WORKING PAPER 464

A PILOT GEOGRAPHICAL INFORMATION SYSTEM BASED ON LINEAR QUADTREES AND A RELATIONAL DATABASE FOR REGIONAL ANALYSIS

MARK GAHEGAN AND JAMES HOGG

School of Geography
University of Leeds
Leeds LS2 9JT

July 1986

A pilot geographical information system based on linear quadtrees and a relational database for regional analysis.

by

Mark Gahegan & James Hogg

ABSTRACT

As part of a project to evaluate the potential of geographic information systems (GIS) for regional analysis of land resources, a pilot GIS based on linear quadtrees and a relational database is being developed. It consists of two parts: a spatial database of binary images represented as linear quadtrees; and a relational database of objects abstracted from the binary images and augmented by aspatial data provided by users. The two parts are integrally linked within the GIS to allow users interactive access to explore spatial objects and relationships. Preliminary results show that this approach provides flexibility and adaptability to the needs of different users of the GIS. They show also that use of an existing relational database provides a comparatively easy and quick means of developing a GIS and work is continuing to evaluate its performance.

Introduction

As part of a project to evaluate the potential of geographic information systems (GIS) for regional analysis of land resources, a pilot GIS based on linear quadtrees and a relational database is being developed. It consists of two parts: a spatial database of binary images represented as linear quadtrees; and a relational database of objects abstracted from the quadtrees and augmented by aspatial data provided by users. The two parts are integrally linked within the GIS to allow users quick access to explore spatial objects and relationships.

In this paper, we describe the structure and components of the pilot GIS and illustrate its potential for regional analysis of land resources. We focus on the use of a relational database to store information about objects abstracted from the spatial database and supplied by users.

We begin by examining briefly the need for GIS and the nature of linear quadtrees and relational databases in order to place the current work in context. Then we outline the structure and components of the pilot GIS that we are developing and illustrate how it is used. Finally we discuss briefly the scope of the GIS and draw conclusions.

GIS, quadtrees and relational databases

A geographic information system (GIS) is an integrated suite of computer programs for handling and analysing geographic data. As the volume and variety of geographic data in digital form increase, the need for more efficient procedures for integrating such data into GIS and for analysing complex inter-relationships becomes more pressing. The availability of massive volumes of geographic data from satellite remote sensing and raster scanning of archival maps not only expands the range and sophistication of potential applications of GIS, but also highlights the need for more flexible and efficient GIS which have the capacity to deal with large volumes of geographic data and to respond interactively to queries involving dynamic computation of textual and geometric properties.

Most GIS provide the capacity to store, manipulate and display spatial data such as cartographic data for automated mapping. They have generally been developed independently, on different computers and over varying periods and are often optimised for particular applications. There exists therefore a diversity of GIS (Marble and Peuquet, 1983; Tomlinson, 1984). In spite of this diversity, Peuquet (1984) identified the different types of digital data models for storage and retrieval of geographic data and classified and discussed them in a critical review (fig 1). She found that most GIS were based on vector, raster or some hybrid type of data model and that many of them were experiencing severe problems of inefficiency. She attributed this to two major shortcomings: (i) to a rigidity and narrowness in the range of applications and types of geographic data which could be accommodated and (ii) to unacceptably low levels of efficiency of storage and speed of use with increasing volumes of data. Peuquet (1984) speculated however that major advances in the performance of spatial data models and GIS were likely over the next few years and that these would allow us to meet the needs for efficiency, versatility and integration.

A critical factor in achieving these improvements is the choice of spatial data model for the GIS. Among the traditional spatial data models, vectors and rasters are probably most widely used, but these suffer from major shortcomings identified by Peuquet (1984). Attention has focused increasingly therefore on hierarchical data models to represent large volumes of spatial data in a compact form which allows efficient retrieval and processing. One such data model is the quadtree (Samet, 1984).

Regions as quadtrees

Quadtrees have been studied extensively over the last decade and a large number of algorithms have been published for performing operations on them. Early work on the subject, like that of Klinger

(1971) and Finkel and Bentley (1974), stimulated much of the later work on a large number of hierarchical data models, including quadtrees (Oliver and Wiseman,1983; Rosenfeld et al. 1982, 1983; Samet et al. 1984, 1985; Abel, 1985). Samet (1984) provides a review of quadtrees and related hierarchical data models which are based upon recursive decomposition of an image.

A quadtree is constructed from a square binary array of pixels which represents an image. The set of connected black pixels in the image is referred to as a region, such as an area of forest (Fig 2). If we assume that an image is comprised of a $2^n \times 2^n$ binary array of pixels (Fig 3), then a quadtree encoding represents this image by recursively sub-dividing it into four quadrants until no further sub-division is necessary (Fig 4). This occurs when we obtain square blocks (possibly single pixels) which are homogeneous in value (i.e. either all black or all white) or when we reach the level of resolution that we require. This process is represented by a tree with four branches, in which the root node corresponds to the whole image and the four sons of the root node correspond to the four quadrants (Fig 5). The leaf nodes of a quadtree correspond to homogeneous blocks for which no further sub-division is necessary. Nodes at level n represent square blocks of size $2^n \times 2^n$. Further details of this process are described by Hogg and Gahegan (1986) and Samet (1984).

While a large number of data models for quadtrees have been proposed, that of Gargantini (1982) is probably most attractive because it is economical in its requirements for space. Gargantini(1982) proposed a data structure to represent quadtrees which is known as a linear quadtree. It is in the form of a linear list consisting of the quadtree nodes in some order of traversal of the tree. A number of different forms of keys is available to map a set of ancestors to a numeric key (Morton, 1966; Gargantini 1982; Abel and Smith 1983). Each node possesses a unique key, which can be processed to show the level at which the node was formed and its x,y address in the original image. Two quadtree files can be processed together (to find the intersection or union) simply by reading the nodes one at a time. Alternatively, nodes can be matched to x,y addresses by decoding their keys in a bitwise fashion (Peuquet, 1984). This provides a flexible means of accessing the data represented by the quadtree.

Objects in Relational database

An object is an entity that exists in the real world and possesses attributes that we are interested in (Oxborrow, 1986). An object, such as a particular reservoir or wood, can be related to its location in an image by its unique position in the linear quadtree. It may have many attributes which are specific to it alone, such as size or shape. In addition however objects may have aspatial attributes such as ownership of a region of land, population of a census district or fertilizer treatments on a particular field of crops. Quadtrees are generally unsuitable for storing this type of data, so an alternative must be found.

One ready alternative is a relational database. This allows objects and their attributes to be organised in a way that shows their logical relationships. Furthermore, a relational database management system provides quick access to data along a variety of access paths, so many types of query can be satisfied. Detailed discussion of the theory and applications of relational modelling is beyond the scope of the current study, but there has been considerable research into the subject over the past 15 years. The concept of the relational model originally goes back to Codd (1970) but many others have contributed to the development of the concept. For a detailed discussion of these, readers should refer to relevent chapters in Date (1986), Ullman (1984) and Oxborrow(1986).

The application of relational models for interactive geographical data processing has been explored by several workers including Steiner and Gilgen (1984), Smith and Pazner (1984) and Abel(1986).

Structure and Components of pilot GIS

The structure and major components of the pilot GIS, and the relationships between these, are shown in figure 6. The function of each component is described below.

<u>Encoding binary images as linear quadtrees</u>

The program for quadtree encoding accepts binary, square arrays or images which have been generated by vector to raster scan conversion, by raster scanning of maps or by remote sensing systems. A recursive routine is used to encode such images by progressively viewing smaller areas of the image until a homogeneous area or region is found (i.e. pixels have the same colour or value) and a node is formed. The linear quadtree that is formed is stored on disk.

<u>Operations on linear quadtrees</u>

A set of functions can be applied to the linear quadtrees to perform various operations. These include a set of low level functions which are often required by higher level operations, for example, functions for returning the x,y address or colour of a node, the neighbour of a node or traversing along a line across an image. The advantage of separating such low level functions is not only to avoid repetition in higher level functions but to allow the structure of the quadtree to be modified without affecting the validity of the higher levels.

Functions for set operations on linear quadtrees form the backbone of the GIS for regional analysis. They allow for intersecting two quadtrees to find areas common to both, for union of two quadtrees to find the amalgamation and for finding the complement of a quadtree.

More complex functions perform various operations on linear quadtrees. These include functions for windowing, where a relevant sub-section is extracted from a quadtree; for traversing the border of a quadtree to return the length of the border (Samet and Tamminen, 1985); and for labelling all the separate regions or components. The labelling function is an extension of the border program that recognises and labels the separate components of an image. A large number of useful functions for operations on linear quadtrees have been devised and reported in the literature. Many of these appear to be potentially useful for regional analysis of land resources but have yet to be added to the GIS.

Functions for many operations on linear quadtrees have been implemented in the C programming language on a Vax 11/780 computer running Berkeley Unix 4.2.

Relational database

The relational database is used to store information about objects. By applying the border function, the separate components of a quadtree can be identified, labelled and stored in the database as objects. Thus these provide an alternative way of addressing a query i.e. by object as opposed to by region.

The relational database INGRES was used to store all information about objects in this study (Stonebraker et al.1976;). It was chosen for three major reasons: (i) INGRES is a relational database, the query language (QUEL) being strongly based on relational calculus (Stonebraker et al 1976; Date, 1986). Searches to satisfy a query may proceed across several relations, searching on any attribute or combination of attributes. New relations can readily be added as new data is brought to light without damaging the integrity of the design. (ii) INGRES can readily be interfaced to C programs of the GIS by the preprocessor, EQUEL. This allows commands to INGRES to be integrated into programs at appropriate points, while keeping the database invisible to users. (iii) One of the authors has considerable knowledge and previous experience of using INGRES in earlier research and this was judged important for making rapid progress in the time available.

Query language

A query language forms the interface between the user and the

GIS. It governs the operation of functions and the opening and closing of files holding linear quadtrees in the spatial database. Users enter queries by a single capital letter e.g. U (union); I (intersection); N (complement). For example, if we wish to find the intersection of grit rocks and gley soils between heights 500 - 700 feet, we enter the compound query:

grit I gley I (height500_600 U height600_700)

The inner-most expression in parenthesis is evaluated first. The results from this expression are then processed with the next expression. Results from the above query can be formed into a new quadtree by prefixing the query with a file name and an equals sign:

high_ground = (height900_1000 U height1000_1100)

or sent directly to the plotter to produce a map (Hogg, Gahegan & Stuart, 1986).

Plotting software

Output from the GIS is currently displayed on a Versatec electrostatic plotter which is capable of producing black and white maps up to A4 size (Fig 7). While this gives a high level of resolution, it places restrictions on size of output and colour. All quadtrees that are to be plotted as maps are first processed by software for document production which is provided by the Unix operating system ("pic" and "vtroff"). This converts them to a form which is suitable for plotting.

Database interface.

The result of applying the component labelling function to a quadtree is a quadtree where each node is labelled to show to which separate object it belongs. Information about these separate objects, such as their location, size, shape and orientation, can be collected during the second pass of the component labelling function and stored in the database.

Suppose a user wishes to study all areas of urban land that occur within a selected study area. He can use the component labelling function to identify each separate area by tracing its perimeter. A relation can then be created automatically (called "urban_land"), and a tuple appended to this corresponding to each individual urban area, as it is recognised by the component labelling function. A structure for this relation is given below:

label	x1	x2	y1	y2	size

For example:

1	114	135	86	117	206
2	188	245	177	233	873

'label' corresponds to the number assigned automatically to that particular object by the component labelling function. The four figures X1, X2, Y1 and Y2 are the diagonally opposite coordinates of the rectangle that neatly surrounds the object (Fig 8). The figures, along with the area of the object (i.e. the number of pixels it covers) are easily collected during the second pass of the component labelling function. Hence any queries about the area or position of objects can now be answered directly by referring to the database.

Objects can now be considered as separate entities and processed individually. For example, "urban_land_1" can be extracted from the urban_land quadtree by taking only those nodes whose label is 1 . If a labelled version of the quadtree is no longer available, (it may have been deleted to save space), then the two pairs of diagonal coordinates denoting opposite corners of the rectangle that surrounds it can be used as an index to the relevent parts of the quadtree . The quadtree has of course to be reprocessed to locate the object in question but this can be achieved efficiently especially if the quadtree can be indexed using a B-tree memory management system (Abel, 1984).

By forming a database of objects, it becomes possible to store aspatial data, related to individual objects, which can be supplied from various sources. For example, if census districts are stored in the spatial database, then census statistics relating to each district or region can be linked. The type of information added depends on the nature of the objects, but there is no reason why several relations, all holding different aspatial data, cannot be added to the database as a need arises. In this way the users can build up the relations that they require to solve problems in their particular area of study. For example, in urban and regional planning the facilities of the relational database can readily be used for regional modelling of flows between regions.

The database can be used to store various definitions of objects which can then become a part of the query language. For example, the definition of 'high ground'

high_ground = height900_1000 U height1000_1100

could be stored in a relation enabling the query parser to relate the term 'high_ground', when it is employed by the user, to a specific definition that has been entered previously.

Other information relating to the region as a whole, again obtained from the component labelling function, can readily be added along with these definitions. This might include, for instance, the number of occurrences of an object, the total area and total perimeter length. Hence information at a higher level can be obtained from the database without referring to the quadtrees again.

An example to illustrate operations

Examples of the response of the GIS to queries entered by users are given. Requests to the database are formed by the query parser at run-time as a response to a user query, and are passed to the database. The results obtained from the database can then be processed by other parts of the GIS. As an example of this, consider the case when a user requests that all objects of type "woods" are discovered and placed in the database.

Firstly the GIS searches the relational database for an object called "woods" and retrieves the definition of this object. Secondly, the definition is passed to the query parser which, with the aid of the set operations, processes the query and forms a new quadtree as a result (Fig 9). Thirdly, a new relation is formed, if one does not already exist, to store the new information on the objects of type 'wood'. Fourthly the new quadtree is passed to the component labelling function which recognises and automatically labels the separate parts of the quadtree (Fig 10). Fifthly, a tuple is appended to the new relation that contains data corresponding to each individual part or object encountered (fig 11). Finally, information about that class of objects (e.g. number, size, perimeter length, etc.) is added to the relational database and the initial processing is complete.

After the initial processing of geographic data, users can employ the GIS to answer queries about objects by referring only to the relational database, thus avoiding the need to reprocess quadtree images. For example, the queries:-

find all woods covering an area > 3000

find all woods in the area x1, x2, y1, y2

find total and average size of all woods could be answered immediately from the relational database.

Aspatial data can be included in the relational database, allowing greater flexibility in the types of geographic data that can be used and in the types of queries that can be addressed (Fig 12). For example:-

find the size of Holly wood

what percentage of the woodland areas is coniferous

In this way, two sets of data exist within the GIS. The first is a set of spatial quadtrees, which are required to begin with. The second is a set of relational data which corresponds to the various objects contained within the quadtrees and is generated gradually as the quadtrees are used. It can then be used as an alternative, faster means of satisfying some types of query, since it is already in a processed form. Aspatial data can be added to this, to enhance the flexibility of the system.

Discussion

Individual users can share data in both databases, or have their own dedicated set of quadtrees and relations or some combination of shared and individual databases as required. This allows users from different disciplines, for example to share the same satellite remote sensing data but to include different types of ground data to meet individual requirements. It offers a flexible approach.

In building the GIS, we have adopted a pragmatic approach in order to get a working system within a relatively short time. We recognise however that the structure of the database is not ideal and represents a balance between various options. For example, it would probably be better to express the relationship between information on a single object and information on a class of the same object as a hierarchy, as in the extended relational model of Codd (1979). The use of such an extended relational database would be worth investigating for applications of this nature.

The need for a colour graphics display to reap the full benefits from the GIS is apparent to users. The speed of operation only becomes an advantage when users have interactive access and can see the response to their queries as a colour map on a high resolution display. Such a display would allow us to make far better use of functions such as those for windowing and generalisation of maps which are major benefits of the quadtree representation.

Conclusion

The pilot study described here demonstrates the feasibility of building a GIS using linear quadtrees to store spatial data and a relational database to store objects to enhance flexibility and increase the level of performance of the GIS. Linear quadtrees provide rapid access to multiple binary images and give the GIS performance characteristics which are satisfactory for interactive analysis in geographical research. Use of an existing relational database provides a means for storing objects and is comparatively easy and quick to implement. It gives acceptable levels of performance though it is stressed that tests have so far been carried out on a relatively small set of data. Full evaluation of the GIS will require construction of a larger database of images and objects.

One advantage of using an existing relational database is the comparative ease and speed with which a flexible and powerful pilot GIS can be developed to handle spatial and aspatial data. The pilot GIS provides opportunities for evaluating the potential of GIS in a range of different applications and has thus enabled geographers and other field scientists to further their knowledge and appreciation of GIS. This advantage must be considered in relation to inherent limitations in adapting existing relational technology to specific needs and in particular to how closely it can be adapted to achieve satisfactory levels of performance for various users of the GIS. At this stage, the benefits appear to outweigh the limitations at least in the short term. In the long term, it may prove to be more efficient to build special object databases which are tailored explicitly to the needs of users of geographic data.

References

Abel, D.J.& Smith, J.G. 1983 A data structure and algorithm-based on a linear key for a rectangular retrieval problem. Computer Vision, Graphics & Image Processing, Vol.24,pp1-13

Abel, D.J. 1984 A B-tree structure for large quadtrees. Computer Vision, Graphics & Image Processing, Vol 27, pp 19-31.

Abel, D.J. 1985 Some Elemental Operations on Linear Quadtrees for Geographic Info Systems. Computer Journal, vol 28, No1 pp73-77.

Codd, E.F. 1970 A relational model of data for large shared data banks. Comm of ACM Vol 13 no 6,pp 377-387.

Codd, E.F. 1979 Extending the database relational model to capture more meaning. ACM Trans. Database Systems Vol 4 No 4 pp 397-434.

Date, C.J. 1986 Database systems. Addison-Wesley,(4th edition),pp 209-228

Finkel, R.A. & Bentley, J.L. 1974 Quad Trees: A data structure for retrieval on composite keys. Acta Inf. 4, 1, pp 1-9

Gargantini, I. 1982 An Effective Way to represent Quadtrees. Communications of A.C.M. vol 25 No. 12 pp905-910.

Hogg, J. Gahegan, M. & Stuart,N. 1986 Evaluation of regional land resources using geographic information systems based on linear quadtrees. Proc. ISPRS Commission VII, Enschede, Holland, 25-29th August. pp 9

Hogg, J. & Gahegan, M. 1986 Regional analysis using geographic information systems based on linear quadtrees. Proc. ISPRS Commision IV Edinburgh, Scotland, 12-17th Sept pp 11

Klinger, A. 1971 Patterns and search statistics. In: Optimising Methods in Statistics by J. S. Rustagi (ed) Academic Press, London, pp 303-337.

Marble, D & Peuquet, D 1983 Geographic Information Systems and Remote Sensing. Chapt 22 in Manual of Remote Sensing, American Society of Photogrammetry, Falls Church, Va. 2nd Ed. Vol 1.

Morton,G. 1966 A computer oriented geodetic data base and a new technique in file sequencing. IBM Canada Ltd, unpublished report.

Oliver, M.A.& Wiseman,N.E. 1983 Operations on quadtree encoded images. The Computer Journal, Vol.26, No 1, pp 83-91.

Oxborrow, E. 1986 Databases and database systems: concpets and issues. Chartwell-Bratt Ltd, Bromley, Kent, U.K. p254

Peuquet, D.J. 1984 A conceptual framework and comparison of spatial data models. Workshop W2, International Symposium on Spatial Data Handling, 1984, Zurich, pp.80.

Rosenfeld, A. Samet, H Shaffer, C. Webber, R.E. 1982 Application of hierarchical data structures to geographical information systems. TR-1197 Computer Vision Laboratory, University of Maryland. pp 160

Rosenfeld, A. Samet, H. Shaffer, C.A. Webber R.E. 1983 Application of heirarchical data structures to geographical information systems: Phase II. TR-1327 Computer Vision Laboratory, University of Maryland, pp76

Samet, H. 1984 The quadtree and related hierarchical data structures. ACM Computing Surveys 16, 2:187-260.

Samet,H. Rosenfeld, A. Shaffer, C.A Nelson, R.C. Huang, Y.G. 1984 Application of hierarchical data structures to geographical information systems: Phase III CS-TR-1457 Center For Automation Research, University of Maryland.

Samet H. Rosenfeld A. Shaffer,C.A. Nelson,R.C. Huang,Y.G. Fujimura, K. 1985 Application of hierarchical data structures to geographical information systems: Phase IV CS-TR-1578 Computer Science Department, University of Maryland, pp 61.

Samet, H. & Tamminen M. 1985 Computing geometric properties of images represented by linear quadtrees. IEEE Trans. Patern Analysis and Machine Intelligence, 7,1

Smith,T.R. & Pazner,M.I. 1984 A knowledge based system for answering queries about geographical objects.Proc. 9th William T. Pecora Memorial symposium on spatial information technologies for remote sensing today and tomorrow. pp 286-289.

Steiner, D & Gilgen, H. 1984 Relational modelling as a design and evaluation techique for interactive geographical data processing. Proc. Int.Symp. on Spatial Data Handling, Zurich, Switzerland. pp 61-77

Stonebraker,M. Wong, E. Kreps,P. & Held, G. 1976 The design and implementation of Ingres. ACM Trans. Database Systems. Vol 1 No 3 pp

189-222.

Tomlinson, D.J. 1984 Geographical Information Systems - A New Frontier. <u>Proc. Int. Symp. Spatial Data Handling,</u> 1984, Zurich.

Ullman, J.D. 1982 <u>Principles of database systems.</u> Computer Science Press, London. p 483.

.....

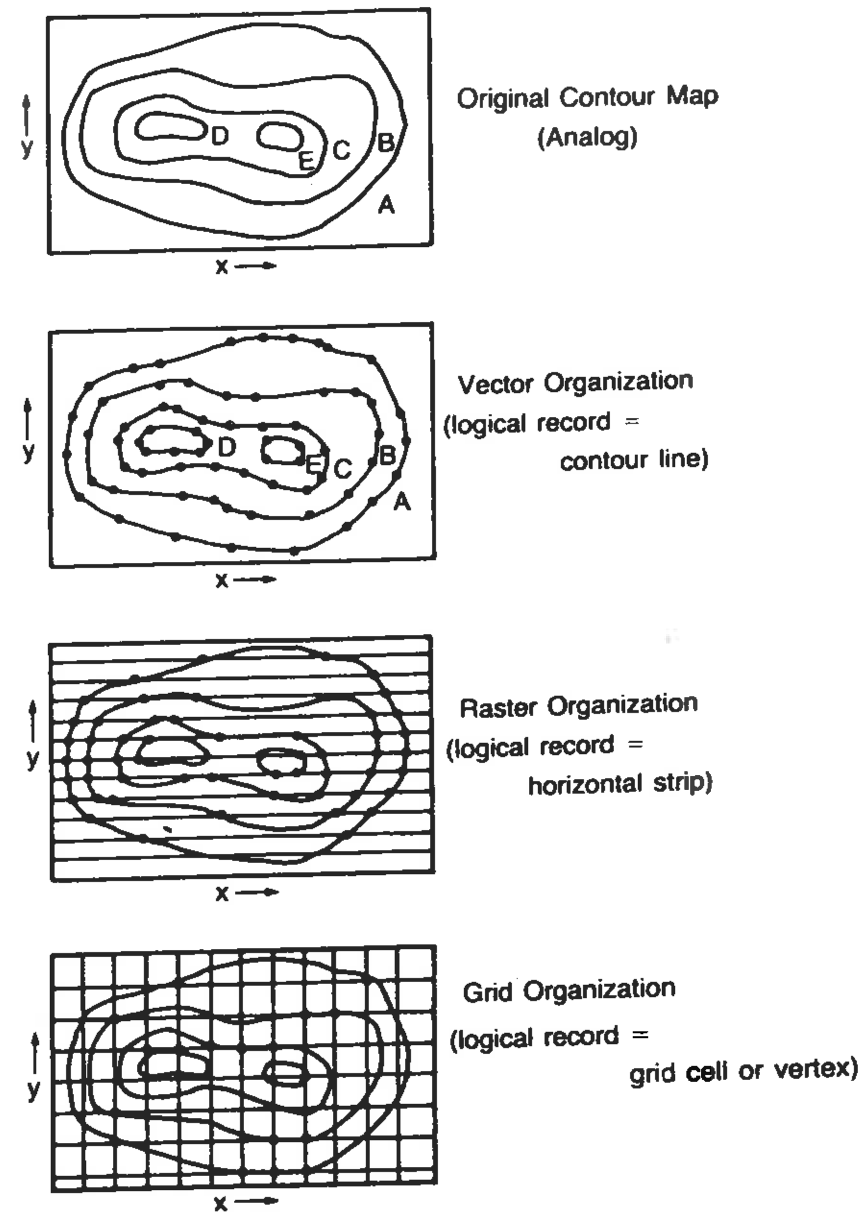


Fig I Types of data organisation for vector and raster spatial data structures (after Marble and Peuquet, 1983).

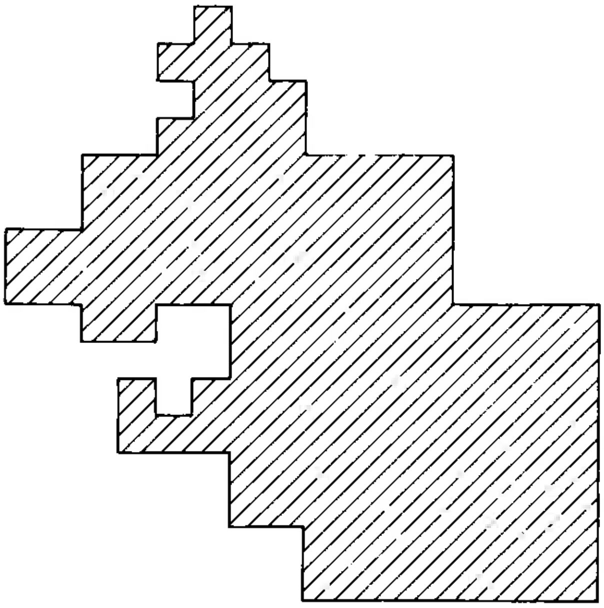


Fig 2 A region such as a field or a forest.

0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
0	0	0	0	1	1	1	0	0	0	0	0	0	0	0	0
0	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0
0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0
0	0	1	1	1	1	1	1	1	1	1	1	0	0	0	0
0	0	1	1	1	1	1	1	1	1	1	1	0	0	0	0
1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0
1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0
0	0	1	1	0	0	1	1	1	1	1	1	1	1	1	1
0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1
0	0	0	1	0	1	1	1	1	1	1	1	1	1	1	1
0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1
0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1
0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1
0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1

Fig 3 A binary image of Fig 2.

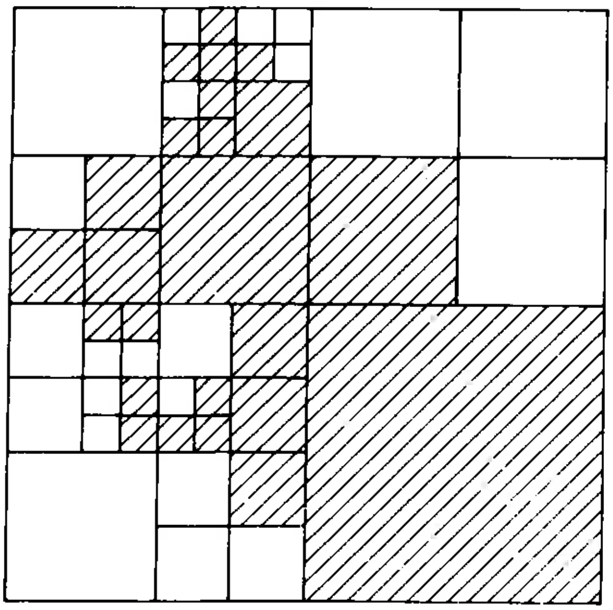


Fig 4 Block decomposition of the region in Fig 2 showing the maximal blocks.

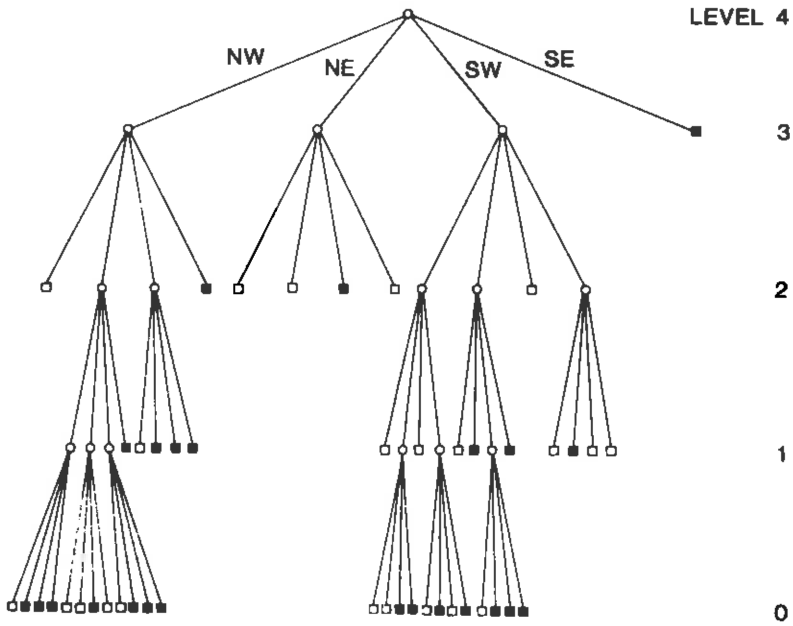


Fig 5 Quadtree representation of the blocks in Fig 4 with black nodes shown by solid dots

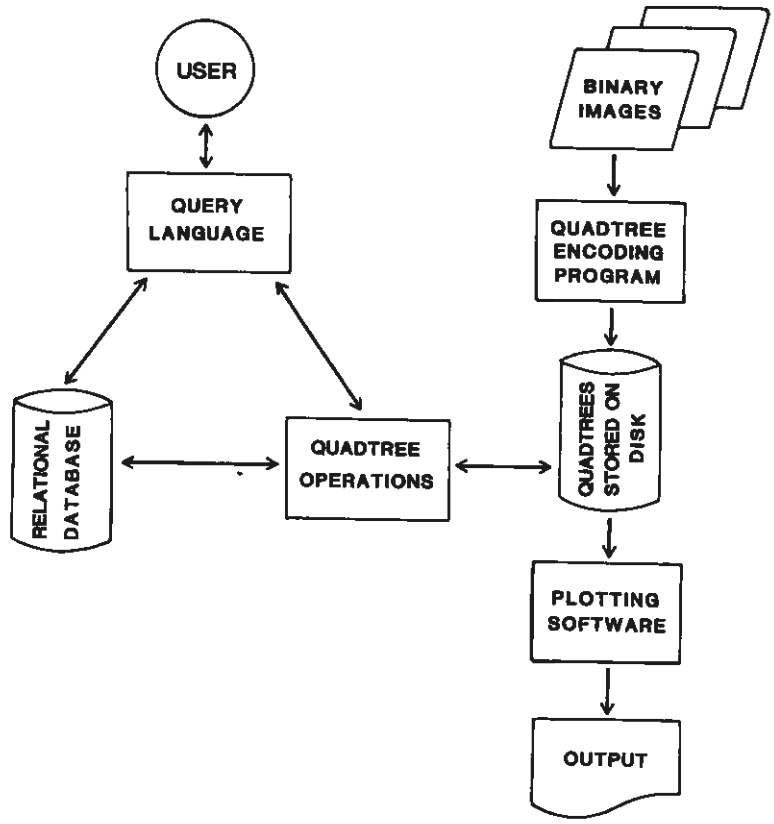


Fig 6 Structure and components of the pilot GIS.

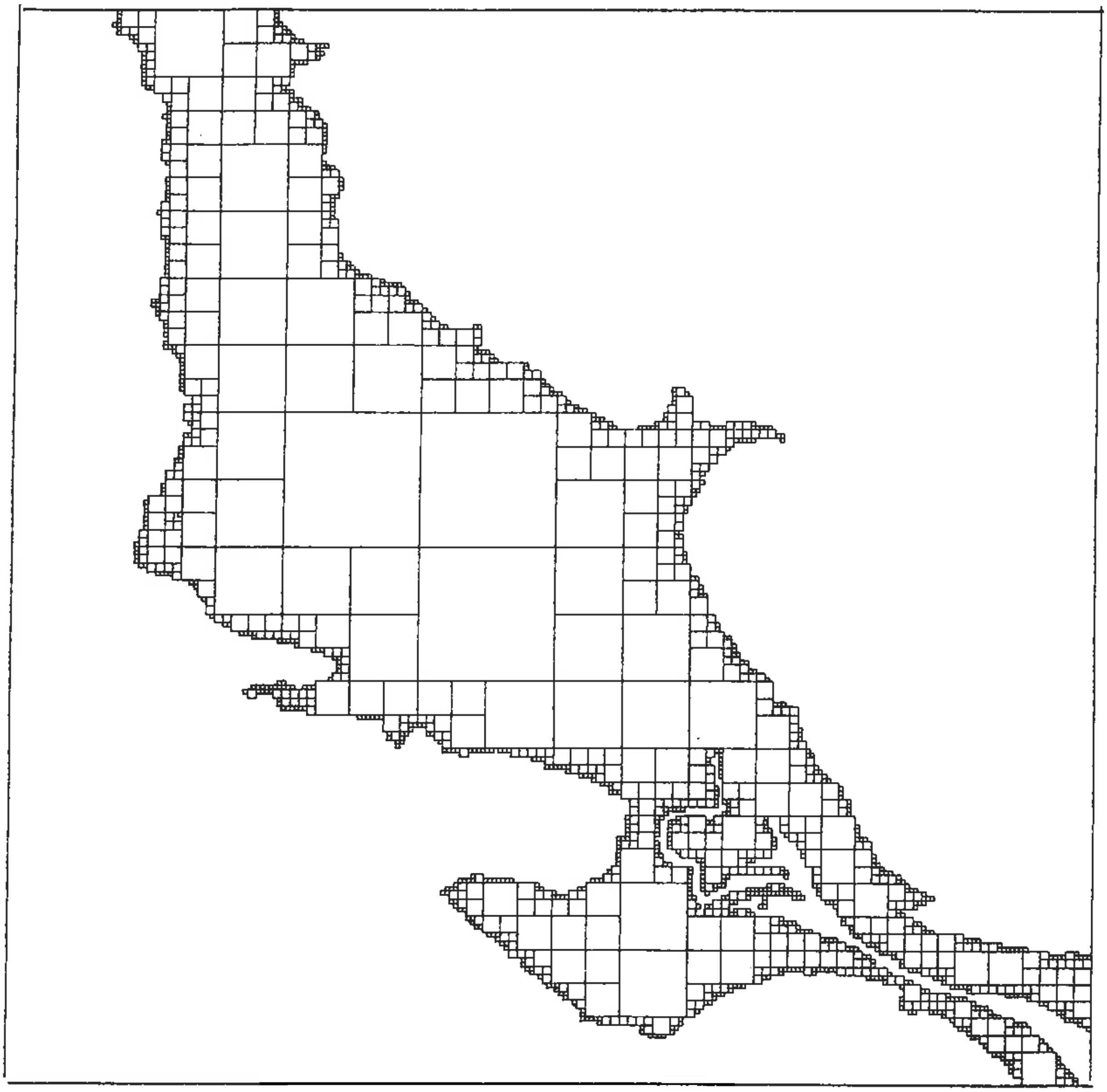


Fig 7 A quadtree map output on the Versatec electrostatic plotter showing heights between 300 and 400 feet.

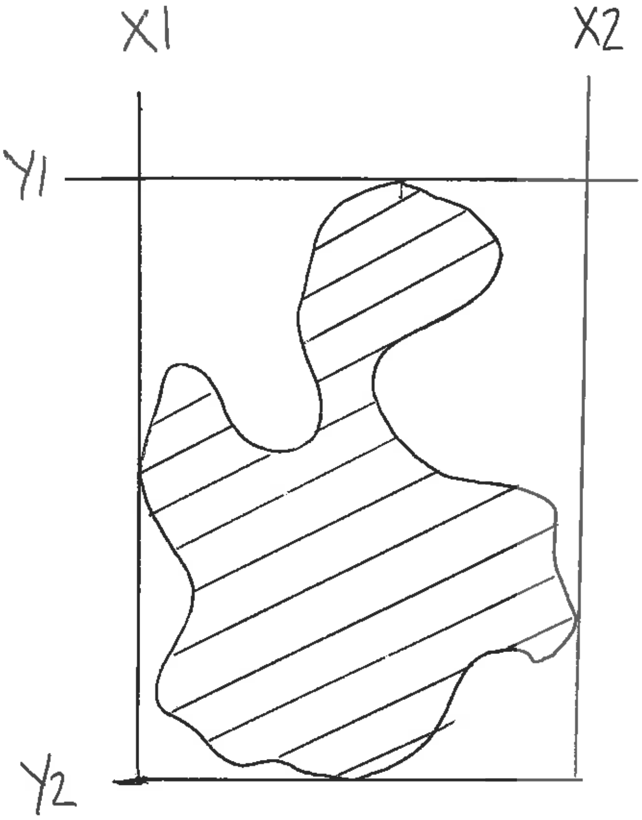
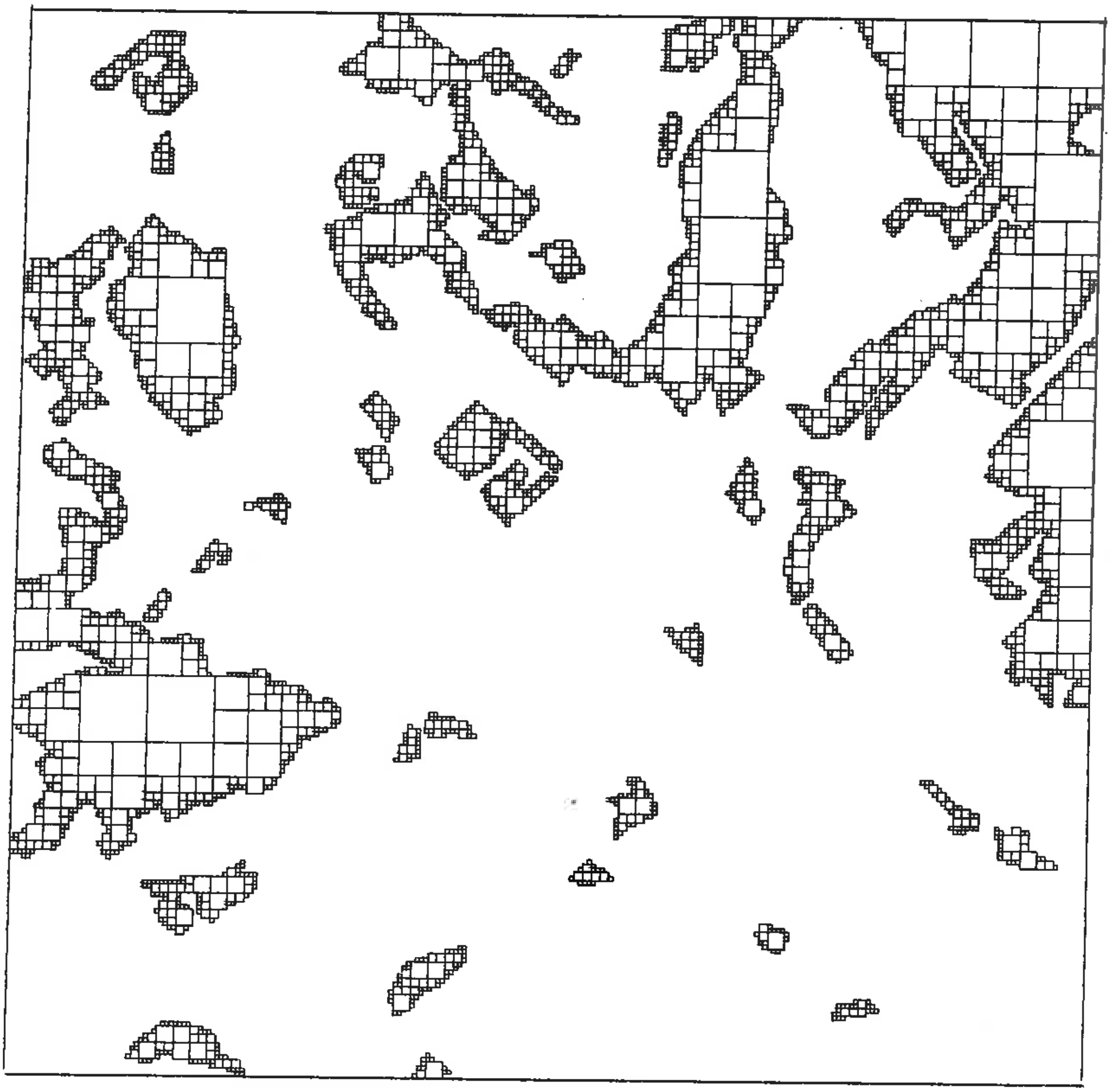


Fig 8 A rectangle is used to neatly surround an object or region in order to define its location.



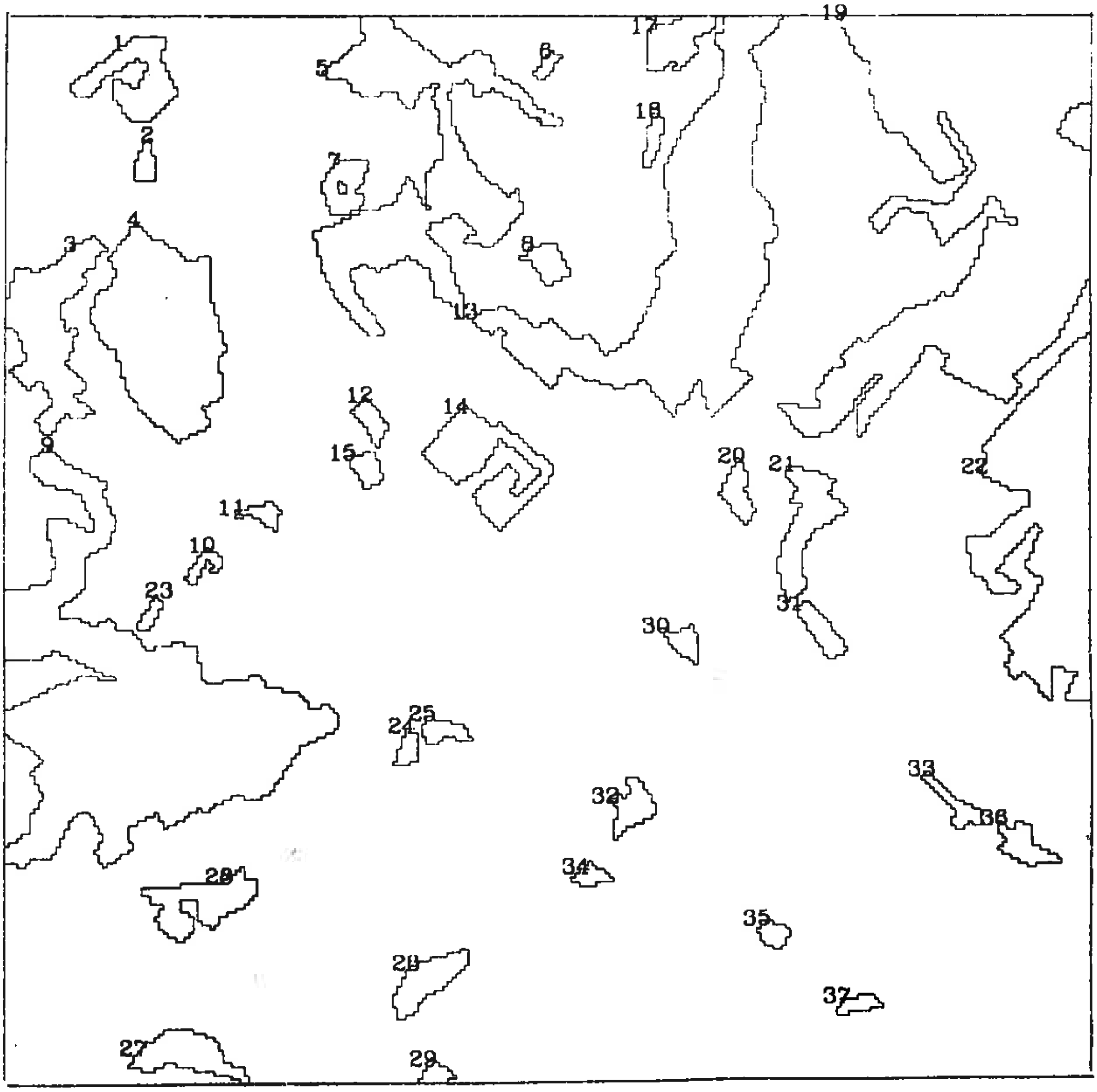


Fig IO Map output from the component labelling function which recognises and automatically labels the separate parts of the woods map in Fig 9.

woods relation

label	x1	x2	y1	y2	size
1	15	40	5	25	241
2	30	35	29	39	37
3	0	24	52	100	529
4	20	52	49	102	1056
5	72	131	0	76	1383
6	124	131	8	15	22
7	74	85	34	47	99
8	121	133	54	64	67
9	0	78	103	204	3066
10	42	51	127	135	32
11	54	65	116	123	36
12	81	90	91	103	49
13	109	183	0	96	2196
14	99	129	93	112	237
15	80	89	104	113	48
16	110	128	107	123	123
17	151	167	0	13	140
18	150	155	23	36	38
19	182	256	0	101	3775
20	168	177	106	122	68
21	182	199	108	141	233
22	227	256	76	165	1500
23	31	37	139	147	24
24	91	97	171	180	35
25	98	110	168	174	48
26	32	59	204	222	221
27	29	57	243	256	178
28	91	109	224	241	141
29	97	106	250	256	31
30	154	163	146	150	45
31	187	199	141	155	71
32	142	153	183	198	85
33	216	230	183	196	52
34	133	143	203	209	32
35	177	185	218	225	37
36	234	249	194	204	78
37	196	207	236	241	34

Fig I1 The woods relation showing data corresponding to each separate component in the woods map

label	name	type
1	Holly wood	mixed
3	Hill wood	deciduous
4	Hillcar wood	mixed
5	Northwood carr	deciduous
19	Seventy acre plantation	coniferous

Fig I2 Aspatial data can be added to the relational database such as names and types of woods from the O.S. map.