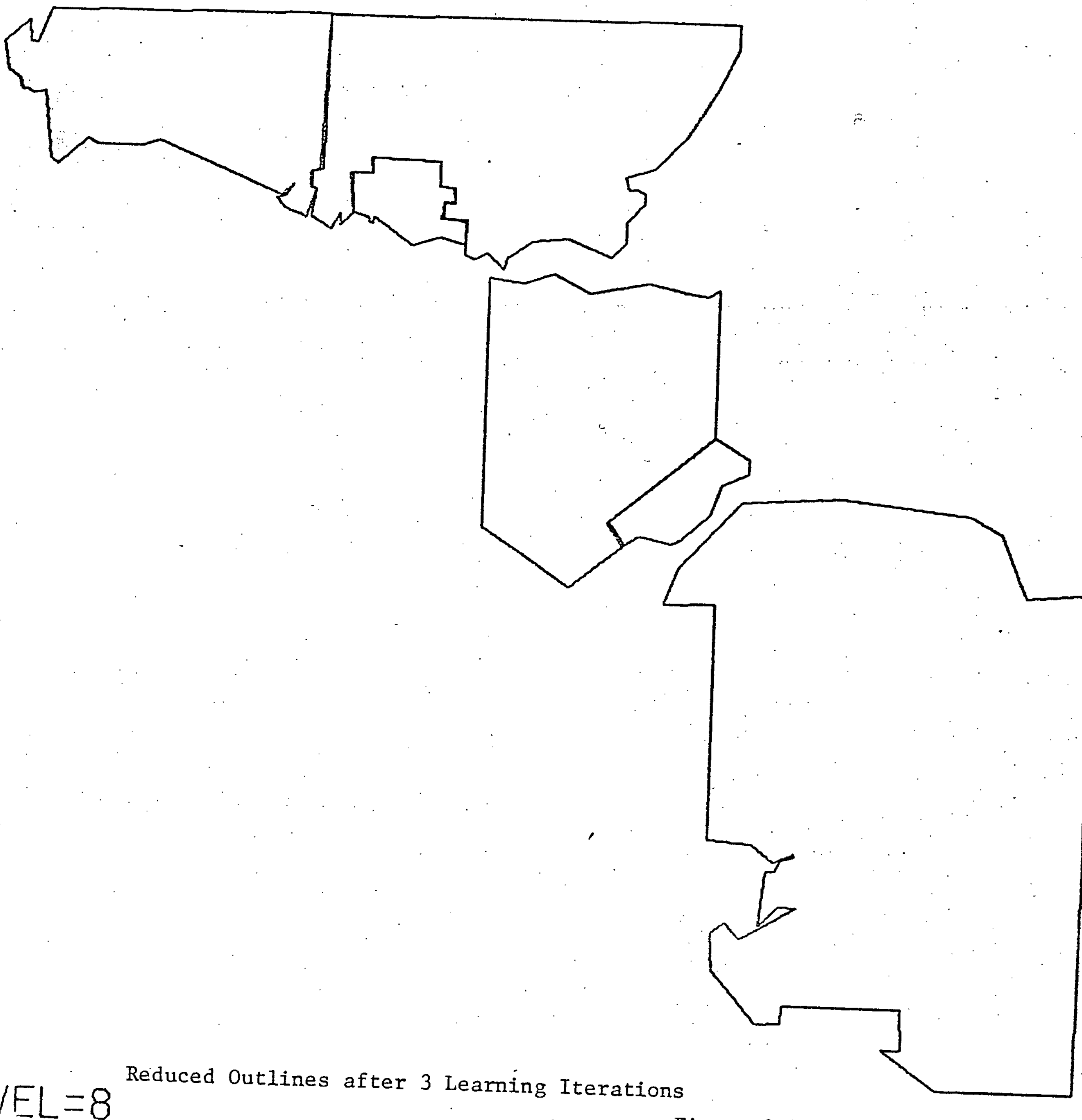
LEVEL=8 Reduced Outlines after 1 Learning Iteration

Figure 3.18



LEVEL=8 Reduced Outlines after 2 Learning Iterations

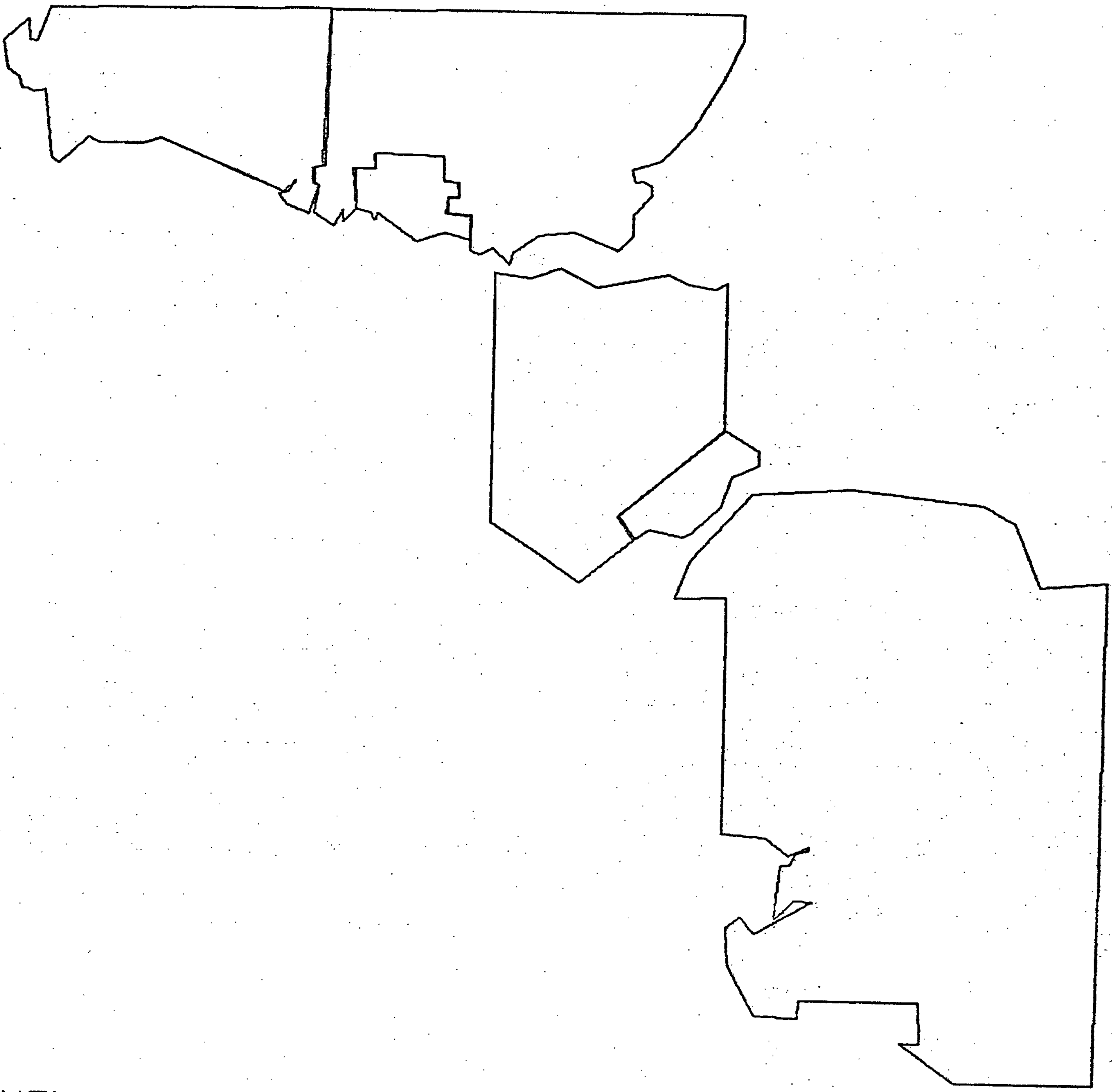
Figure 3.19



LEVEL=8

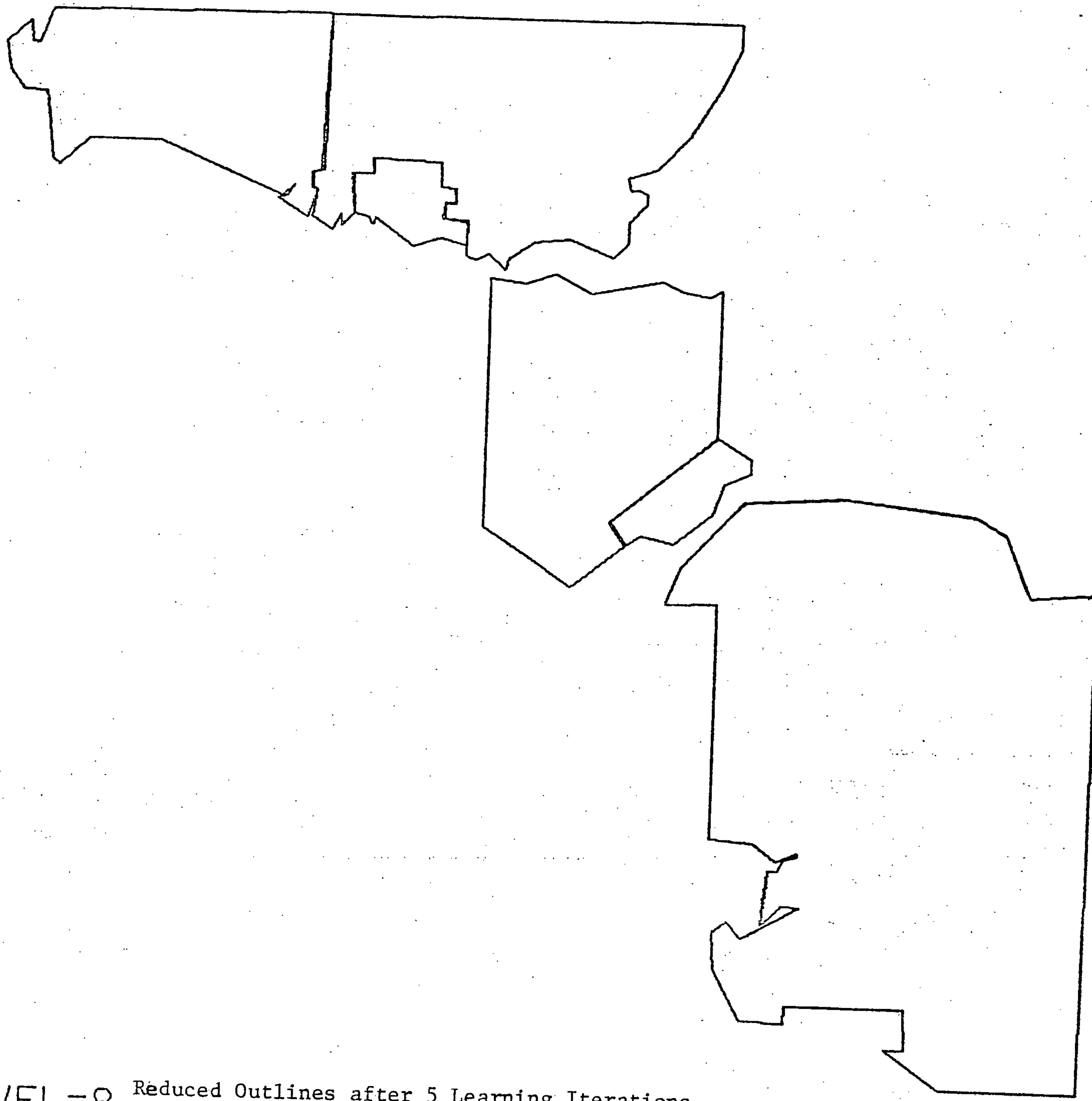
Reduced Outlines after 3 Learning Iterations

Figure 3.20



LEVEL=8 Reduced Outlines after 4 Learning Iterations

Figure 3.21



LEVEL=8 Reduced Outlines after 5 Learning Iterations

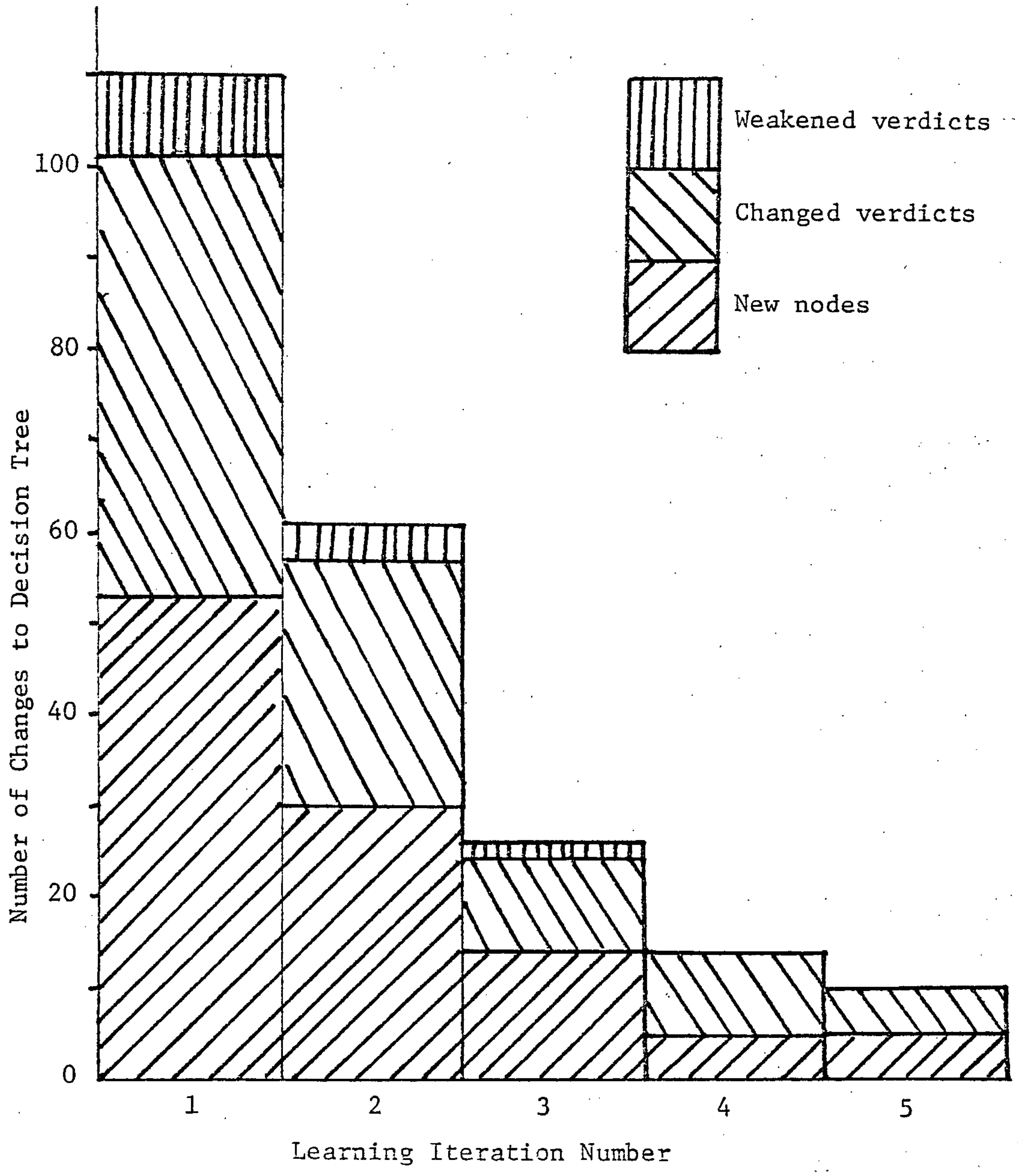
Figure 3.22

manually produced version gives a better rendition of the original line than the automatically generated result. It is necessary to note that at this stage all the measures indicate that the learning has stabilized.

The fact that the system is capable of fairly precise mimicry is encouraging because it suggests that the local viewing of lines that the system is forced to adopt is quite adequate. If the system can account for a person's generalization behavior over a representative set of lines then there should be enough information available locally to satisfactorily generalize any line. However, the problem remains of how to ensure that one has a truly representative set of lines.

The next step in evaluating the system's ability to mimic is to determine the rate at which it learns. There are several measures available to us and they all give roughly consistent results. For each iteration we can look at the total number of misclassifications of points, the difference between the number of points in the master copy and imitation (see Table 3.5), and the changes that the decision tree has undergone (see Table 3.4). A graph summarizing this last set of information is shown in Figure 3.23. From it we can see that the learning seems to be fairly rapid, although there is no independent standard for comparison.

Of course the real test of the system comes when we look at how well it generalizes outlines that it has never met before. To test this aspect of performance the generalization component



Learning Progress

Figure 3.23

Learning Statistics

		Number of Changes to Terminal Node Verdicts			Number of Tree Expansions
		Stronger	Weaker	Changed	
Iteration 1					
Line	A	71	0	6	15
	B	105	2	13	14
	C	30	1	4	2
	D	45	2	4	8
	E	22	0	6	1
	F	99	4	15	13
Total		372	9	48	53
Iteration 2					
Line	A	76	3	8	5
	B	121	0	8	5
	C	34	0	1	2
	D	51	0	2	6
	E	26	0	3	0
	F	113	1	5	12
Total		421	4	27	30
Iteration 3					
Line	A	88	1	1	2
	B	129	0	2	3
	C	35	0	1	1
	D	54	1	0	4
	E	26	0	3	0
	F	124	0	3	4
Total		456	2	10	14
Iteration 4					
Line	A	89	0	1	2
	B	133	0	1	0
	C	35	0	2	0
	D	57	0	1	1
	E	27	0	1	1
	F	127	0	3	1
Total		468	0	9	5
Iteration 5					
Line	A	89	0	2	1
	B	133	0	0	1
	C	36	0	0	1
	D	58	0	1	0
	E	27	0	1	1
	F	129	0	1	1
Total		472	0	5	5

Table 3.4

Learning Results

Number of Points in Outline

Outline	Original Outline	Learning Master	After Learning Iteration 1	2	3	4	5
A	92	23	33	29	29	29	23
B	134	41	48	40	41	41	41
C	37	19	19	19	19	19	19
D	59	16	15	15	15	15	15
E	29	10	11	11	11	11	10
F	131	33	36	34	33	33	33

Table 3.5

Generalization Results

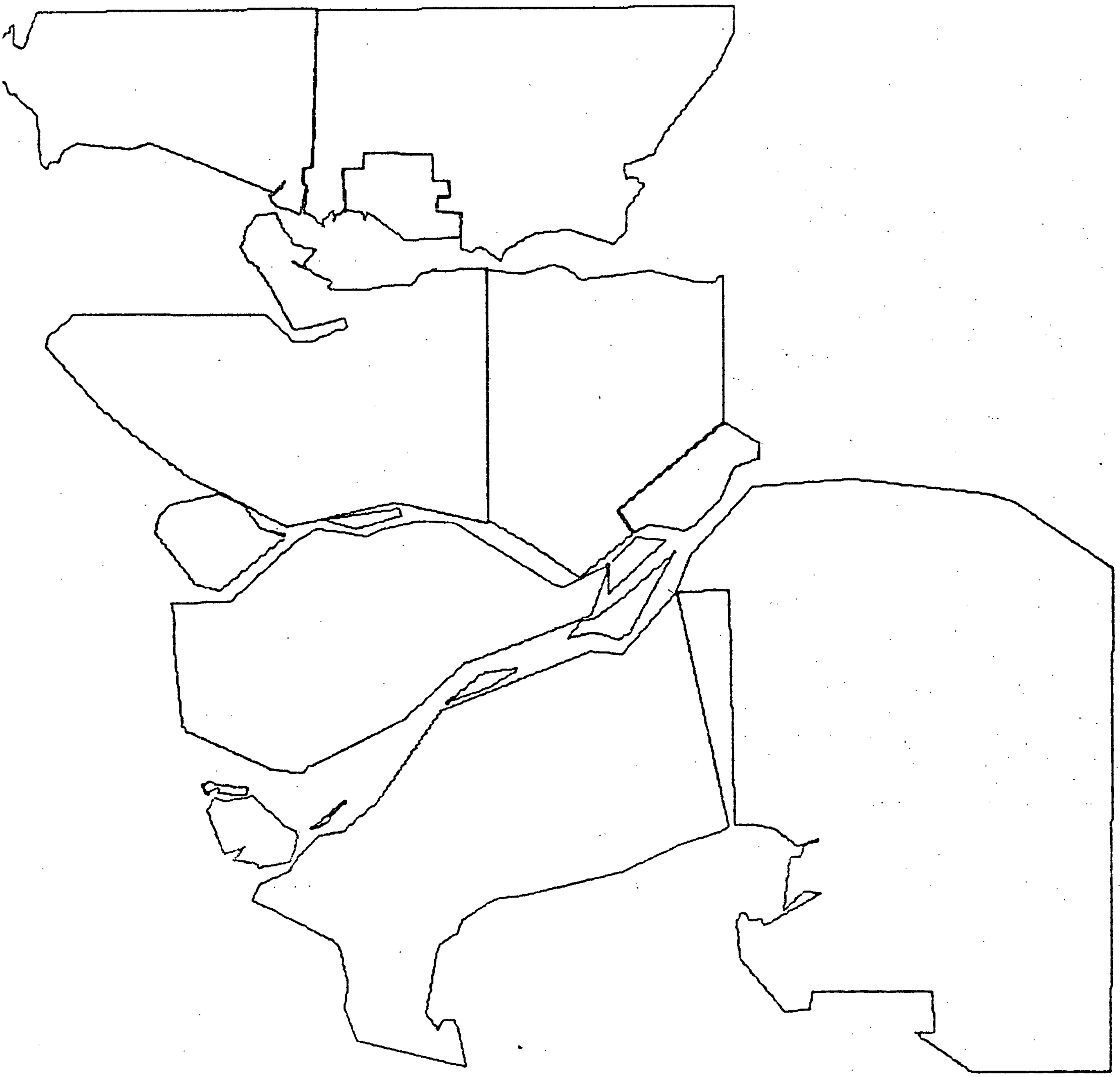
Number of Points In Outline

Outline	Original	1/4 Original	Learning Iteration 1	2	3	4	5	Douglas Method 0.05"	0.04"
A	92	23	33	29	29	29	23	16	26
B	134	35	48	40	41	41	41	30	35
C	37	9	19	19	19	19	19	14	16
D	59	15	15	15	15	15	15	13	15
E	29	7	11	11	11	11	10	10	10
F	131	33	36	34	33	33	33	32	33
G	308	77	44	39	40	40	39	38	48
H	51	13	10	12	11	11	11	10	10
I	23	6	5	5	8	8	8	5	5
J	123	31	18	15	17	15	16	19	24
K	38	10	9	9	9	10	9	5	9
L	55	14	13	15	15	15	14	10	11
M	27	7	5	5	5	5	4	6	6
N	182	46	26	28	25	24	24	24	32
O	29	7	6	6	5	5	5	4	4
P	51	13	8	9	8	8	8	5	8
Q	26	7	5	5	5	5	5	5	5

Table 3.6

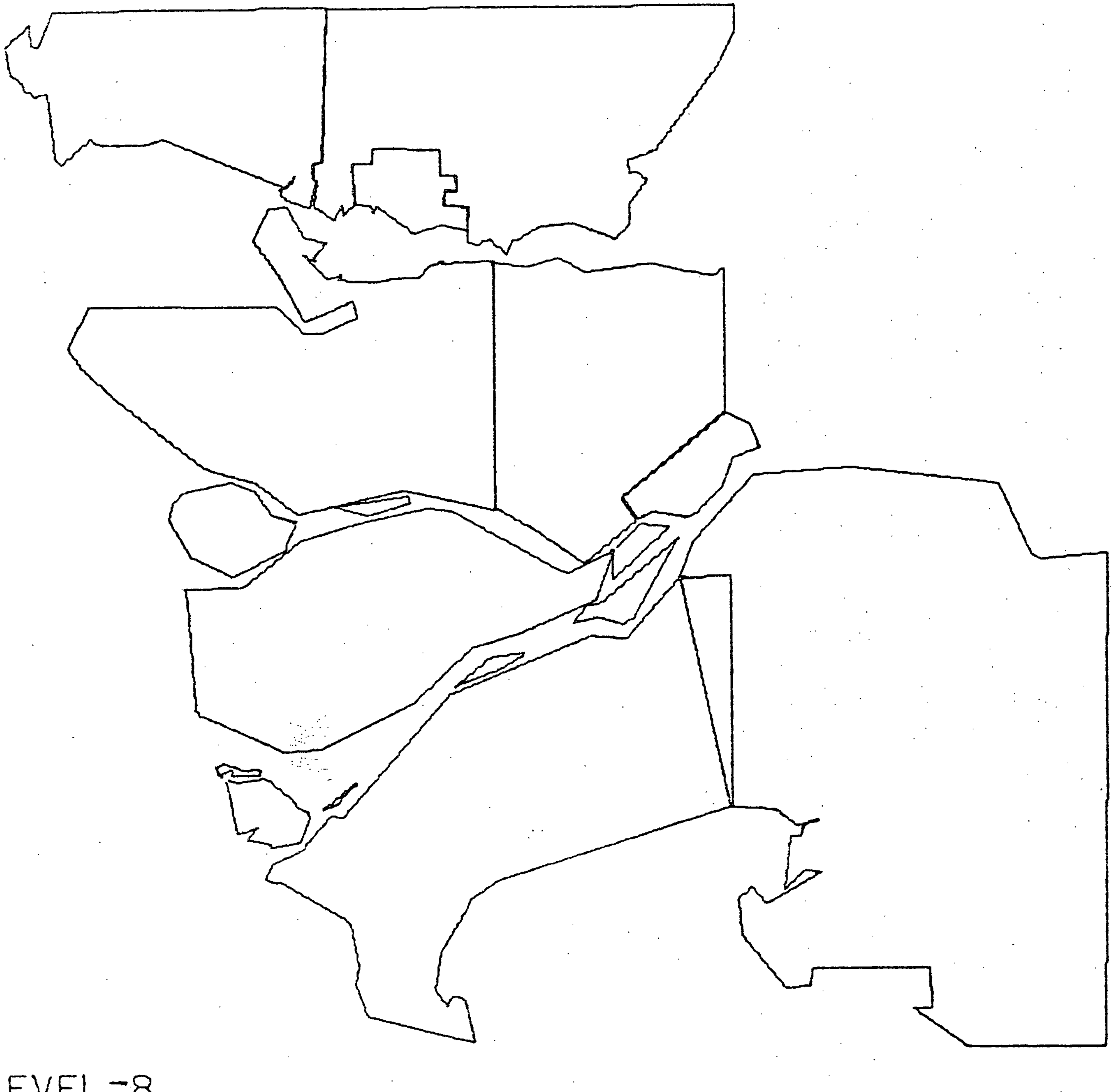
of the system was presented after each of the five learning iterations, with all the outlines shown in Figure 3.15. The resulting outlines are found in Figures 3.24 through 3.28, while Table 3.6 gives the number of points in each outline originally and the number of points actually remaining after each generalization.

We can see that the generalization has been successful in that the general shapes of the outlines have been well maintained and that the reduction in the number of points is of the desired degree. However, there are some glaring flaws (especially in outline N) that immediately catch the eye. There are also a number of other serious distortions, some of which diminish with increased learning although there is generally little change throughout the sequence. This is somewhat discouraging since, in view of the fact that the learning was virtually stabilized, we cannot look to additional learning to remedy the situation. The actions that are left open are: alter the learning parameters (ie. the angle thresholds, minimum/maximum lengths and tree depth), increase the learning set making it more representative, re-reduce the learning set and give a more general rendition, or manually reduce the offending portions while learning at the same time. The last alternative is a particular case of the preceding one and both are reasonable steps to take if there is a large, relatively homogeneous set of outlines that need to be generalized. The second alternative is probably not likely to make much difference especially since it is not clear where the current manual reduction could be improved significantly. If we restrict

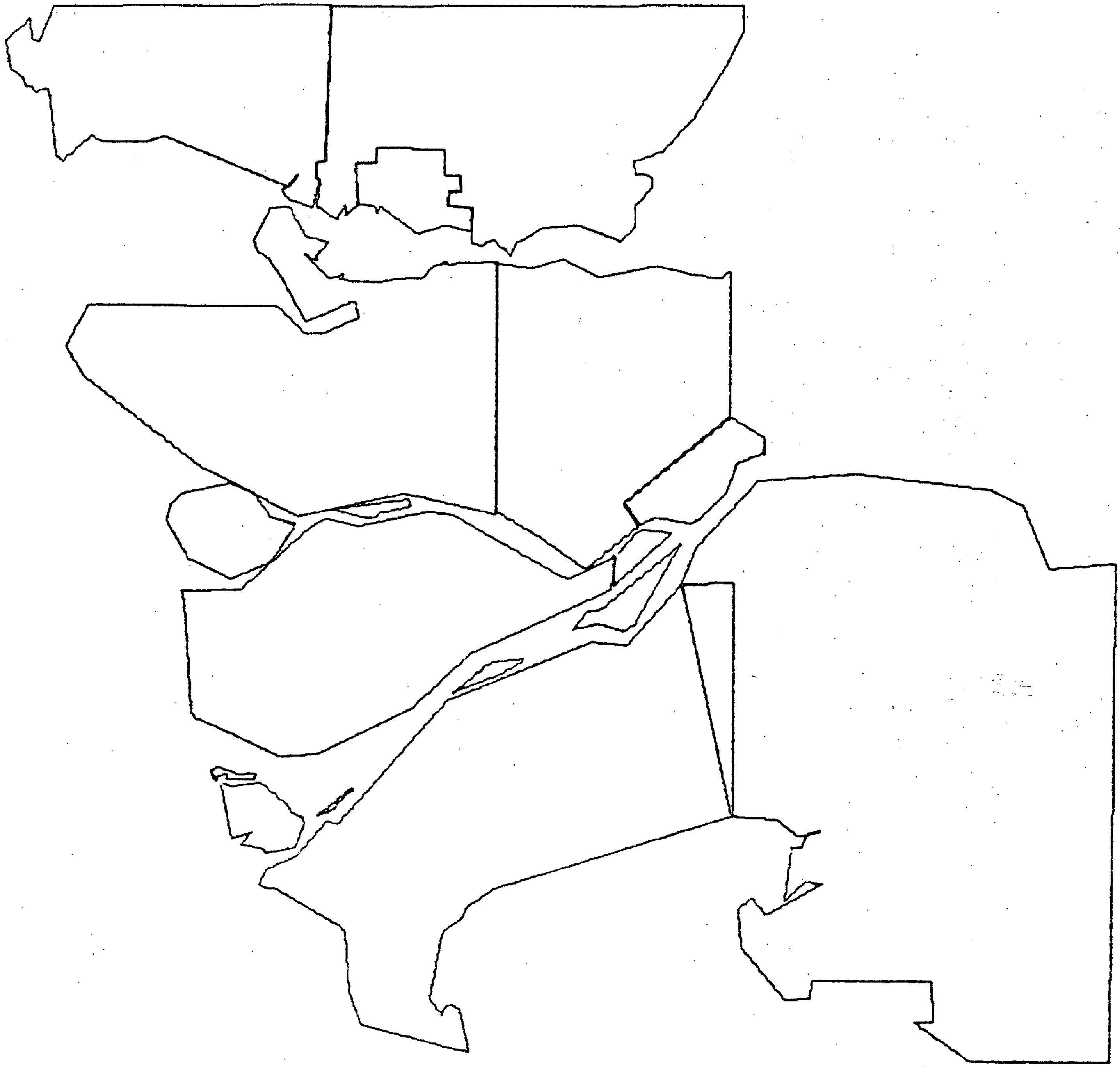


VEL=8 Reduced Outlines after 1 Learning Iteration

Figure 3.24

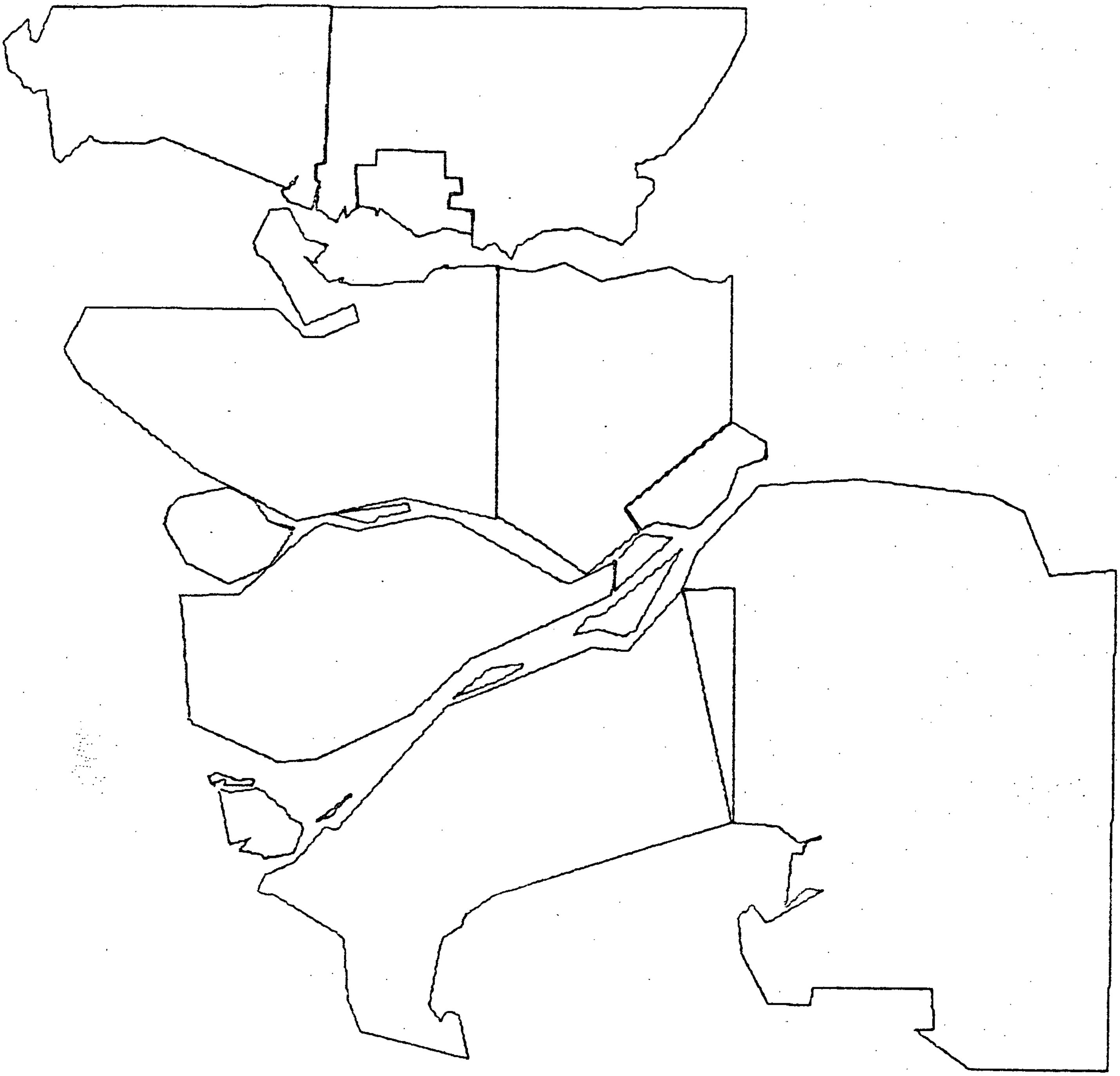


Reduced Outlines after 2 Learning Iterations Figure 3.25



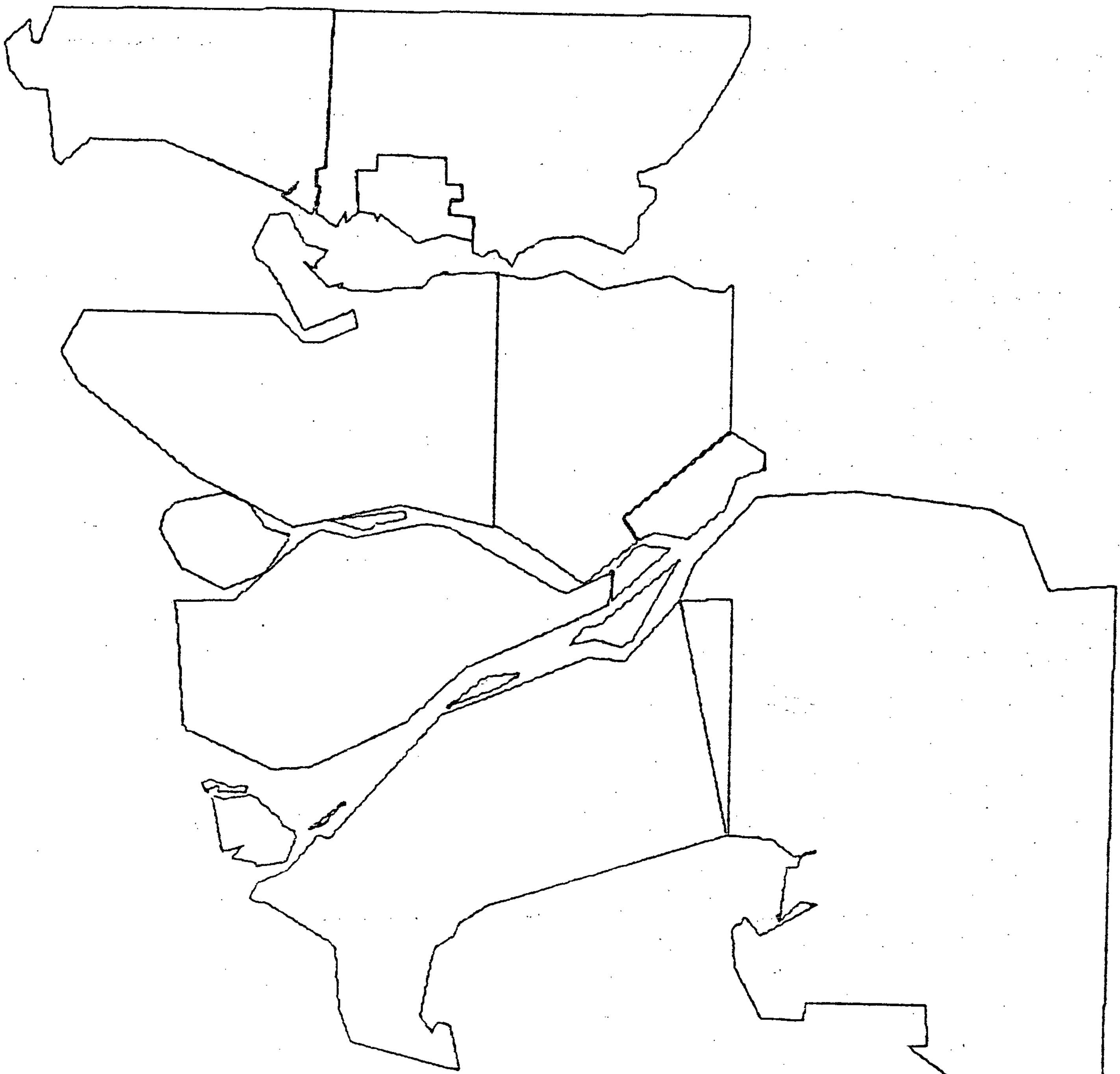
LEVEL=8

Reduced Outlines after 3 Learning Iterations Figure 3.26



EVEL=8

Reduced Outlines after 4 Learning Iterations Figure 3.27



LEVEL=8

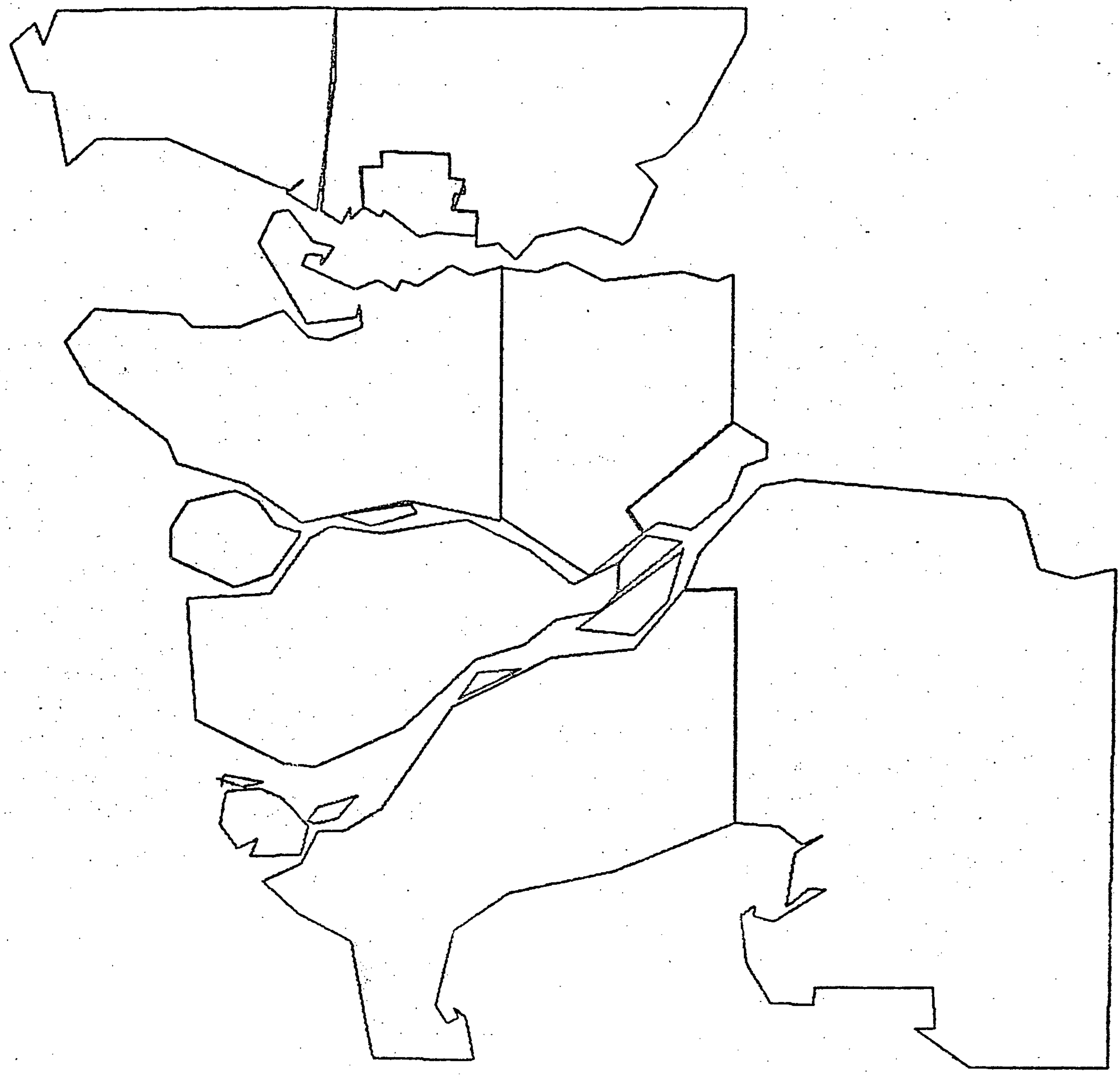
Reduced Outlines after 5 Learning Iterations

Figure 3.28

ourselves to the original given set of outlines then an obvious step to take is to alter some of the learning parameters.

Another way to evaluate the generalization performance of this learning method is to compare it with other methods of generalization. The method chosen for comparison is the one proposed by Douglas (1972) and described in Chapter 1. The routines required for this method are straight-forward and were easily incorporated into the system as just another means for selecting subsets of points whose values are to undergo a specified transformation. This method was chosen for comparison not only for convenience, but also because it is a good method. It has the advantage that effectively global considerations are involved in the selection of points, and it also guarantees that no point in the original line is further than a specified distance from the generalized version. Thus some of the glaring mistakes encountered with the previous method will be avoided. Using this method with deviation thresholds of 0.05" and 0.04" the complete set of outlines was generalized. The resulting outlines are shown in Figures 3.29 and 3.30 respectively while the last two columns in Table 3.6 record the number of points in each of the outlines. There is little doubt that these are, on the whole, better generalizations than the earlier ones. Also, the computation required to do them is considerably less (approximately 2.0 seconds of CPU time as opposed to 7.5 seconds). It is therefore clear from this comparison that the learning method does have some deficiencies.

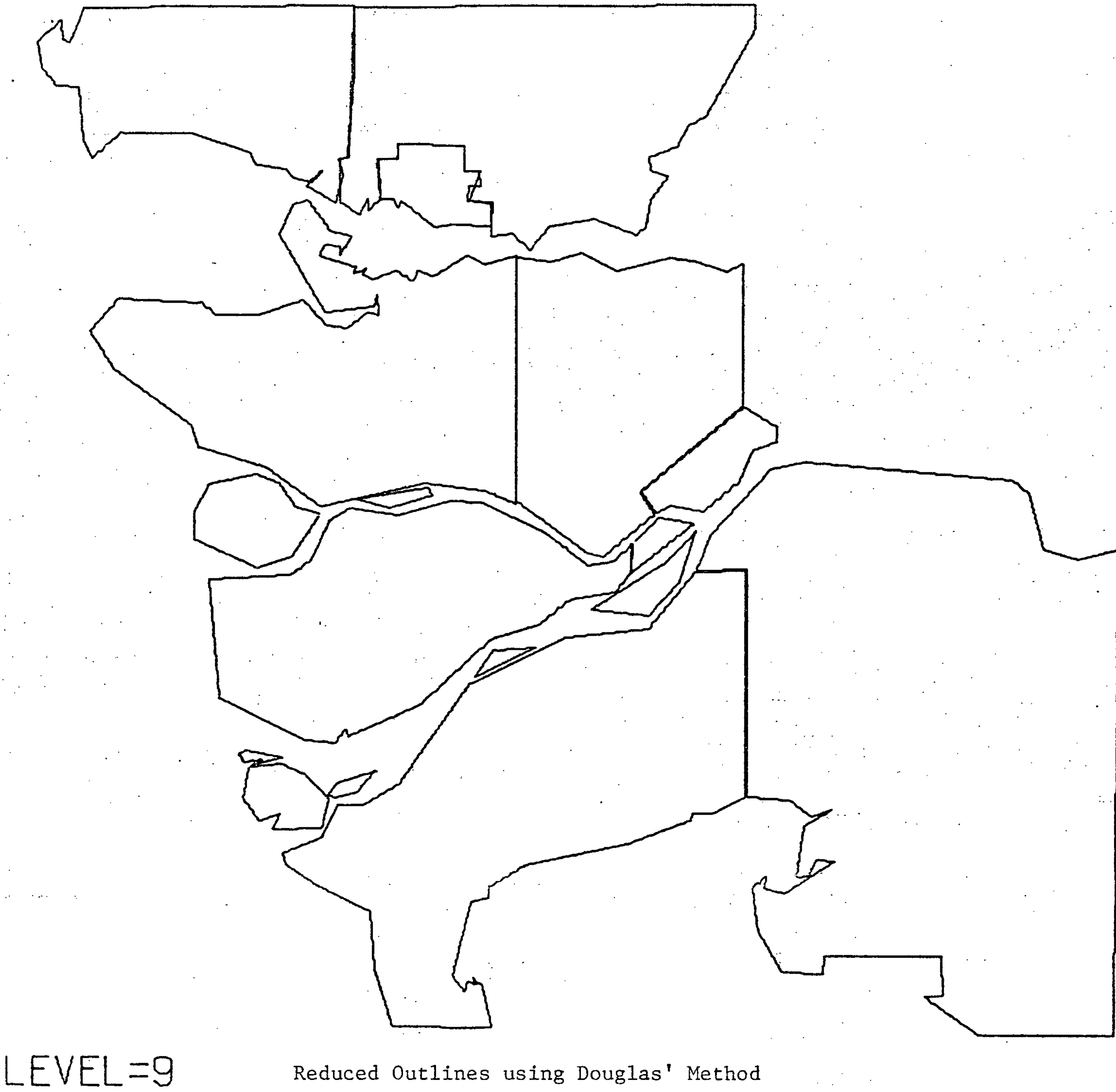
The learning method for generalization does, however, enjoy



LEVEL=9

Reduced Outlines using Douglas' Method
Tolerance=0.05"

Figure 3.29



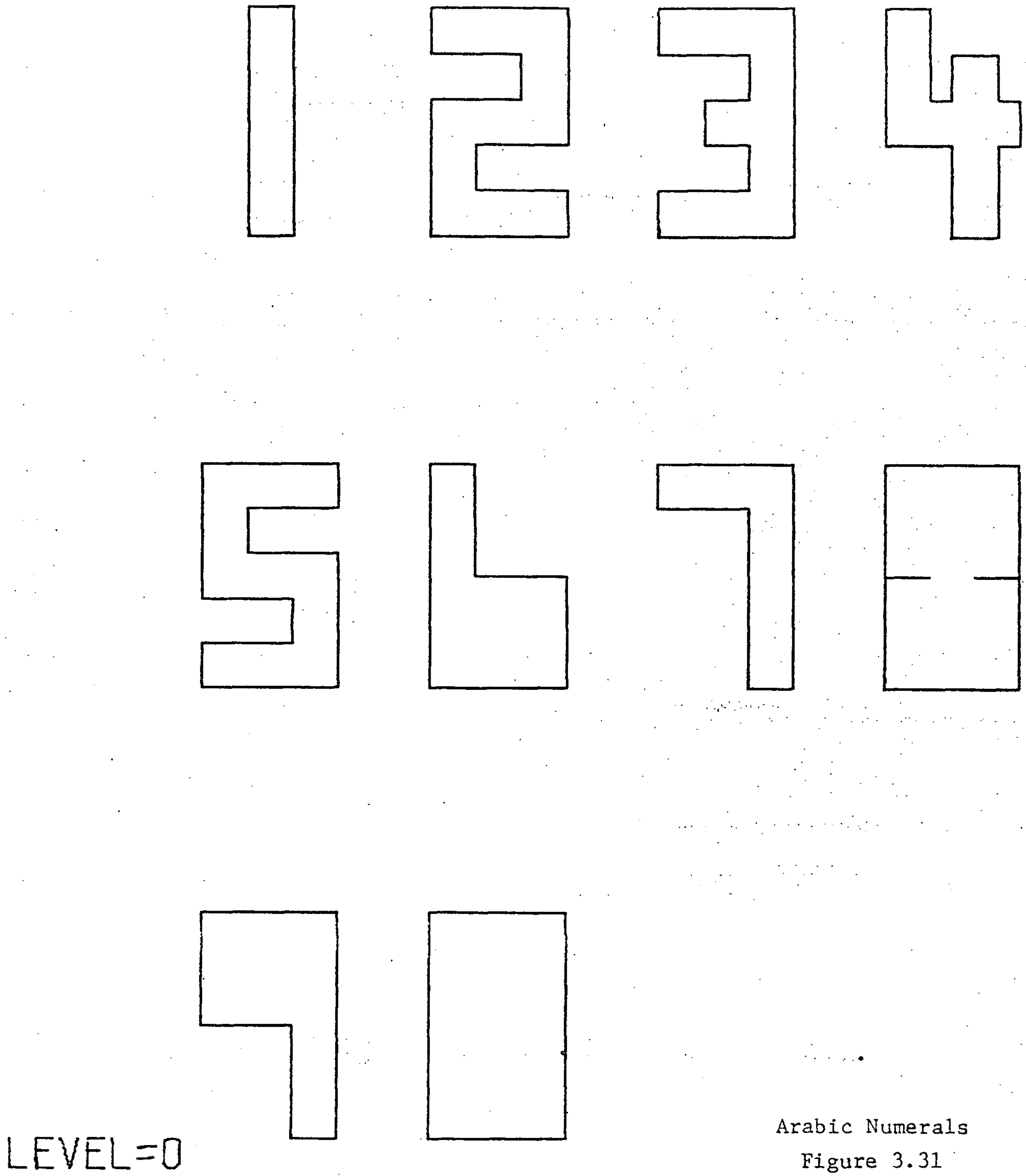
Reduced Outlines using Douglas' Method
Tolerance = 0.04"

Figure 3.30

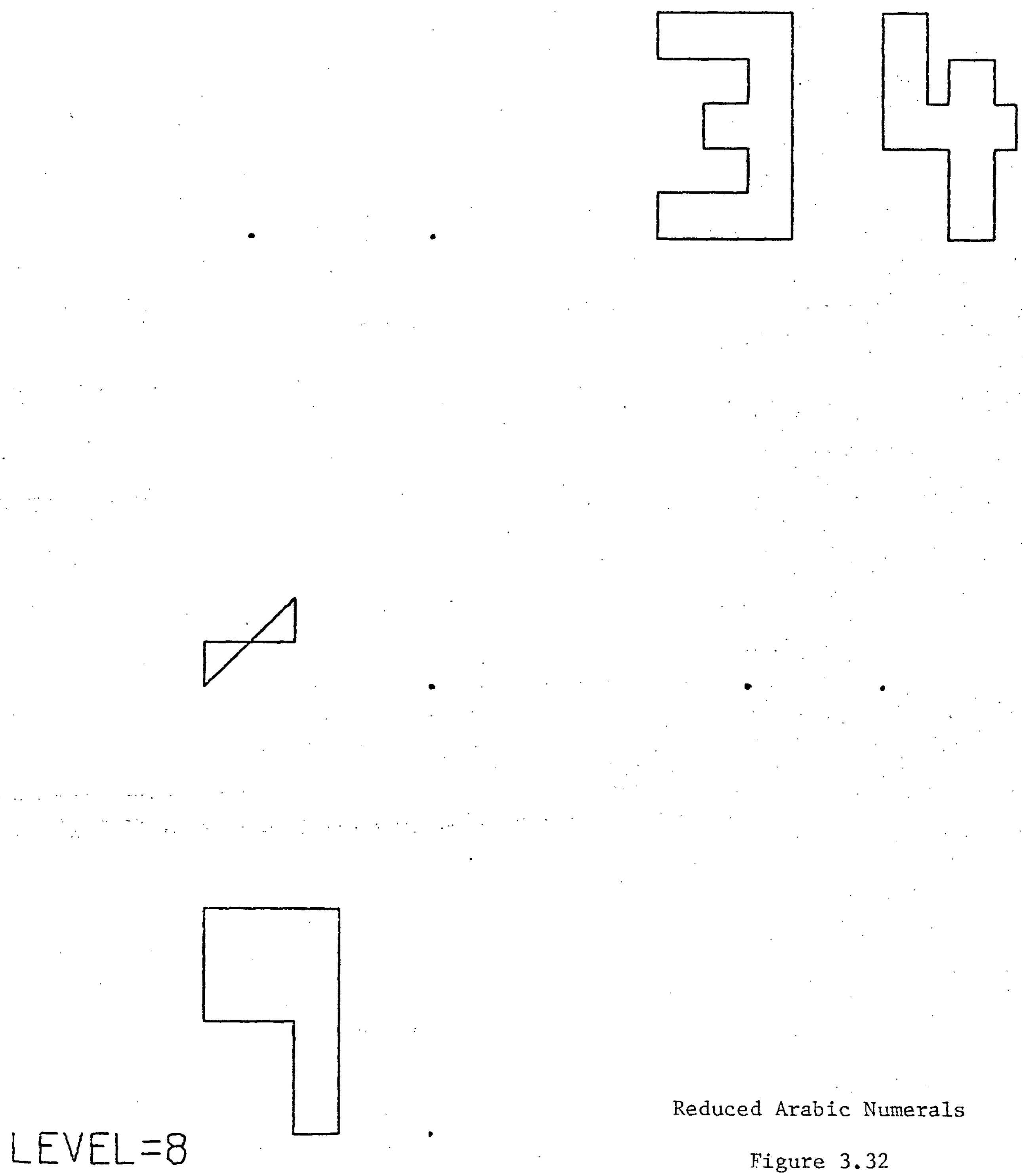
an advantage when it comes to special types of lines that are being generalized for special purposes. This is because the generalization is done essentially by recognizing patterns within the line. An extreme example illustrates this point.

Stylized versions of the arabic numerals 0,1,2,...,9 were digitized (see Figure 3.31) and then given to the learning component (with suitable adjustment of parameters) with the instruction that the numerals 3,4 and 9 were to be retained while the remaining numerals were to be reduced to a point. After each of the digits had been presented several times they were all reduced according to what had been learned. The result appears in Figure 3.32.

A more practical example involves the generalization of a coastline containing docks. From the outline depicted in Figure 3.33 one person (a ship's pilot, say) may want a generalization that keeps the docks (Figure 3.34) while another person may want the docks removed to show the original landform (Figure 3.35). These two versions were used to teach the system (on separate occasions) and after four iterations the learning had stabilized to the point that generalizations of the original line were virtually identical to their respective models (see Figures 3.36 and 3.37). The same learning was employed to generalize a portion of the Vancouver waterfront (see Figure 3.38, which is an enlargement of a section of the upper edge of outline G in Figure 3.15) giving the two versions shown in Figures 3.39 and 3.40. The key in this case was the adjustment of the angle thresholds to recognize right angles.

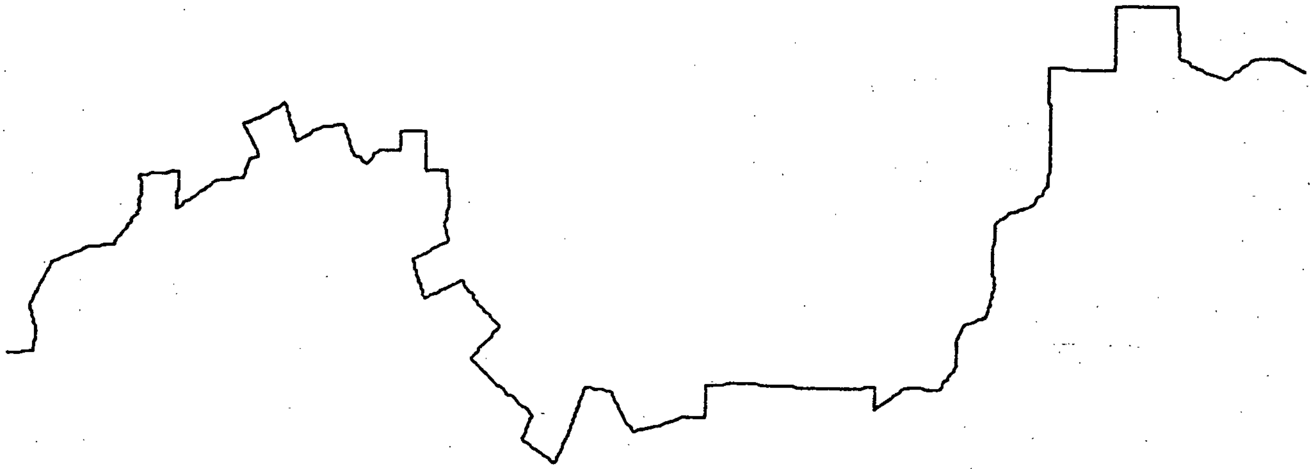


Arabic Numerals
Figure 3.31



Reduced Arabic Numerals

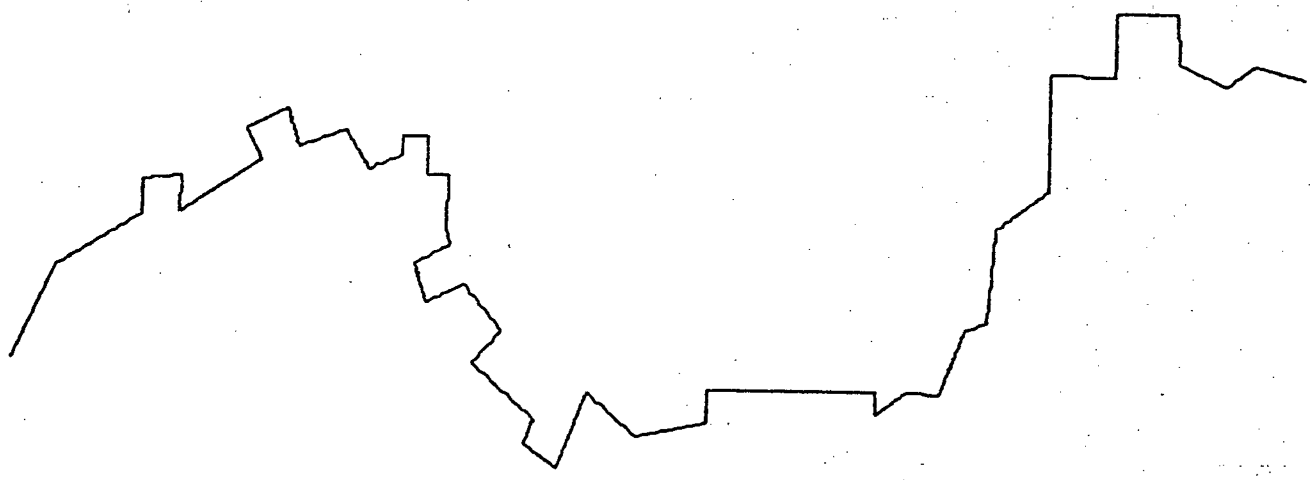
Figure 3.32



A Coastline with Docks

LEVEL=0   DOCKS

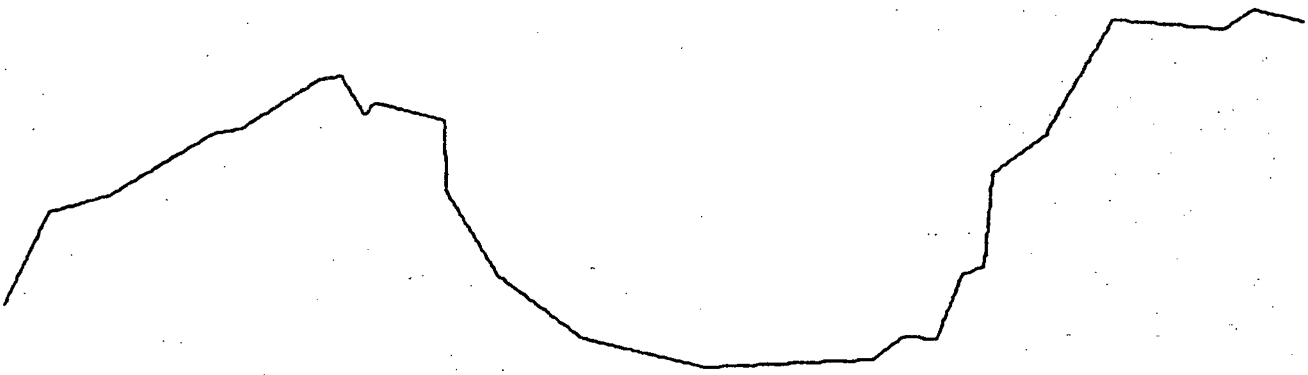
Figure 3.33



Reduced Coastline for Learning (docks kept)

Figure 3.34

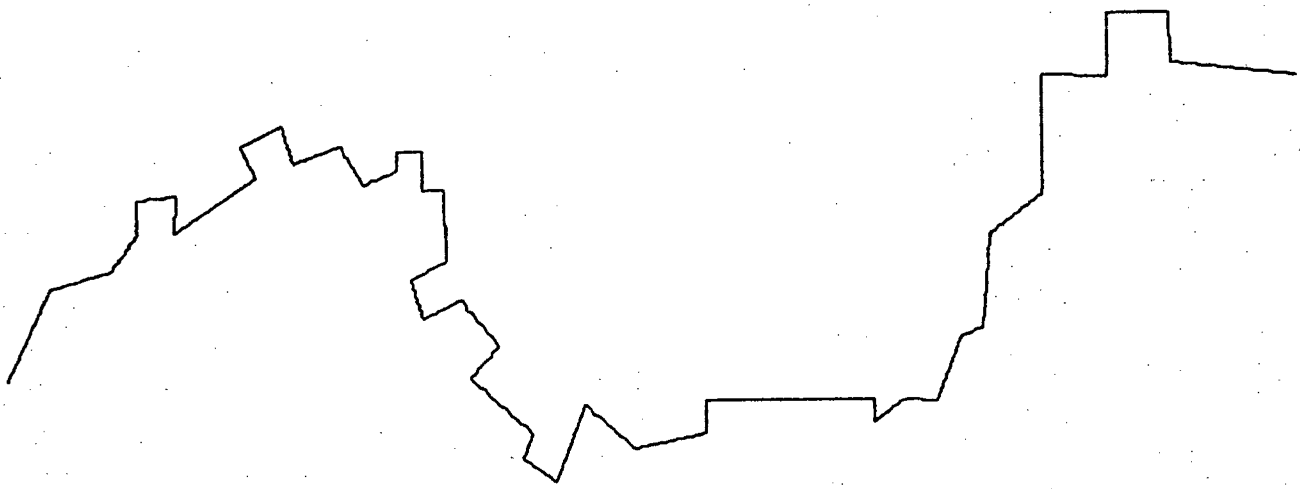
LEVEL=8 DOCKS_KEPT



Reduced Coastline for Learning (docks removed)

Figure 3.35

-EVEL=8 DOCKS_GONE



Reduced Coastline after Learning (docks kept)

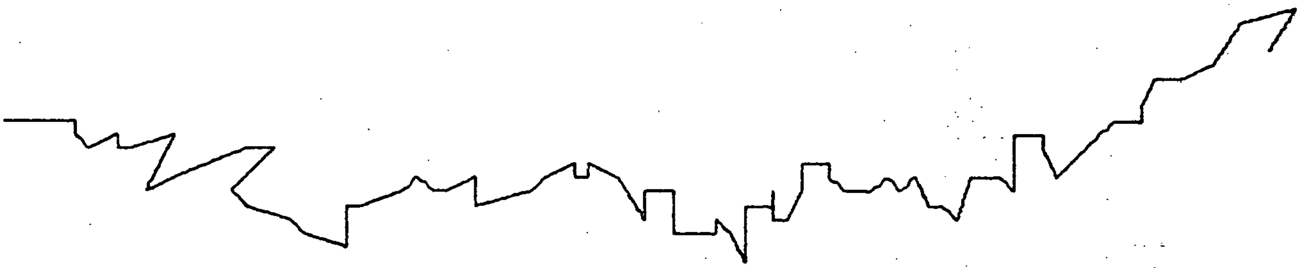
LEVEL=8

Figure 3.36

Reduced Coastline after Learning (docks removed)

LEVEL=8

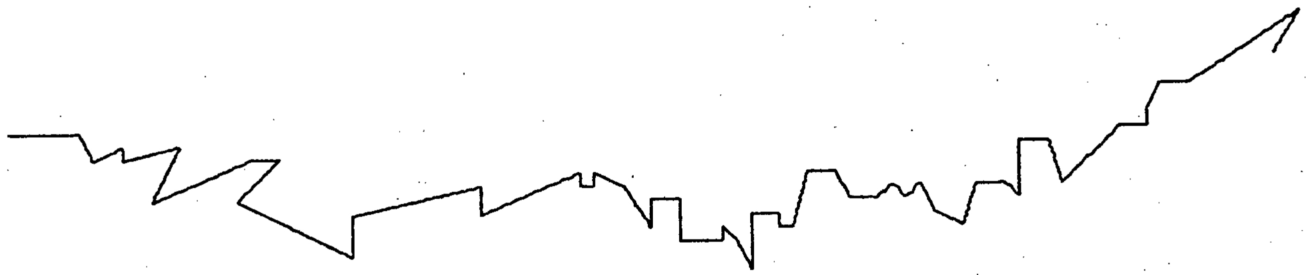
Figure 3.37



Original Vancouver Waterfront

LEVEL=0   VANCOUVER_DOCKS

Figure 3.38



Reduced Vancouver Waterfront (docks kept)

LEVEL=8

Figure 3.39

Reduced Vancouver Waterfront (docks removed)

LEVEL=8

Figure 3.40

3.4 Other Questions

Apart from the questions that have been discussed so far, there are many other questions that can be asked about the system and answered through experiments. One largely untouched problem is that of measuring the homogeneity of a set of lines and selecting representative subsets. I have made some initial attempts in this area by looking at the distribution of lengths and angles within lines and also by considering the quantized lengths and angles along lines as Markov chains. Nothing very promising has yet emerged, however. Without some understanding of the similarities and differences between lines the selection of a representative learning set and the assignment of parameter values is very much a hit or miss affair. A related problem that deserves more attention is how the rate of learning and the transferability of this learning is affected by particular assignments of parameter values.

Another interesting question that might be asked concerns the "psychology" of the learning process. Since the learning mechanism is EPAM-like (Feigenbaum (1963)) we should expect to see evidence of such psychological features of learning as oscillation, forgetting, interference and so on.

Chapter IV EVALUATION

4.0 Introduction

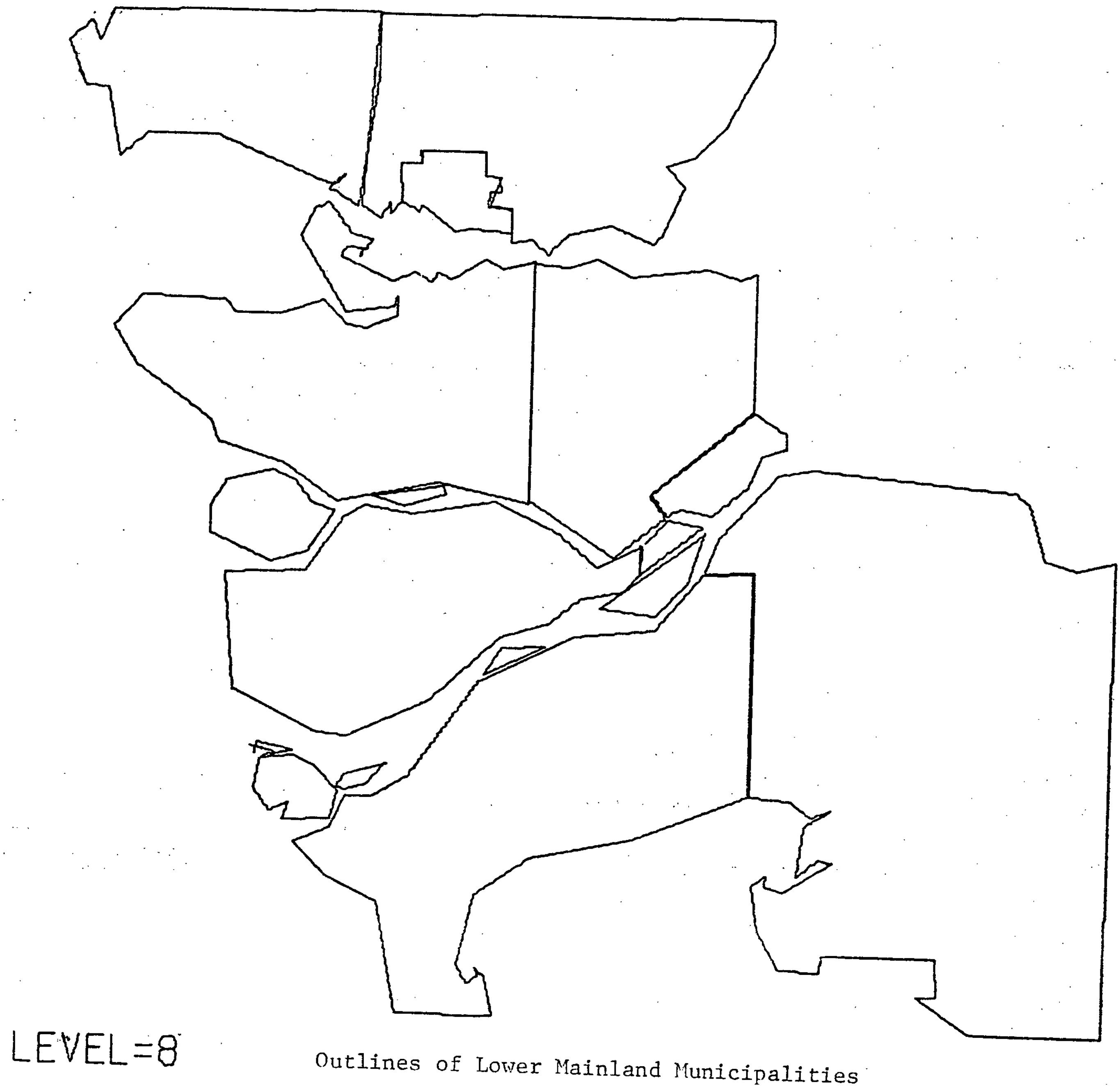
There are three main aspects of this work that may be of some significance. These are :

- an encoding scheme that is suitable for the representation of hierarchically generalized lines.
- an interactive graphics system that provides the means to manipulate lines represented in this way and generalize them either manually or automatically.
- a system that performs generalization by learning to recognize patterns in lines.

These are summarized critically in this chapter as well as related topics that I think deserve future investigation. Some of my ethical concerns are touched on and the chapter ends with a summary and conclusions.

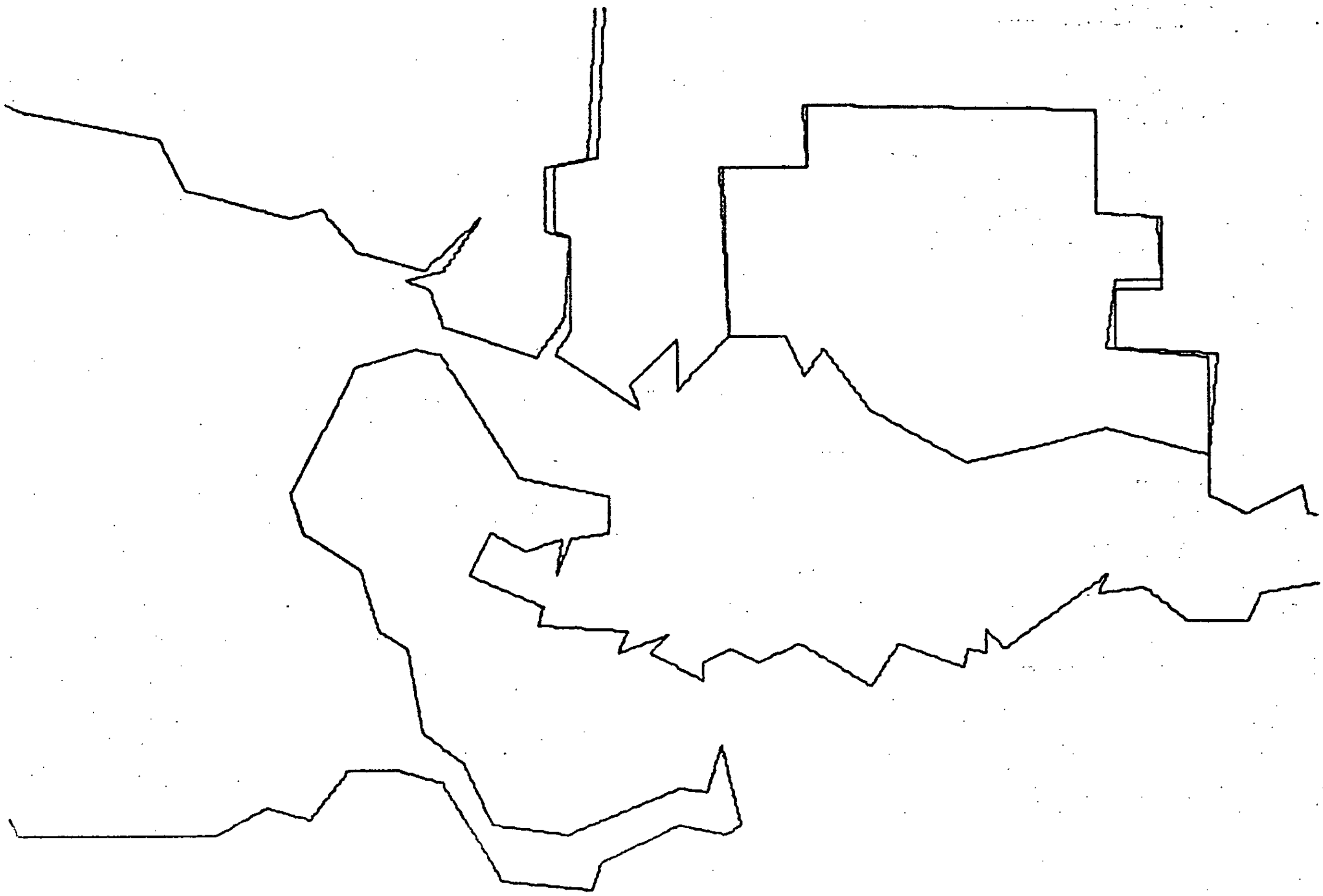
4.1 Critical Summary

I believe the idea of a line with levels attached to the points to be a potentially useful notion. This is because several versions of a line, each generalized to a different degree, can be represented within a single entity. Apart from being compact, it also allows a conceptually elegant way of referring to a whole family of related lines. For example, with only a single level attached to each point, representing the importance of the point in conveying the message of the line, it is possible to conveniently specify the enlargement process observed in the sequence of Figures 4.1 through 4.3. This is



Outlines of Lower Mainland Municipalities
with reduced Detail

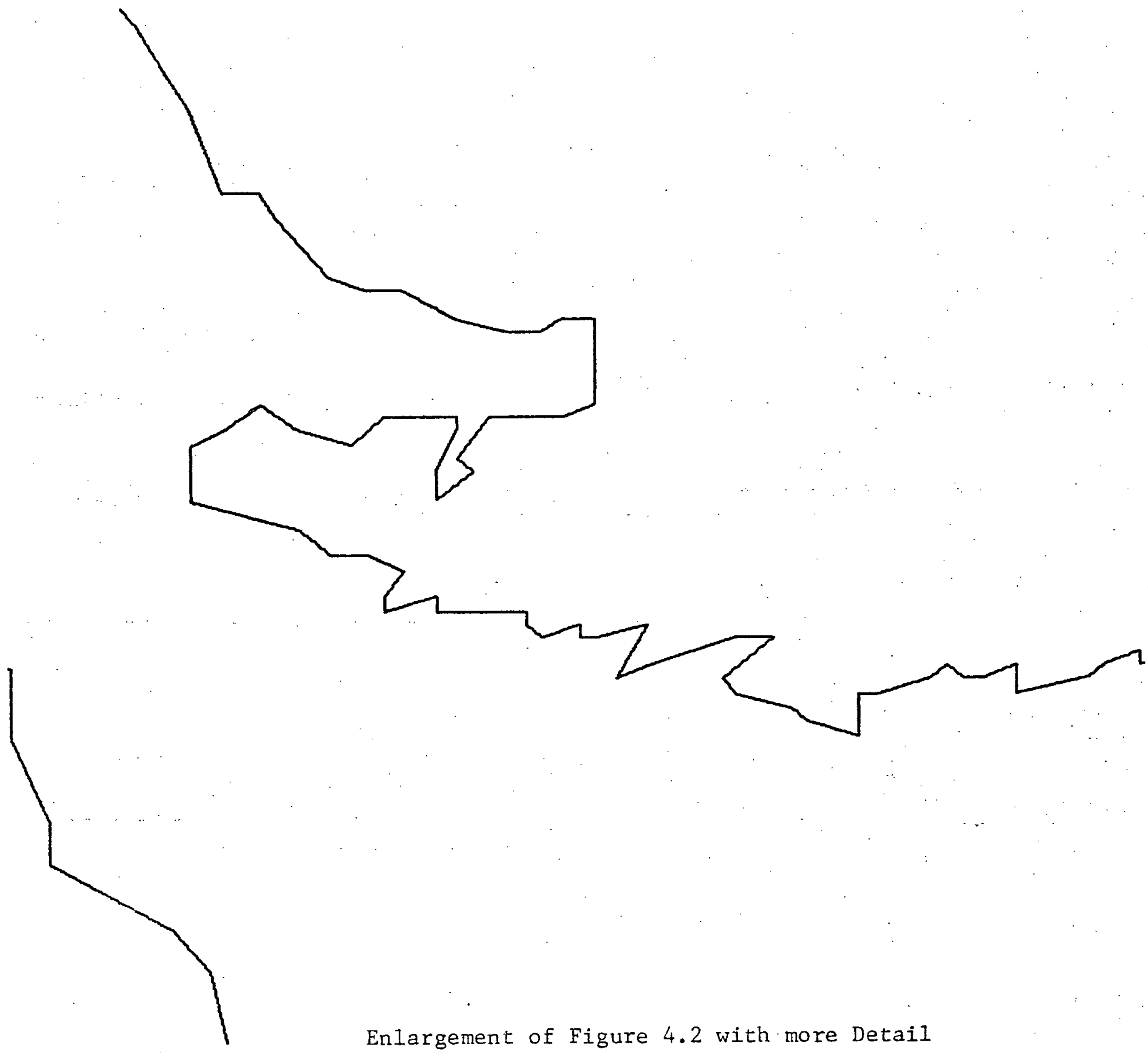
Figure 4.1



LEVEL = 4

Enlargement of Figure 4.1
with more Detail

Figure 4.2



Enlargement of Figure 4.2 with more Detail
Figure 4.3

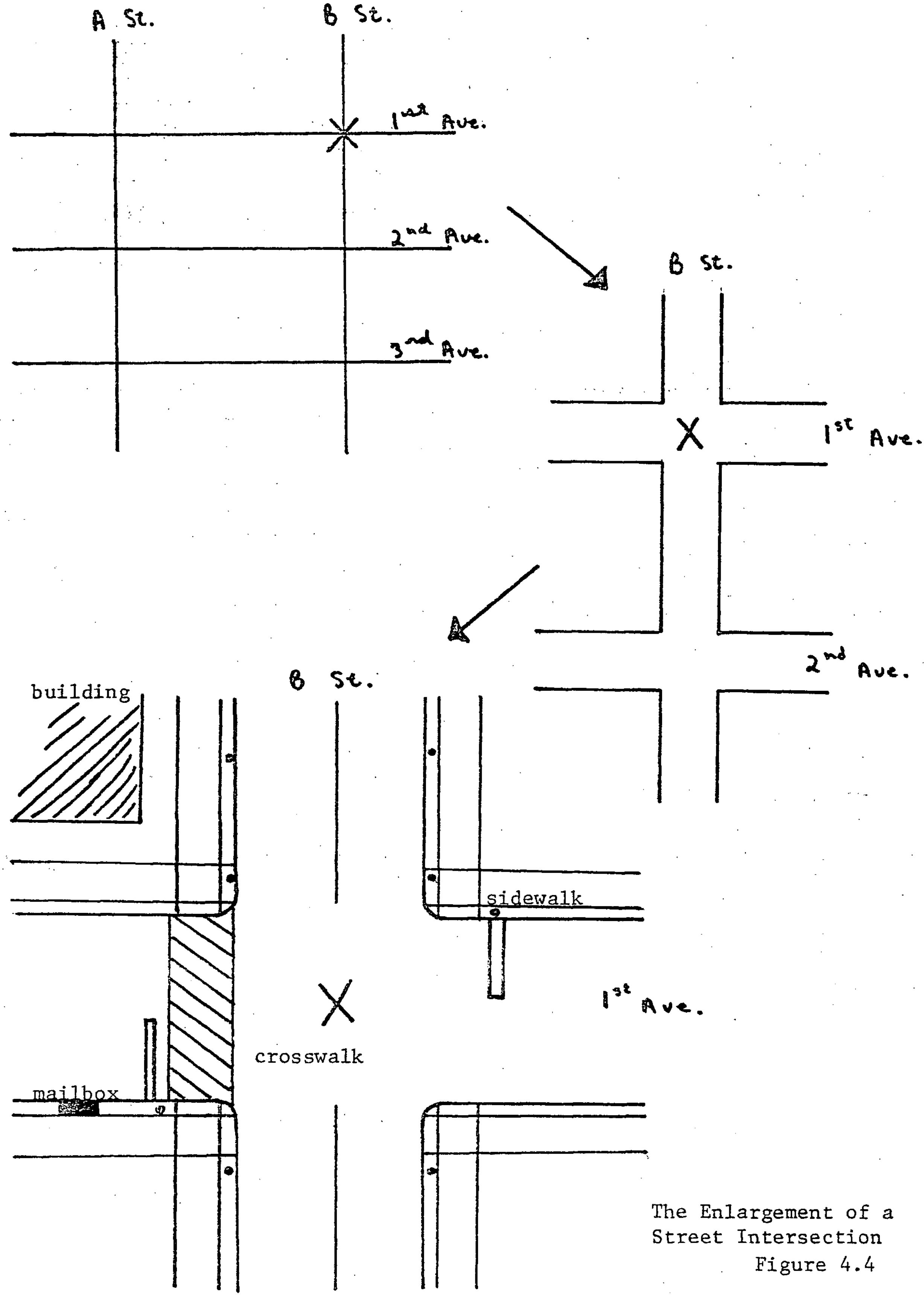
LEVEL = 0

typical of many interactive graphics applications and in this case can be handled by simply specifying the lines to be displayed, the window parameters and the relationship between scale and level of display. For the enlargement observed in Figure 4.4 two levels attached to each point are adequate for the representation of the streets that are involved.

The compactness of encoding arises from the fact that the individual points that are shared by several versions of the same line need be stored only once. The actual saving of course depends on factors such as the character of the lines, the number of levels of generalization, and the particular internal machine representation of the points.

This line representation technique does have some drawbacks. It means that every time a line is used each point must be inspected to determine if it is to be included. It might be cheaper and more convenient, if a good generalization scheme is available, to generalize the line to the desired degree every time. This way the specific needs do not have to be anticipated and all the generalization done beforehand. The relative tradeoffs here are just another instance of the "display structures" versus "display procedures" (Newman (1971)) argument.

An interactive graphics system suitable for line generalization is another aspect of this work that is of some interest. It provides a means of bringing in lines from external storage, associating values with the points along the lines and displaying these lines at various levels. The manipulation of



The Enlargement of a Street Intersection
Figure 4.4

values can be done manually using a rapidly updated image on the screen of a graphics terminal or else by a variety of automatic methods with the results available within seconds for correction if necessary. Such capabilities are desirable features in any generalization and geographic editting scheme. However, since the system was designed mainly as a vehicle for developing and testing the learning part, it lacks many of the components necessary for it to be a practical and useful system. It has, I believe, a sound basic structure and conceptual framework but has a number of inconsistencies and is often clumsy to use. I could tolerate these since I knew it intimately and had restricted goals, but they would prove to be stumbling blocks for the average user. For example, if one makes a mistake in assigning a value it can often be difficult to correct since points have to be processed sequentially within lines. Making provision for the use of the light pen or cross hairs would not be hard to do and would overcome much of this current difficulty. At present there is no way to change the coordinates of a point although this would be necessary in many real applications. The fact that the system is not presently very useful is a shortcoming of this work since through its use not only would more people benefit, but also a much better understanding of generalization could be obtained. However, the system does represent an initial step and does have potential for development into a flexible and useful tool for map generalization.

The major emphasis in this work has been the development of a technique for the generalization of map outlines which operates by recognizing patterns that have been taught to it by people. The aim of this generalization is to produce maps suitable for use in interactive graphic situations, which means they must require much less data storage but at the same time retain their recognizability. Since people have widely differing views about the similarity of maps and since these generalized maps may be used for a variety of purposes there is a need to tailor the generalization to a particular individual's wants and tastes. The system described previously was constructed to satisfy this need by learning to mimic the user's behaviour at generalizing lines manually. Experiments with people suggest that a program should be able to generalize by recognizing fairly local patterns within lines and in fact the performance of the system bears this out. It is able to learn to mimic almost precisely the person, with relatively few learning trials. It is also able to generalize new lines but is not as reliable nor as proficient as existing analytic methods with general types of lines. This appears to be due to gaps or holes in the learning that are a result of not having a representative enough set of lines to learn from and not being able to generalize its learning sufficiently from the cases presented to it. While I have little doubt that the system could satisfactorily generalize any reasonably homogeneous set of lines, the trouble taken to do this would not be worthwhile. Like a delicate instrument it would require a great deal of tinkering before working properly. This rather defeats the purpose of being easily suited to an individual's requirements. One possible way to overcome this difficulty is to have a basic repertoire of learning lines that contain many of the commonly occurring features. Then the system is taught with extra lines that are specially chosen to "tune" it for a particular application. Another disadvantage of the current system is that it is significantly more expensive to use than other methods, both in terms of processing and storage requirements. Perhaps with decreasing hardware costs this factor will diminish in importance as an obstacle to the application of this technique for generalization.

4.2 Future work

Since the technique for generalization in this thesis is new there are several areas that require further investigation before the technique becomes a useful one. One such area concerns the way in which people perceive and recognize maps, especially when their data content has been reduced drastically. Related to this, there must be more study of outline generalization. What are the various criteria that dermine how well a particular generalization approximates the original line? It is this question that has been at the heart of my investigation.

Another area of interest is in trying alternative techniques of pattern recognition for generalization. Currently each point is considered in turn and its immediate environment inspected to determine whether the point should be removed or not. This approach was taken simply for convenience but it would perhaps be more logical to recognize certain features in lines (such as bays, inlets, docks, rocky shorelines, peninsulas, etc.) and to generalize the whole feature at once. Either the feature could be removed completely or replaced by a few suggestive lines or just simplified slightly. Which of the possible actions is taken could be chosen according to various probabilities that are dependent on various parameters of the feature so that it would be possible to maintain the character of large sections of outline (e.g., a deeply indented coastline such as that of northern British Columbia) . In order to obtain a representative learning set for any pattern recognition scheme more work must be done to categorize different types of lines.

On a more concrete level there is much to be done on the currently implemented technique before its potential for improvement is exhausted. Apart from seeing in more detail how the learning and generalization performance depend on parameter values and on the types of lines used in the learning, modification to the heuristics employed in the learning component might also lead to significantly improved performance. I suspect that changes to the way in which the decision tree is expanded could be particularly fruitful.

4.3 Ethical Concerns

Approximately half of the papers that I made use of in my work were supported by the U.S.A. Military. This disturbs me because it makes me wonder whether the uses to which my work is put, if any, will be beneficial. This fact alone is not sufficient to convince me that I should do other work but it is an aspect that I must consider. What disturbs me more is that other people are apparently not so concerned about the potential uses of their work. Very seldom are the dangers of the potential applications of work discussed. Nowhere have I found any mention of the dubious use of pattern recognition in the analysis of air photos for military espionage for instance, an often cited application. There should be a much more open discussion of how research might be used and what the consequences of this use might be. I believe this to be essential if research is going to be of net benefit to mankind.

4.4 Conclusions

It has been a principal aim of mine in doing this work to explore techniques for making the computer a useful tool to serve people. The problem of generalizing map outlines for use in interactive graphics situations is one area which requires of a computer system a certain amount of tailoring to an individual's needs. I believe I have shown that the application of interactive graphics and pattern recognition to this problem can be fruitful in providing flexible enough systems.

BIBLIOGRAPHY

Attneave, Fred (1954), "Informational Aspects of Visual Perception", _Psychological Review_, Volume 61, pp183-193.

Blum, H. (1964), "A Transformation for Extracting New Descriptors of Shape", _Symposium on Models for the Perception of Speech and Visual Form_, Wather-Dunn Weiant (eds.), pp362-380.

Deeker, Gordon (1970), "Interactive Graphics and a Planning Problem", M.Sc. Thesis, Department of Computer Science, University of Alberta (Edmonton), (Fall 1970).

Deeker, G. & Penny, J.P. (1972), "On Interactive Map Storage and Retrieval", _INFOR_, Volume 10, #1, pp62-74.

Douglas, David (1972), Personal communication - not yet published.

Feder, Jerome (1968), "Languages of Encoded Line Patterns", _Information and Control_, Volume 13, pp230-244.

Feigenbaum, Edward (1963), "The Simulation of Verbal Learning Behavior", Computers and Thought, Feigenbaum and Feldman (eds.), pp297-309.

Freeman, Herbert (1961), "On the Encoding of Arbitrary Geometric Configurations", Transactions of the IRE on Electronic Computers, EC-10, pp260-268.

Freeman, Herbert (1971), "Techniques for the Digital Computer Analysis of Chain-Encoded Arbitrary Plane Curves", Proceedings of the National Electronics Conference, Volume 17, pp421-432.

Hershey, T. (1963), "A Plotting of Maps on a CRT Printer", NWL Report, #1844.

Jarvis, C.L. (1971), "A Method For Fitting Polygons to Figure Boundary Data", The Australian Computer Journal, Volume 3, #2, pp50-54.

Keates, John S. (1972), "Symbols and Meaning in Topographic Maps", International Yearbook for Cartography, pp168-180.

Koeman, C. & van der Weiden, F.L. (1970), "The Application of Computation and Drawing Instruments to Structural Generalization", Cartographic Journal, Volume 7, pp47-49.

Koffka K. (1935), *Principles of Gestalt Psychology*, Harcourt Brace, New York.

Kolers, P.A. & Eden, M. (eds.) (1968), *Recognizing Patterns - Studies in Living and Automatic Systems*, MIT Press.

Lang, T. (1969), "Rules for Robot Draughtsmen", *Geographical Magazine*, Volume XLII, #1, pp55-51.

Licklider, J.C.R. (1960), from *Perspectives on the Computer Revolution*, Zenon W. Pylyshyn (ed.), Prentice-Hall (1970).

Miller, W.F. & Shaw A.C. (1968), "Linguistic Methods in Picture Processing - A Survey", *Proceedings of the FJCC*, Volume 33 part 1, pp279-290.

Miller, O.M. & Voskuil, R.J. (1964), "Thematic-Map Generalization", *Geographical Review*, pp13-19.

Montanari, Ugo (1968), "A Method for Obtaining Skeletons Using a Quasi-Euclidean Distance", *Journal of the Association for Computing Machinery*, Volume 11, #4, pp600-624.

Montanari, Ugo (1970), "A Note on Minimal Length Polygonal Approximation to a Digital Contour", Communications of the Association for Computing Machinery, Volume 13, #1, pp41-47.

Newman, William M. (1971), "Display Procedures", Communications of the Association for Computing Machinery, Volume 14, #10, pp651-660.

Nilsson, N. (1965), Learning Machines: Foundations of Trainable Pattern Classifying Systems, McGraw Hill.

Pfaltz, John (1970), "Web Grammars and Picture Description", Technical Report 70-138, Computer Science Center, University of Maryland.

Pfaltz, J. & Rosenfeld, A. (1967), "Computer Representation of Planar Regions by Their Skeletons", Communications of the Association for Computing Machinery, Volume 11, #2, pp119-123.

Robinson, A.H. & Sale, R.D. (1969), Elements of Cartography (3rd edition), John Wiley & Sons.

Ryan, T.A. & Schwartz, C.B. (1956), "Speed of Perception as a Function of Mode of Representation", American Journal of Psychology, Volume 69, pp60-69.

Seeley, D.A.R. (1970), "The Use of Finite-State Models for Pattern Detection with Application to Electroencephalographic Data", PhD. Thesis, Department of Industrial Engineering, University of Toronto.

Sherman, R. & Ernst, G.W. (1969), "Learning Patterns in Terms of Other Patterns", Pattern Recognition, Volume 1, pp301-313, Pergamon Press.

Sklansky, J., Chazen, R.L. & Hansen, B.J. (1972), "Minimum Length Polygons of Digitized Silhouettes", IEEE Transactions on Computers, C-21, #3, pp260-268.

Slagle J.R. & Lee, R.C.T. (1971), "Application of Game Tree Searching to Sequential Pattern Recognition", Communications of the Association for Computing Machinery, Volume 34, #2, pp163-170.

Srnka, Erhart (1970), "The Analytic Solution of Regular Generalization in Cartography", International Yearbook for Cartography, pp48-62.

Sukhov, V.I. (1970), "Application of Information Theory in Generalization of Map Contents", International Yearbook for Cartography, pp41-47.

Topfer, F. & Pillewizer, W. (1966), "The Principles of Selection", The Cartographic Journal, #3, pp10-16.

Uhr, L. & Vossler, C. (1963), "A Pattern Recognition Program that Generates, Evaluates, and Adjusts its own Operators", Computers and Thought, Feigenbaum and Feldman (eds.), pp251-268.

Weiner, Norbert (1960), "Some Moral and Technical Consequences of Automation", Science, Volume 131, pp1355-1358.

Zahn, C.T. (1969), "A Formal Description for 2-Dimensional Patterns", Proceedings of the International Conference on Artificial Intelligence, Washington 1969.

Zahn, C.T. & Roskies, R.Z. (1972), "Fourier Descriptors for Plane Closed Curves", IEEE Transactions on Computers, C-21, #3, pp269-281.