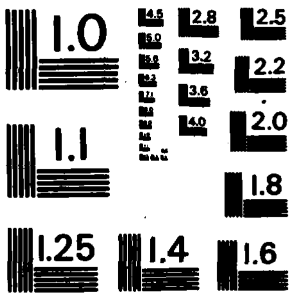
AD-A135 532 HEXAGONAL DATA BASE STUDY(U) INTERACTIVE SYSTEMS CORP LITTLETON CO T R BUNDY SEP 83 DAAK70-82-C-0133 1/1

UNCLASSIFIED F/G 9/2 NL

END
DATE
FILMED
1 84
DTIC


1.0
1.1
1.25
1.4
1.6
1.8
2.0
2.2
2.5
2.8
3.2
3.6
4.0
4.5

(6)

ETL-0338

Hexagonal data base study

AD-A135 532

Thomas R. Bundy

Interactive Systems Corporation
5000 South Sycamore Street
Littleton, Colorado 80120

September 1983

Prepared for

U.S. ARMY CORPS OF ENGINEERS
ENGINEER TOPOGRAPHIC LABORATORIES
FORT BELVOIR, VIRGINIA 22060

E T L

Unclassified
SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER ETL-0338	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) Hexagonal Data Base Study		5. TYPE OF REPORT & PERIOD COVERED Final Report August 1982 - August 1983
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) Thomas R. Bundy		8. CONTRACT OR GRANT NUMBER(s) DAAK 70-82-C-0133
9. PERFORMING ORGANIZATION NAME AND ADDRESS Interactive Systems Corporation 5500 South Sycamore Street Littleton, Colorado 80120		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS U.S. Army Engineer Topographic Laboratories Ft. Belvoir, Virginia 22060		12. REPORT DATE September 1983
		13. NUMBER OF PAGES
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)		15. SECURITY CLASS. (of this report) Unclassified
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE
16. DISTRIBUTION STATEMENT (of this Report) Approved for public release; distribution unlimited.		
17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)		
18. SUPPLEMENTARY NOTES		
19. KEY WORDS (Continue on reverse side if necessary and identify by block number) Hexagonal data structures, automated mapping, polygon processing		
20. ABSTRACT (Continue on reverse side if necessary and identify by block number) This is a final report for a study in the application of hexagonal data structures to handling geographic information. This report presents the results of an experimental software implementation utilizing hexagonal data structures as applied to test data base.		

Accession For	
NTIS GRA&I	☑
DTIC TAB	☐
Unannounced	☐
Justification	
By	
Distribution/	
Availability Codes	
Dist	Avail and/or Special
A-1	

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

Unclassified
SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

PREFACE

This report was produced under contract DAAK 70-82-C-0133. The report was prepared for the U.S. Army Engineer Topographic Laboratories (ETL), Fort Belvoir, Virginia 22060. The Contracting Officer's Representative was Mr. Carl S. Huzzen. This report was prepared by Mr. Tom Bundy with the assistance of members of the Interactive Systems Corporation staff.

CONTENTS

List of Figures

List of Tables

CHAPTER 1

INTRODUCTION

Since 1980 Interactive Systems Corporation (ISC) has been developing software systems to process geographic information. The project described in this report was performed to assess the capabilities of the techniques in data representation and handling which have evolved during that time. The system functions to be tested were specified by the U.S. Army Engineer Topographic Laboratories (ETL), the agency for which the study was performed. ETL also supplied a small test data set.

The ISC approach to the representation of this type of information in a computer is sometimes referred to as a hexagonal structure. This derives from the fact that locations are expressed not in terms of cartesian coordinates of points but rather in a coordinate system called GBT which uses strings of octal integers to address hexagonal regions. The introduction to the mathematics of this system is presented in Appendix B.

Chapter 2

THEORY AND IMPLEMENTATION

2.1 Overview of the Software

The work of Interactive Systems in hexagonal data structures and their application to geographic information processing has resulted in a commercial software system, AGIS. This software has four main subsystems as shown in Figure 2.1:

- IGC (Interactive Graphics Controller)
  - Controls all graphical functions
- GRAM (GBT Record Access Manager)
  - A hexagon based spatial data manager
- RIM (Relational Information Managment System)
  - A commercially available relational data manager
- ODM (Overlay Display Manager)
  - Controls all mapping functions.

These four modules each comprise libraries of subroutines that can be called by applications programs.

AGIS was used to perform the tasks required by the ETL study. Additional applications programs were written. Much of the polygon processing code was part of this effort. AGIS was enhanced by the addition of such capabilities as overlay intersection, polygon network verification, and polygon display by attribute.

AGIS

AUTOMATED GEOGRAPHIC INFORMATION SYSTEM
SOFTWARE MODULES

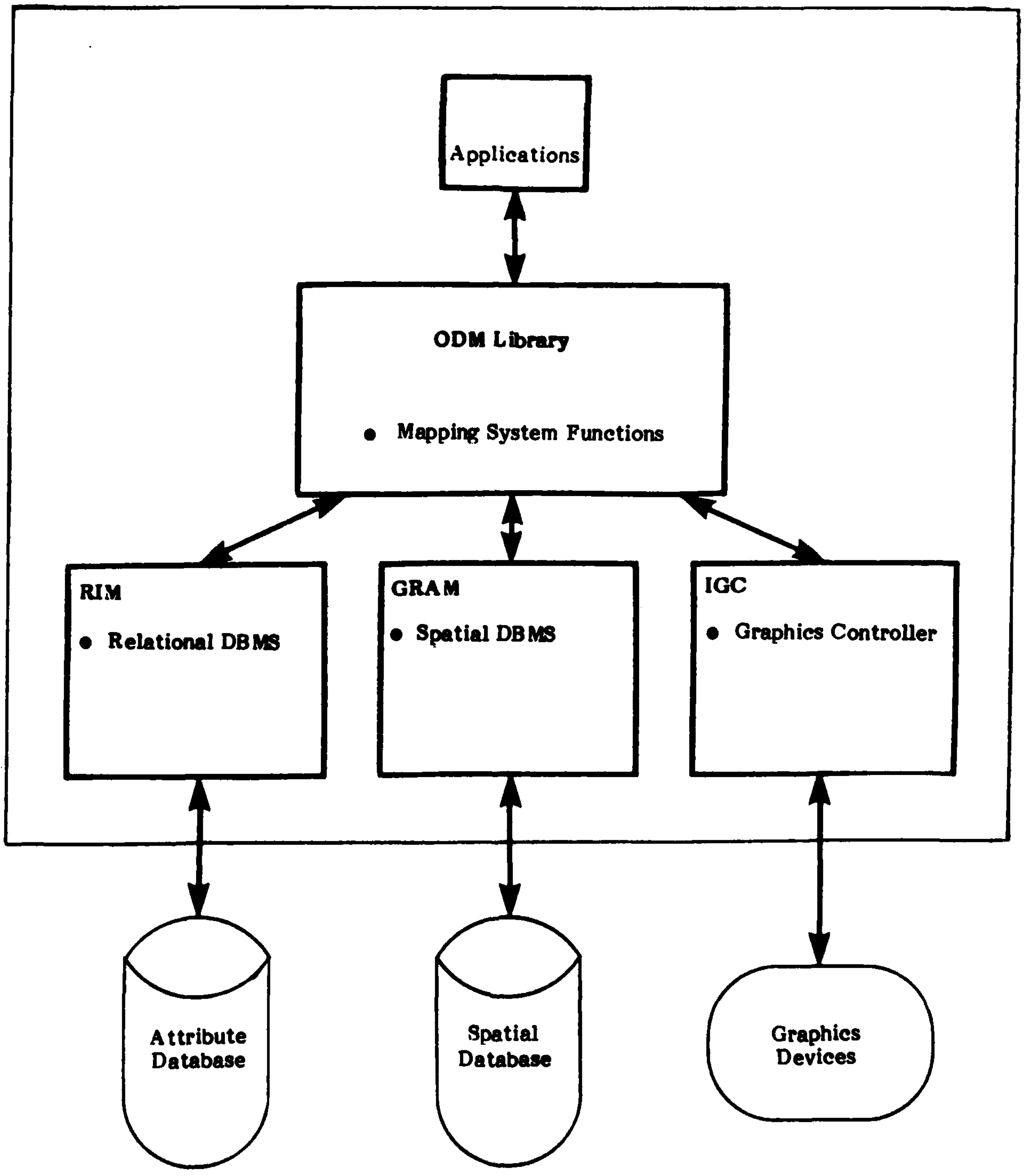


Figure 2.1 The functional relationship between the AGIS subsystems.

2.2 Nodes and Relations

The hexagonal data structure which underlies AGIS is designed to handle spatial information, that is, data for which location or position in some geometric reference system is the key attribute. The basic entity in this data structure is called a node and represents either a hexagonal region or an aggregate of hexagons. In either case, the node has a corresponding GBT address. Data coordinates in other reference systems are transformed into the GBT system. In mapping a data set into the GBT system, consideration is given to the resolution of the underlying GBT grid. For example, if no two points in the data set are closer than 50 meters, a grid built so that the distance between adjacent hexagon centers represents 50 meters or less will insure that no hexagon contains two data points. In most applications the resolution of the grid is selected to be less than the resolution of the data so that point data will be represented by a single node in the data structure.

There are two basic relations defined on nodes which permit more complex data such as lines or polygons to be represented in the data structure. These relations are:

- connectivity
- set membership.

A node can be connected to each of any number of other nodes. The number of nodes to which it is connected is called its degree. There are many kinds of node connectivity (see Figure 2.2). The connectivity type used is a function of the implemenation. For example, connectivity can have an associated equation defining a curve between the points represented by the two nodes. This is useful in graphics applications. The connectivity used in this study, the "packed vector" form, associates a sequence of intermediate points with the node connection. A more detailed explanation of this form will be given in a later section.

Lines and polygon boundaries are examples of node sets. In applications, other attributes may define such sets. For example, "county line", "interstate highway", and "land use area" all give rise to node sets. The nodes in these sets are connected to each other in such a way that lines or boundaries are accurately represented. Area features are defined by their boundaries. Figure 2.3 shows some node sets representing map data. An

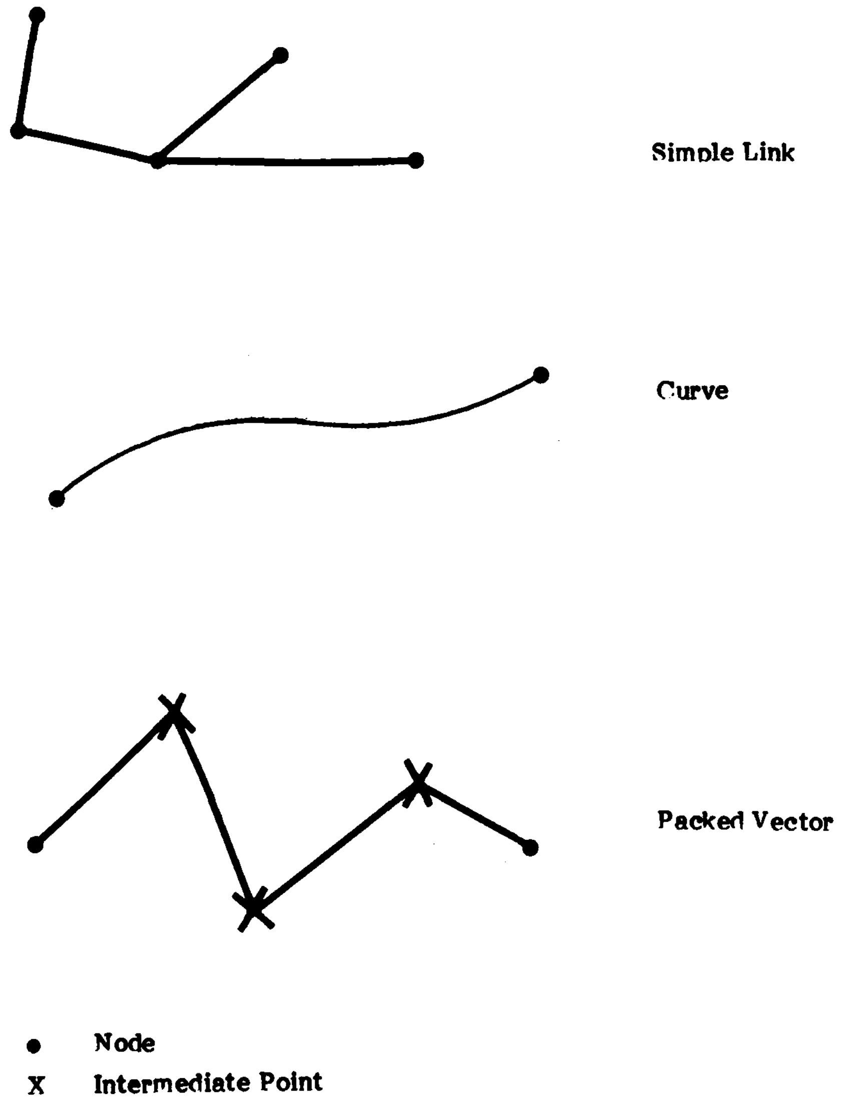


Figure 2.2 Node Connectivity

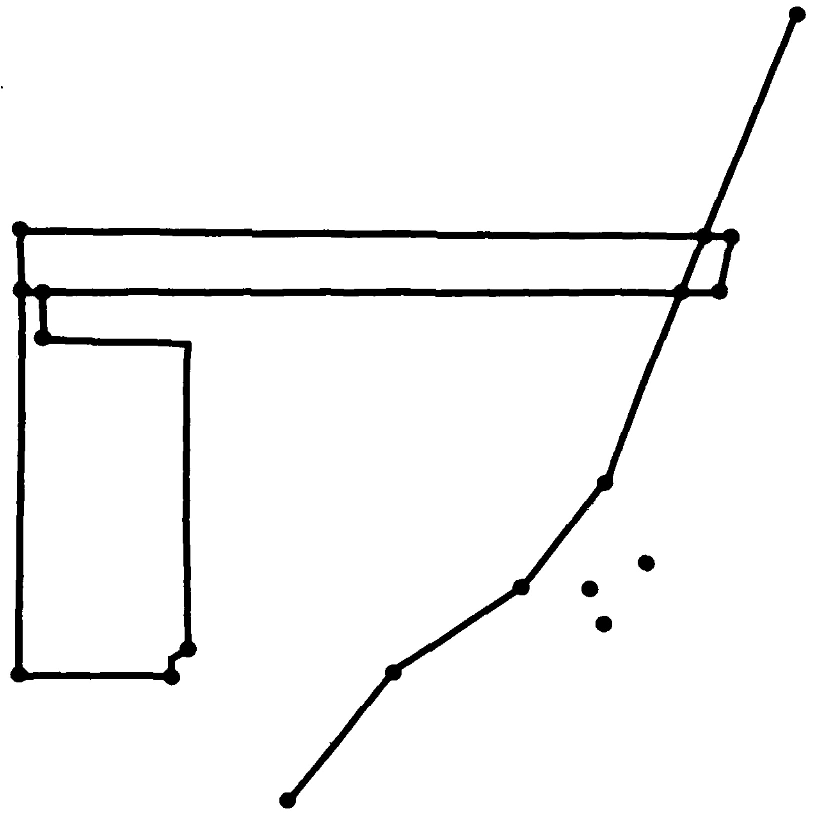


Figure 2.3 Node sets representing map data.

important aspect of the set membership relation is that a given node can belong to more than one set. This means that data of different types can be naturally integrated by using location. Thus, the nodes of a line which has several properties are present in the data structure only once.

2.3 The Addressing Scheme

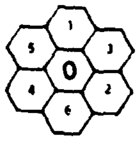
Each node in the data structure has a unique address in a coordinate system called Generalized Balanced Ternary (GBT). We have stated that the nodes correspond to hexagonal areas and to collections of hexagons.

Hexagons have certain advantages over other tessellations or tilings of the plane. The hexagonal covering has the uniform adjacency property, that is, each element of the covering is adjacent to exactly six other elements and shares with each exactly one-sixth of its boundary. In contrast, a covering of the plane with squares does not have uniform adjacencies. Some squares are adjacent to a point while others share a side. It is on the hexagonal cells that the GBT system is built as a hierarchy. At each level, the cells are constructed of cells from the previous level according to a rule of aggregation.

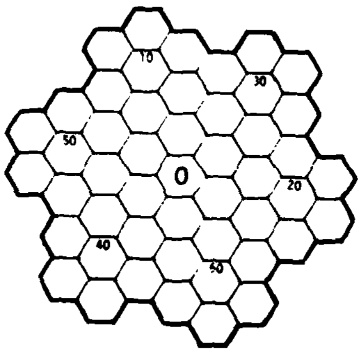
A first-level aggregate is formed by taking a single hexagon and its six neighbors (see Fig. 2.4a). The first level aggregates also cover the plane and have the uniform adjacency property. In general, an aggregate at level n is formed by taking a level n-1 aggregate and its six neighbors. It can be shown that the planar covering and uniform adjacency properties hold at each level. Figures 2.4b and 2.4c show second- and third-level aggregates.

The GBT addressing system is based on the following scheme. In an aggregate, the center cell is labeled 0 and the outer six are labeled, in clockwise order, 1, 3, 2, 6, 4, 5 (see Fig. 2.4a). Each hexagon in the plane has a unique GBT address, a sequence of digits corresponding to the labels of the cells above that hexagon.

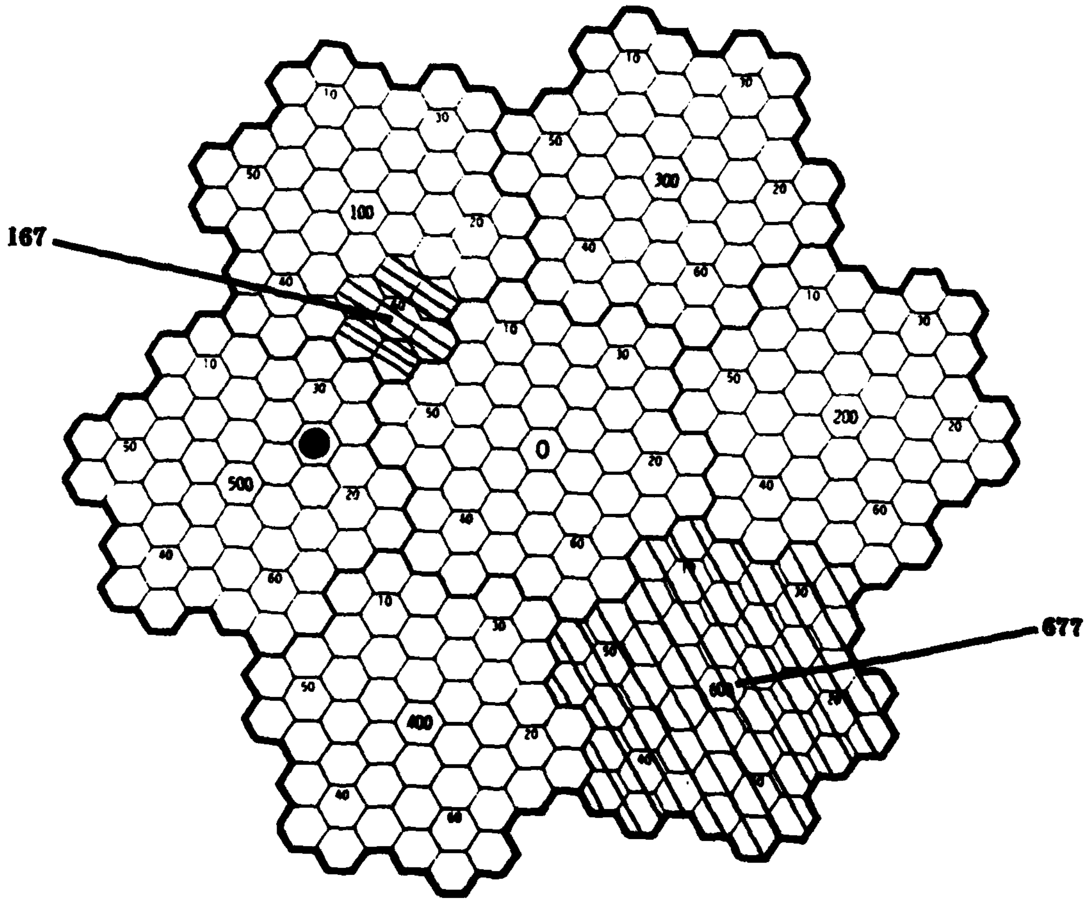
Each digit of the address corresponds to an aggregate level. For example, the address 536 labels the hexagon in the 6 position of the first-level aggregate, which is in the 3 position of the second-level aggregate, which is in the 5 position of the third-level aggregate, which is at the 0 or center position at all higher levels. The hexagon 536 has a dot in it in Figure 2.4c.



a. First Level Aggregate



b. Second Level Aggregate



c. Third Level Aggregate

Figure 2.4 The Aggregate structure.

The digit 7 has a special meaning in a GBT address. It is used to address aggregates rather than individual cells. In figure 2.4c, the first level aggregate 167 and the second level aggregate 677 are shown as shaded regions. By utilizing the digit 7, regions (aggregates) of various sizes can be addressed, which is a useful property in image analysis applications [1].

The GBT system has an arithmetic structure which allows the addresses to be added, subtracted, and multiplied. These operations correspond geometrically to the standard vector operations in the plane. The arithmetic is described in more detail in Appendix B and in [2], [3], and [4].

GBT as implemented in the computer requires 3 bits per digit. The GRAM data manager permits addresses of up to 20 digits to be stored as 8 byte variables. A universe of twenty digits has 7^{20} addressable hexagons. This permits an area equal to the surface of the earth to be filled with hexagons each of a radius of approximately 5 centimeters.

This GBT scheme would be cumbersome if it were necessary to convert to other coordinates in order to perform mathematical operations. The nodal structure used for representing geographical information requires many such operations in processing the data. As implemented, the arithmetic functions are done entirely within the GBT system.

2.4 The Data Tree

The nodes, which are the basic entities of the data structure, and the relations on them exist in the computer as data records in files. The GBT addressing system permits these files to be structured as trees [5]. By this we mean that there is an order relation (a partial ordering) on the nodes. Since each node represents a hexagon or a GBT aggregate of hexagons and has a unique GBT address, the order can be defined simply: "A is higher than B" means that the area represented by B is a subset of the area represented by A. In Figure 2.5, B is the level one aggregate 327 while A is the level two aggregate 377 which contains B. This GBT order results in a tree with one highest or root node representing the universe. Each node in the tree has the potential for seven subordinate branches (see Figure 2.6).

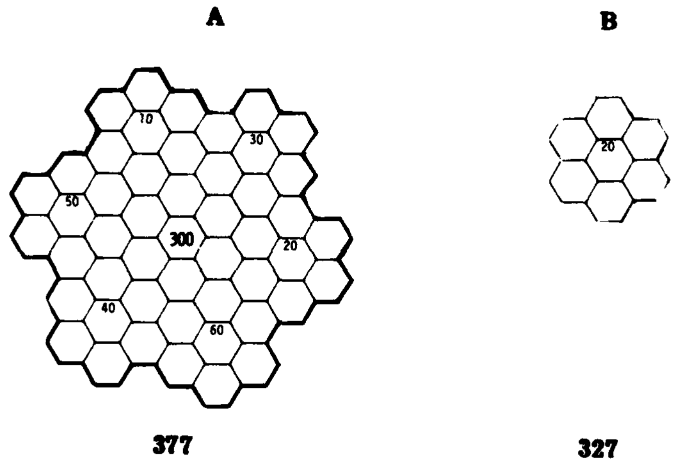


Figure 2.5 The aggregate 377 is higher than 327.

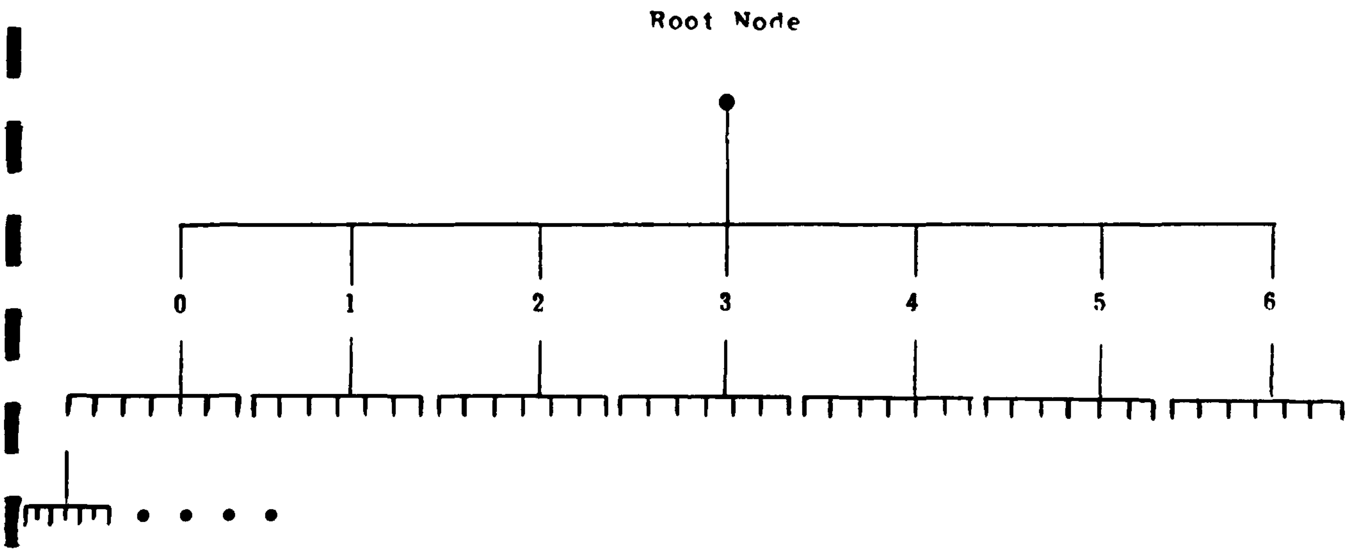


Figure 2.6 The tree generated by a set of GBT addresses.

The tree is implemented in the GRAM data manager in a way that permits it to change with the amount and distribution of data. Initially, a newly created GRAM file has only one tree node in it - the root node. As a program begins to populate the file with data records, the list of records at the root node grows until it reaches a preset maximum length.

A process called "subdividing" is then performed to break the list of records into smaller lists, and then the appropriate tree nodes are created at the next level. The subdividing process occurs whenever the list of records at any given tree node exceeds the maximum allowed.

This technique is efficient in several ways. First, no tree nodes or other structure exist to support areas of space in which there is no data. Second, the refinement or depth of the tree varies as a function of the density of the data in a particular area. In areas where the data is dense, the tree will be full while sparse areas will yield an abbreviated structure. Third, the processes of subdividing and of pruning back the tree take place automatically and continuously as data records are added or deleted.

Finding data records by location with the tree works like this. To find the data record located at address 5133 in the GRAM tree depicted in Figure 2.7, begin at the root node. This has four subordinates, one of which is "5", the highest digit in the address. At the node "5" there are three subordinates, one of which matches the second highest digit "1". Moving to that node, three subordinates are found. The subordinate node "3" corresponds to the third highest digit "3" in the address. This node is a terminal one. Any lower records will be present in a numerically ordered list stored at the location. The address 5133 is on that list. The data record for 5133 has been found in this example by passing through four tree nodes and examining a numerically ordered list. Since the length of this list is a database parameter set by the user, the desired tradeoff between list processing and tree processing can be made.

2.5 Searches

The GBT data tree permits access to the nodes by their addresses, (which correspond to locations in space), in the manner described in the previous section. In order to search an area (either circular or polygonal) for data, a list of aggregates is generated to provide a covering that exceeds the original search area.

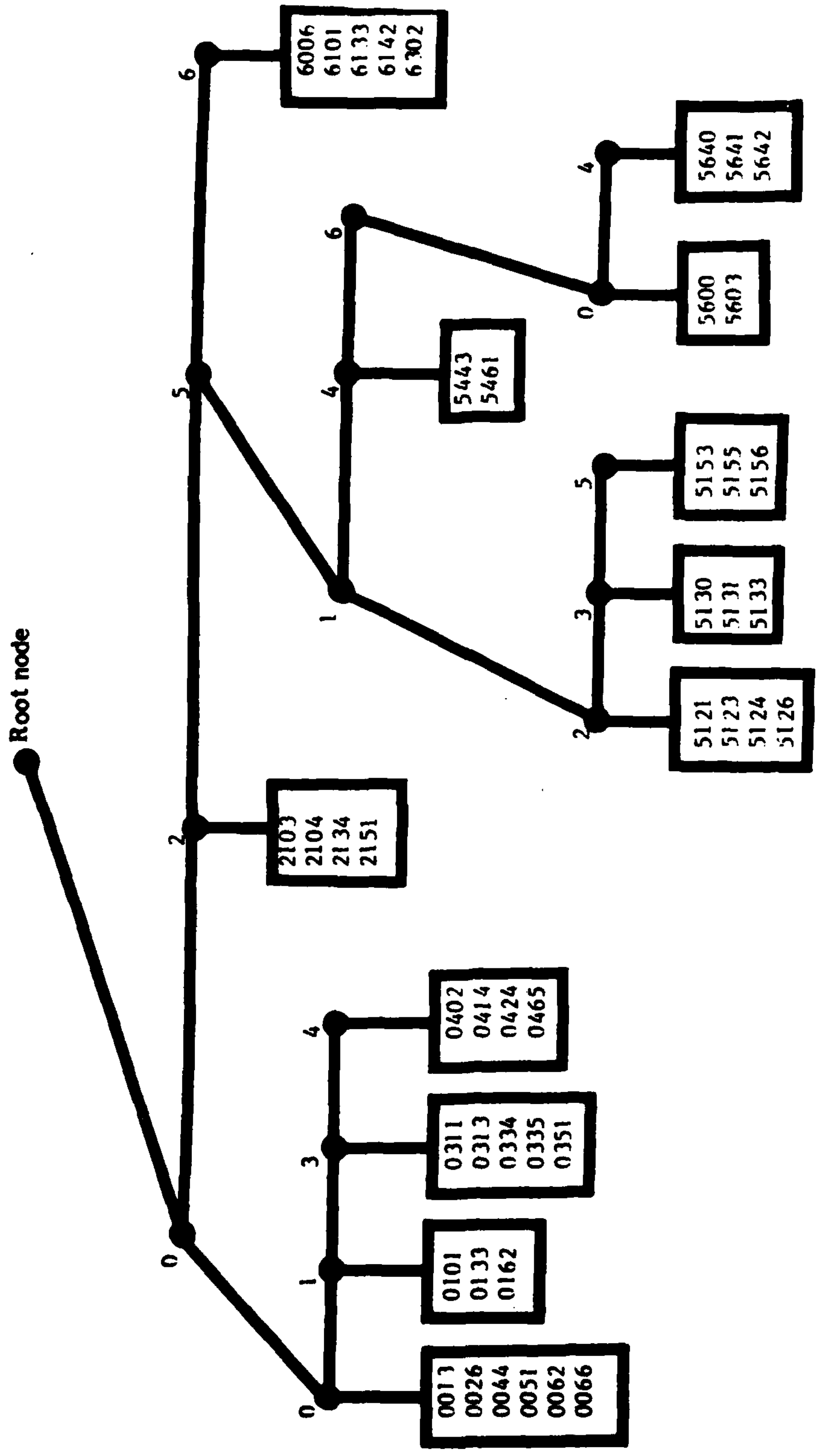


Figure 2.7 A GRAM Tree

The associated tree nodes are then examined for data records and candidate data is checked against the original area. These search-by-location functions are part of the GRAM software. In addition GRAM has some capability for searches on non-locational attributes. Figure 2.8a shows the general form of a tree node data record. A portion of this record, the Boolean field, can be used to store state or characteristics information in individual bits. Figure 2.8b gives an example in which one portion of the field indicates node data type and another portion gives the node degree.

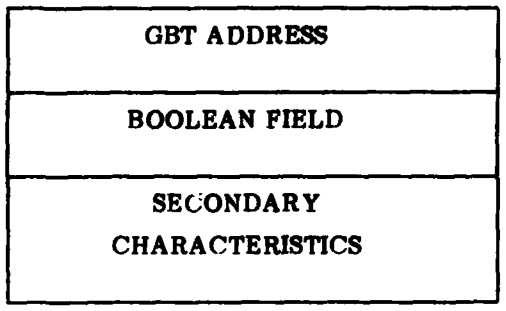
At each node in the tree, a composite Boolean field is automatically maintained. This composite field represents the logical union of the Boolean fields of all the data records that reside in the area represented by that tree node. The root node contains the composite Boolean field of all data records in the file, and lower nodes contain composite Boolean fields of successively smaller areas of the file. A terminating tree node contains the composite field for only the data records that reside at that node. These composite fields are continuously updated by GRAM as data records are added, modified, or deleted.

GRAM is able to retrieve data records based on the Boolean field. Any of a set of relational and logical tests can be applied against specific bytes or words of the Boolean field in the records. The node data records also contain secondary characteristics, such as attribute information in user specified format. GRAM does not perform file searches based on this data.

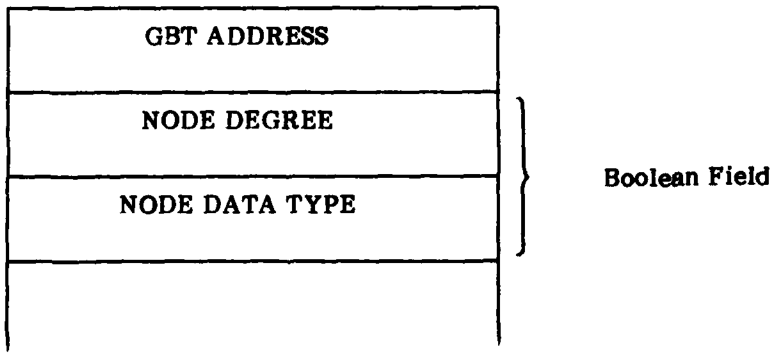
2.6 Attribute Data Management

GRAM itself has limited attribute data handling capabilities. Such information about nodes as their degree and whether they are part of lines or boundaries can be encoded in the GRAM record. The more complicated data associated with geographic entities is more appropriately handled by a relational data manager. The AGIS system uses the Relational Information Manager, RIM. RIM is a commercially available software package with the following basic features:

- Written in FORTRAN
- Interactive (query) interface
- User application FORTRAN interface
- Integer, real, double precision, and text data types
- Moderate memory requirements
- High flexibility
- Highly portable between host computers



2.8a General tree node data record.



2.8b Tree node with Boolean field used for node data type and degree.

2.7 Linking RIM and GRAM

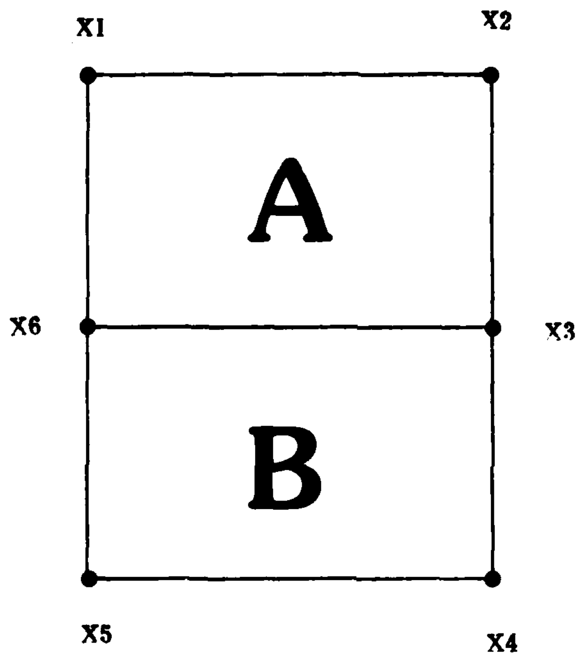
The basic notion in linking the two data managers, RIM and GRAM, is straightforward: RIM records contain GBT addresses associated with their attribute data while tree nodes in GRAM files contain RIM record indentification numbers. This linkage was used to handle the polygon information in the ETL study.

The link is illustrated by the example shown in Figure 2.9, of two polygons, A and B. In RIM, A and B exist as records which contain their various attributes and have integer record numbers, say NA and NB. In GRAM, these polygons exist as their boundary nodes: X1, X2, X3, X4, X5, X6; and the connections between them. Each pair of nodes defines a directed edge, e.g., from X1 to X2 or from X3 to X6. A directed edge has a right and a left side. The edge X3 to X6 has A on its right and B on its left while, for the edge X6 to X3, the reverse is true. This information is encoded in the node records at X3 and X6 using the RIM records IDs NA and NB. In RIM, each polygon record NA and NB contains a pair of GBT addresses which define a directed edge on the boundary of the corresponding polygon. By starting at that edge and taking the rightmost branch at any node of degree higher than two, the boundary of the polygon will be traced. This method of tracing a closed polygon is referred to as the "the right-hand rule" method.

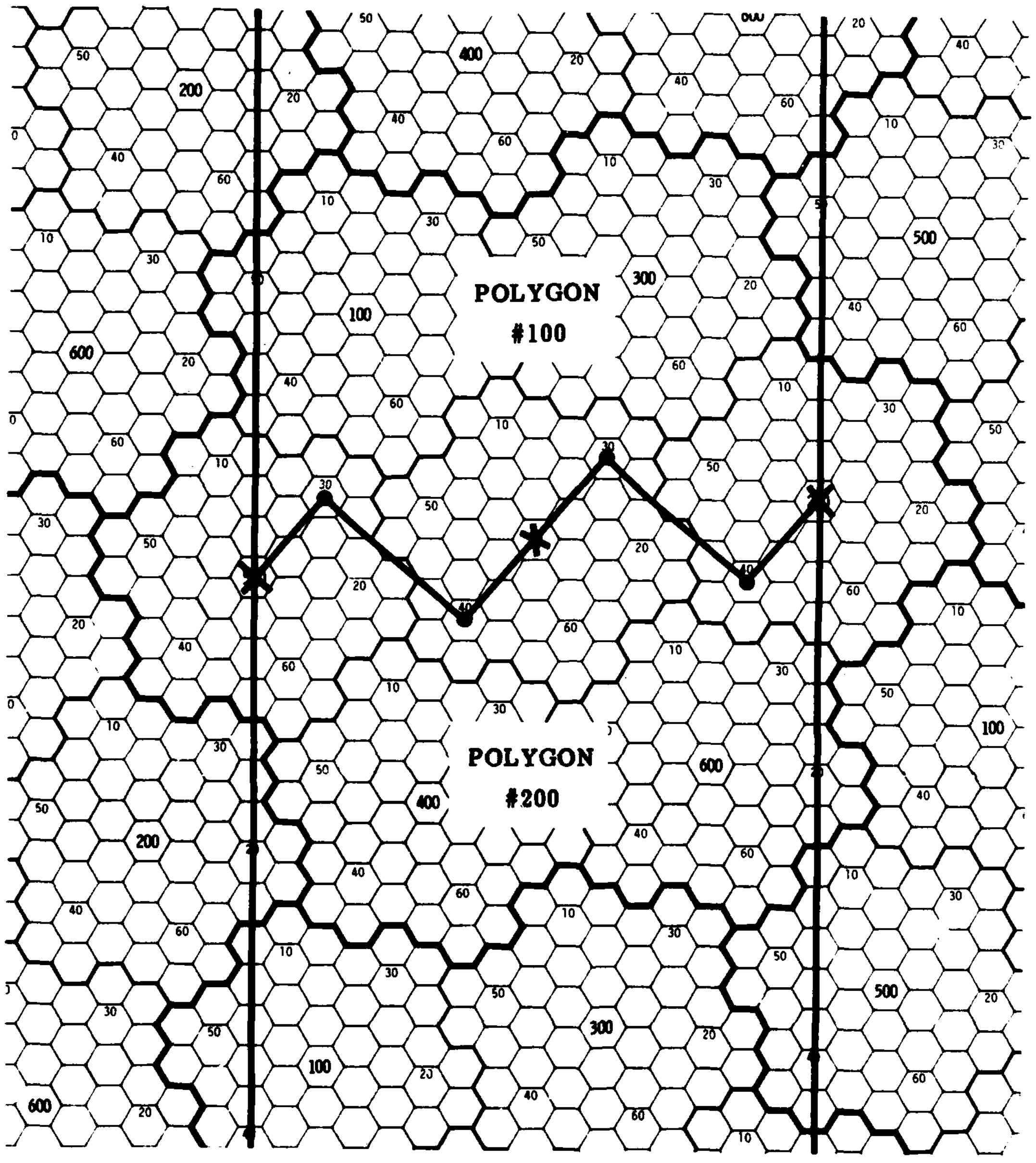
In this way, through the GRAM and RIM record structures, the link between spatial and attribute data is made.

2.8 Record Structures

The AGIS record structure is designed to support map data in the node form. For the ETL study a record form called packed vector was used. This means that a line like that in Figure 2.10 is not represented by a node for each point. Instead, only some points (in this example 500, 000, and 200) are nodes and the rest are stored in lists. Figure 2.11 shows the general AGIS record format for a node and Figure 2.12 shows the record for the point 000 in Figure 2.10. Notice that since the line in that figure is a boundary, the applications characteristics field is used to store polygon identifiers, the RIM keys.



2.9 Two polygons represented by nodes X1 through X6.



● Intermediate Point

X Node

Figure 2.10 An example of a line between two polygons.

NODE TYPE FIELD	Defines the node to be a line, label, or symbol node.
NODE DEGREE	The degree of the node.
LINE TYPE FIELD	Bitmap to define the line node data type.
CONNECTION #1	Address of the first connection.
⋮	
CONNECTION #N	Address of the last connection.
NPV #1	Number of packed vectors between the node and connection #1.
NAC #1	Number of bytes of application characteristics for link between node and connection #1.
PACKED VECTOR LIST #1	Packed vector list containing addresses of points between node and connection #1.
APPLICATION CHARACTERISTICS #1	Application characteristics for link between node and connection #1.
⋮	
NPV #N	Number of packed vectors between the node and connection #N.
NAC #N	Number of bytes of application characteristics for link between node and connection #N.
PACKED VECTOR LIST #N	Packed vector list containing the address of points between node and connection #N.
APPLICATION CHARACTERISTICS #N	Application characteristics for link between node and connection #N.

Figure 2.11 General AGIS record description

1	Implies line node
2	Node degree
1	Line type
500	Connection #1
200	Connection #2
3	3 packed vectors
8	8 bytes of application characteristics
040	Interior address 040
530	Interior address 530
200	Polygon indentifier for left side
100	Polygon indentifier for right side
3	3 Packed Vectors
8	8 bytes of application characteristics
030	Interior address 030
240	Interior address 240
100	Polygon indentifiers for the left side
200	Polygon indentifiers for the right side

Figure 2.12 Sample AGIS Record.

The RIM attribute data base was used to maintain the polygon attribute information for the test data set. A RIM relation is a collection of records, each made up of a tuple. The terms of the tuple are called data elements. In this study, the following four relations were defined on eight data elements:

- PREC - A major relation to store the polygon attributes for every polygon within the database.
- WREC - Essentially the same as relation PREC except it contains only those polygons in the current map work file.
- OLIST - A relation which designates the proper bit setting within the line type field of the AGIS records for each overlay data type.
- IDS - A relation to maintain a bitmap to be used in obtaining a unique polygon identifier.

Figure 2.13 shows the actual definition of the relation in terms of the elements listed in Table 1.

A RIM command requests either all of or some data elements from those records which satisfy a set of conditions called "where clauses". These clauses use the following relational operators:

EQ	-	equal to
NE	-	not equal to
GT	-	greater than
GE	-	greater than or equal to
LT	-	less than
LE	-	less than or equal to

As an example, the relation PREC defined for this study contains 6-tuples. This command:

```
SELECT ALL FROM PREC WHERE OVERLAY EQ LANDUSE AND NAME EQ ACC
```

will result in all complete tuples to be retrieved from relation PREC with overlay element equal LANDUSE and name element equal ACC.

Relation PREC

- Overlay
- ID
- EDGE
- HOLES
- CROSS
- PARENT

Relation WREC

- Same as PREC

Relation OLIST

- OVERLAY
- BITSET

Relation IDS

- IDBITMAP

Figure 2.13 RIM Relatic.ns Definition.

Element-Name	Type	Description
OVERLAY	TEXT	The name of a given overlay type. The following overlay names were used for this study contract: LANDUSE, CONTOUR, and FLOODP.
BITSET	INTEGER	The bit number to set when building GRAM search masks for a given overlay type.
ID	INTEGER	The unique polygon identifier.
EDGE	DOUBLE PRECISION	A directed edge comprised of two GBT addresses. This directed edge is the key from the attribute database to the GRAM spatial database. The two GBT addresses are two adjacent vertices points on the edge of a polygon.
HOLES	INTEGER	A list of island identifiers contained by a particular polygon. A non-zero entry implies that an island polygon is present.
CROSS	INTEGER	A count of the number of crossings that have been found during polygon intersection processing.
IDBITMAP	INTEGER	A bitmap used for the allocation/deallocation of unique polygon indentifiers.
PARENT	INTEGER	The polygon identifier of the parent polygon if the current polygon is an island polygon.

Table 2. RIM record elements.

CHAPTER 3

HOW THE WORK WAS PERFORMED

3.1 WORK ENVIRONMENT

3.1.1 Description of the Hardware

ISC is equipped with two DEC VAXs, several graphics workstations of different types, printers, plotters, and other peripherals. All equipment is off-the-shelf. The hardware used in this study is listed in Table 2.

COMPUTER:	DEC VAX 11/780, VAX 11/750
	- VMS Operating System
	- 2 M bytes of main memory
	- Floating point accelerator
	- 32 bit architecture
DIGITIZER:	SUMMAGRAPHICS ID-48
	- 36" by 48" active area
	- .005" resolution (200 points/inch)
	- 16 button cursor
GRAPHICS TERMINAL:	MEGATEK 7210
	- Calligraphic refresh display
	- 16 K bytes of display list memory
	- 11" by 11" data tablet
A/N TERMINAL:	DEC VT100
HARDCOPY:	VERSATEC V-80
PEN PLOTTER:	HOUSTON INSTRUMENTS CPS-15/6

Table 2. Hardware equipment used in study.

3.2 Input

3.2.1 Map File Creation

A map file was created by the AGIS software to contain all of the spatially oriented data. It was decided to utilize a LAMBERT CONFORMAL projection with standard parallels of 33 and 45 degrees. For every map sheet that was digitized, the operator was prompted to enter in two registration points of 38 35'N, 122 52'30"W and 38 37'30"N, 122 50'W. By knowing the latitude and longitude of each registration point and by their corresponding digitizer x and y coordinates, the proper scaling parameters were calculated to convert each digitized coordinate pair into a GBT address before placing the record into the map file. A GBT universe of ten GBT digits was selected to provide a covering for the map area. Given the latitude and longitude bounds of the map area and the size of the GBT universe, a resolution of .745 meters between adjacent hexagons provided greater accuracy than the digitizer input of 3.077 meters between points.

3.2.2 Manual Digitization

The map file was created, it was loaded by manually digitizing the three hardcopy map products covering the common area of the Healdsburg Quadrangle. These hardcopy maps describe: (1) the land use, (2) the 100-year floodplain, and (3) the elevation contours (see Figures 3.1, 3.2, and 3.3). Each map sheet was registered. Then every x, y coordinate pair was converted into a GBT address and used to form a line node in the AGIS format. Line nodes entered in sequence were connected together to form a line network with each node pointing to its neighbors. The data from each individual input map sheet was entered as one overlay. When all the data had been manually digitized, the map file produced contained three separate overlays.

The operator was provided with basic editing functions with control located on the 13 button digitizing cursor. Therefore, when a line segment reached a point of intersection, the operator instructed the digitizing software to connect the current line segment to the intersection point by simply depressing the connect function button on the 13 button cursor. Figure 3.4 shows a typical situation where the connect function is utilized. Table 3 lists the functions which can be invoked from the digitizer cursor.

Another function button was assigned to provide the capability of starting a line string from an existing line node. Given these two simple editing functions an operator was able to produce a map file with essentially correct connectivity which required minimal map editing time. The AGIS software supports the handling of line data from different overlays within the same file. This is accomplished during file load time. As each new record is added to the GRAM data base, a check is first performed to determine if data already exists at the same GBT address. If it does, the data records are combined to form a node with composite data characteristics. The AGIS mapping software was designed and built to handle line nodes that are either homogenous or non-homogenous with regard to the data type. The operator is provided with visual feedback on the graphics display terminal during the digitizing process. The ability to zoom in on the current work area and produce quick hardcopy allows the operator to input the map data with precision and relatively few errors as evidenced by the digitizing time seen in Table 6.

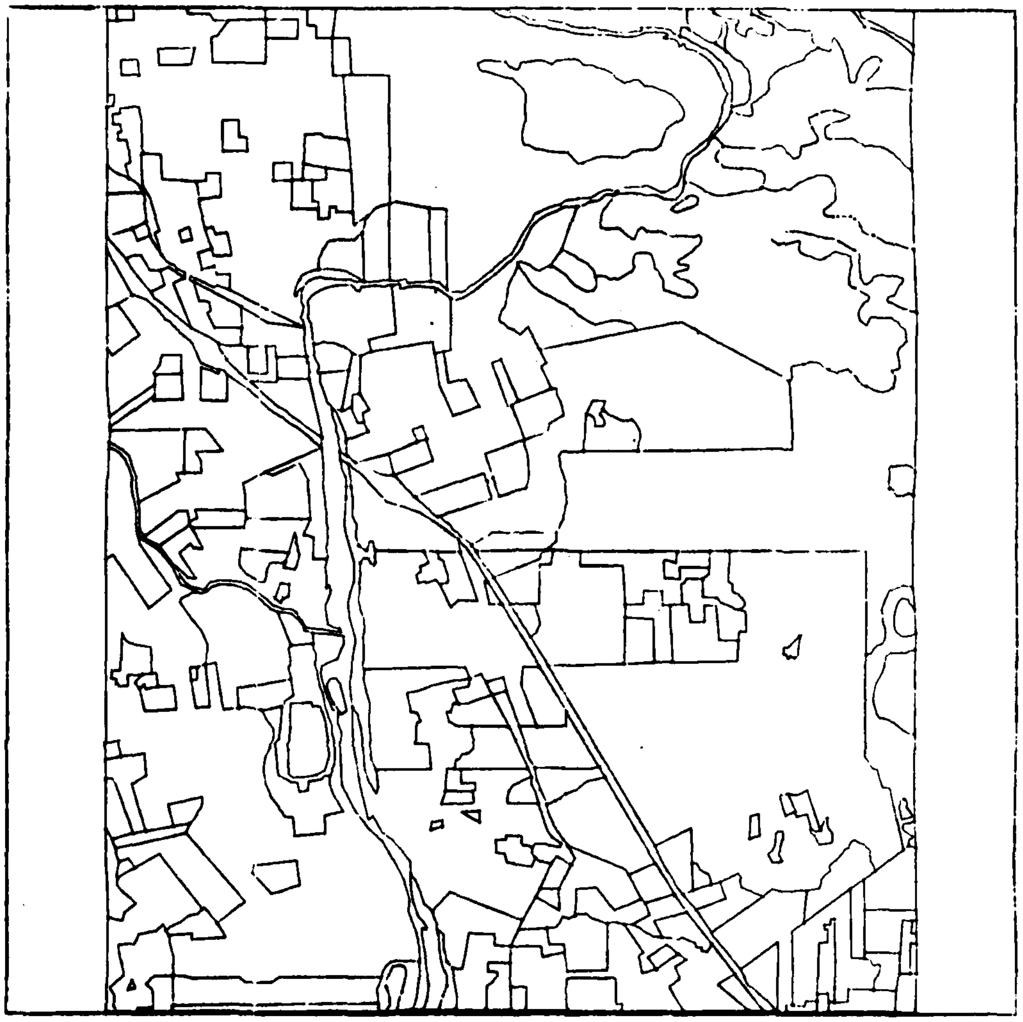


Figure 3.1 Land use map of study area.

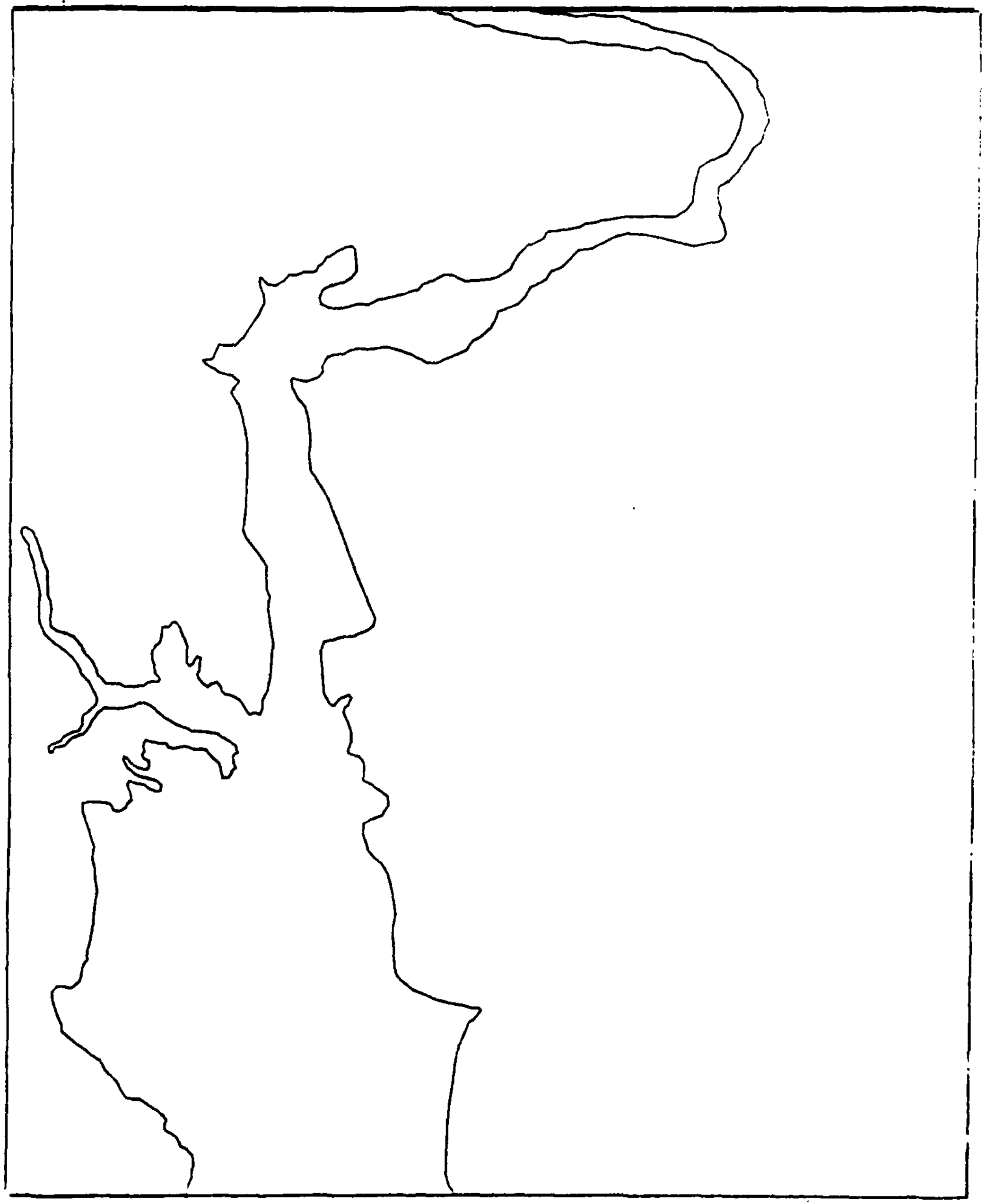


Figure 3.2 100 year floodplain for study area.

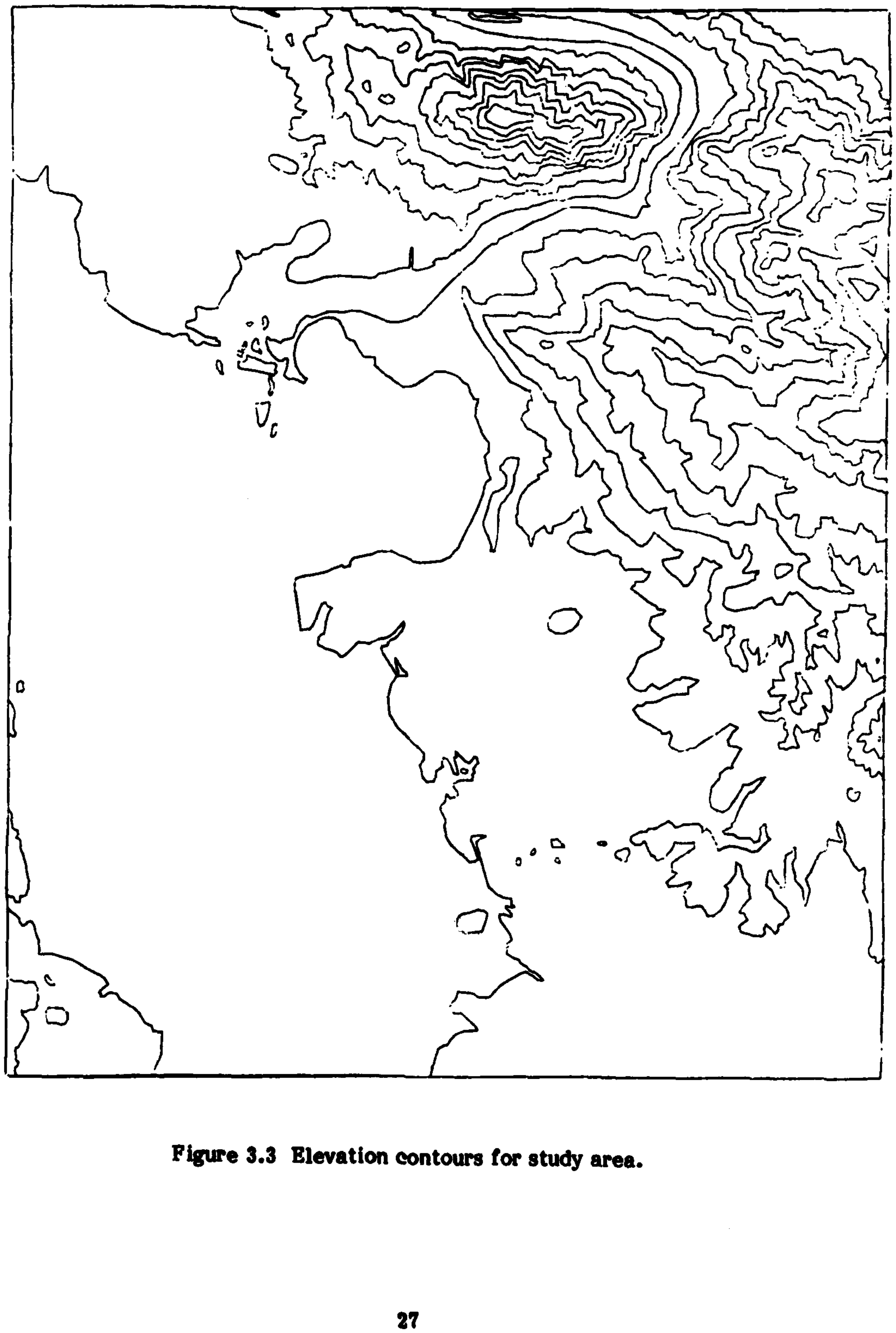


Figure 3.3 Elevation contours for study area.

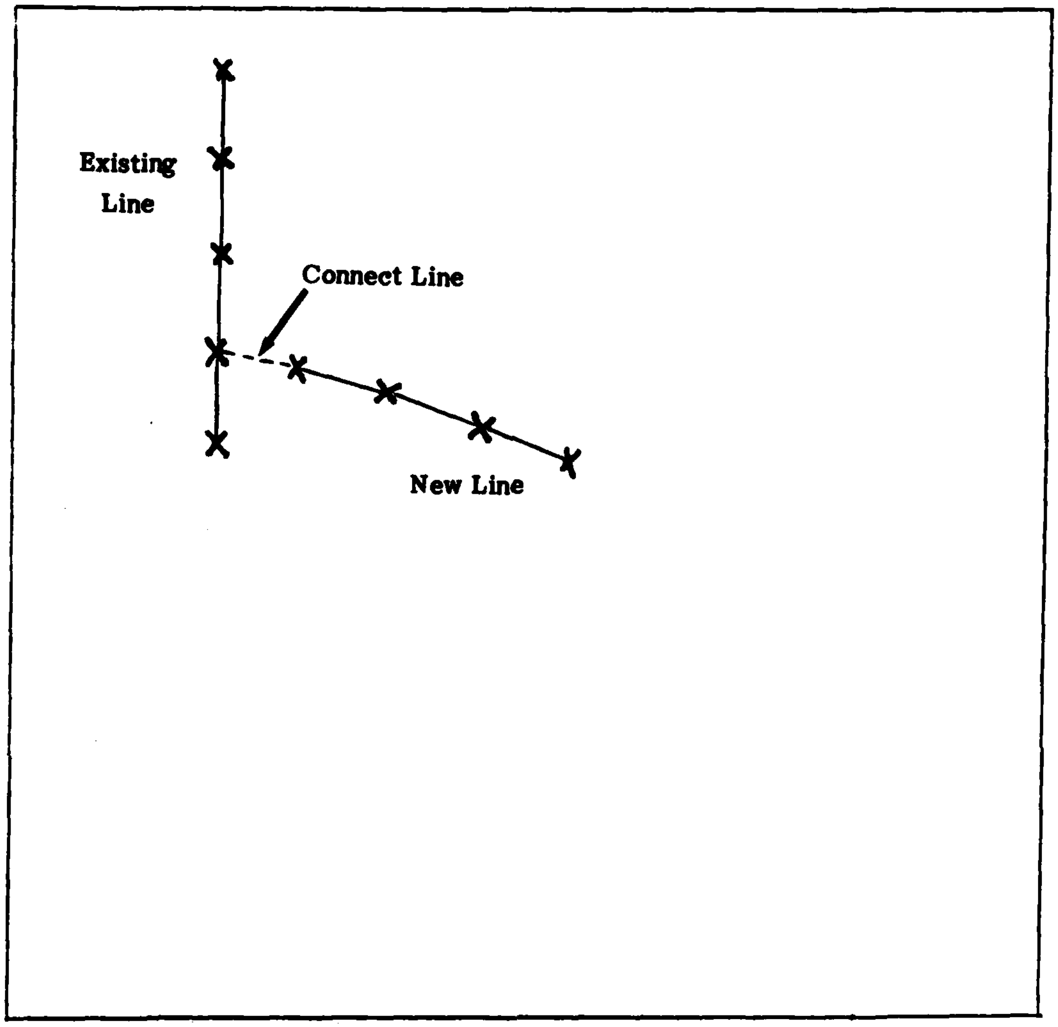


Figure 3.4 "Connect Node" digitizing function.

BUTTON #	FUNCTION
Z	Continue Line
1	Start New Line
2	Start From Existing Node
3	Connect to Existing Node
4	Erase Last Line
5	Redisplay Map
*	Show Cursor Location
0	Set Overlay Type
#	Return Control to Data Tablet

Table 3. Functions Controlled by Digitizer Cursor

3.2.3 Line String Editing

After the basic map file had been loaded through the manual digitizing process, the operator performed map edits using the interactive workstation. Error locations were determined through visual inspection of a hardcopy map and through the use of a line network verification program. The verification program took advantage of two facts concerning data in correct polygonal form. One is that there should be no line nodes of degree one. The other is that each polygonal region should close upon a right-hand tracing rule. Figure 3.5 shows an example of an improperly closed polygon. The algorithm employed by the verification program is to select an arbitrary node and begin tracing the line string until either the same line node is reached (implying polygon closure) or a dead end is reached. Each node is marked processed as the line string is traversed. The process is continued until all line nodes are marked as such. In the event of a bad node or non-closure of a polygon, the x,y coordinate pair is written to a file to indicate the location of the problem area. The operator uses this information file to zoom in on the problem area and use the AGIS editing functions to correct the invalid line string. After all the editing had been performed, the end result was a map file containing the spatially oriented data from all three overlays. The line network verification program has shown that the data was at least topologically correct.

3.2.4 Association of Polygon Attributes

Once the spatial data had been entered, the next step in creating a complete database was to associate the attribute data for each polygon of each data overlay. The method employed to accomplish the attribute association was an interactive method whereby the operator selected an individual polygon for a single data overlay and was prompted to enter the polygon name for the polygon. Once the polygon name was entered, a unique polygon identifier was obtained by reserving a bit location within the polygon identifier bitmap. The unique identifier is simply the bit number relative to the start of the polygon identifier bitmap. For purposes of this study contract, the unique polygon identifier was limited to a range of 1 - 512. The identifier was treated as a signed four byte integer and therefore could take on the range of 1 to 2^{31}. The increased range would require 1 bit to be reserved for every possible number within the range and would therefore increase the file size within the RIM data base. Each record within the RIM database contains a polygon identifier, therefore the identifier bitmap must be allocated one bit for every RIM record. The spatial data describing the topology of the polygon was traced using a right-hand turn decision rule until polygon closure was detected. Every line node visited during the trace process was modified to add the unique polygon identifier to the application characteristics portion of the line node record.

If a non-zero identifier is detected anywhere along the path, an error message is given indicating that an attempt was made to associate a polygon that was already associated. In such an error situation, the operator is given the choice of disassociating the polygon or ignoring the error in the event that the wrong polygon was selected for the association function. The association process is an interactive process. The operator relies heavily upon the visual feedback of having the spatial data displayed on the graphics terminal. The result of the association process is the link from the GRAM records describing the spatial aspects of the polygons to the RIM records describing the polygon attributes.

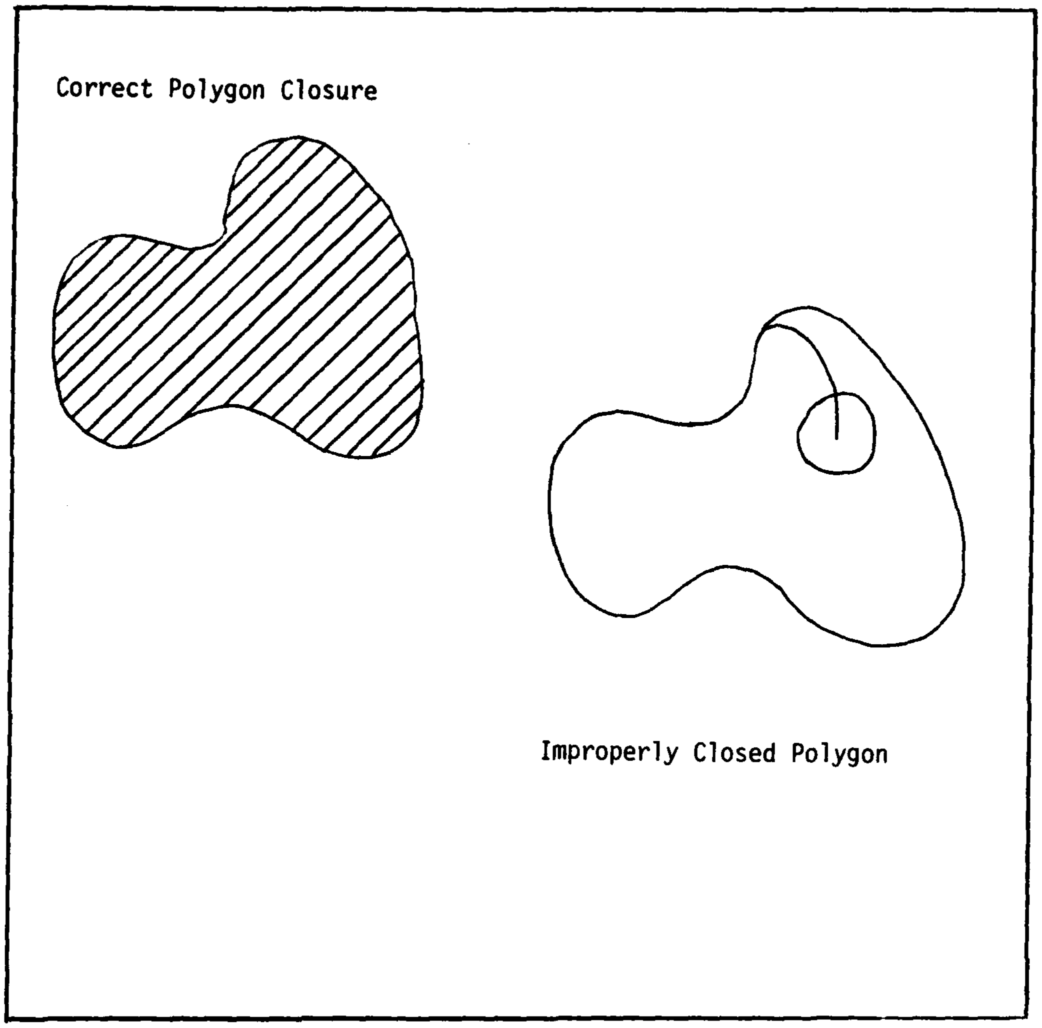


Figure 3.5 Verification detects improperly closed polygons.

3.2.5 Association of Islands

The associating of island polygons with the parent polygon is the final step in establishing the complete database. Figure 3.6 shows an example of a polygon with islands. The actual process involved is one where the operator selects a parent polygon in an interactive mode with visual feedback and then selects an island polygon which is completely surrounded by the parent. Once the two polygons have been selected, the association software modifies the parent RIM record to insert the island polygon identifier into the island list. The island polygon is traced within the map file to place the polygon identifier of the parent into the user application characteristics portion of the AGIS line node records. It is important that the island association process traverses the island polygon in the opposite direction from the normal polygon association process. This is done so that the exterior side of the island polygon line segments will be tagged.

3.2.6 Polygon Island Verification Utility

To verify the final database, a polygon island verification utility was written. This traces every line node within the map file and builds a list of suspect polygon island identifiers where a line segment was determined to have a non-existent or zero identifier on one or both sides of the line segment. Such line segments can only result if the polygon was never associated or if an island polygon was not associated with its parent. If a non-zero suspect list is generated, the operator must zoom in on the suspect polygon and determine the cause of the zero identifier. The problem can be corrected by either associating the polygon or tagging it to its parent. In the case of a zero polygon identifier on both sides of a line segment, the suspect list contains an x,y coordinate pair with which the operator can quickly isolate that problem area. Once the verification utility generates a null suspect list, the operator is assured that all polygons have been associated and that the database is ready for analysis.

3.3 Database Retrieval and Analysis

To perform the analysis tasks required by this study contract, special purpose software was written and incorporated into the standard AGIS mapping system. The philosophy of menu driven software allows the operator to rapidly switch between AGIS editing functions and polygon analysis functions listed in Table 4. Most of the analysis functions work on the principle that the polygon data of interest is copied from the master map file into a temporary work file where the results are either displayed or statistics are compiled for the resultant polygons. The operator can control the selective retrieval by entering phrases which correspond to RIM syntax in response to prompts generated by the analysis functions.

IDENTIFICATION	- The contents of the RIM record are displayed for an individual polygon selected by graphical means.
QUERY	- Display of those polygons within a given overlay satisfying the attributes supplied as input.
INTERSECTION	- Display of the resultant polygons produced via a request involving two or more overlays. The Boolean operators "AND" and "OR" are allowed between overlay expressions.
TABULATION	- Produce a report of the area of each polygon within the current work file.

Table 4. Polygon Analysis Functions

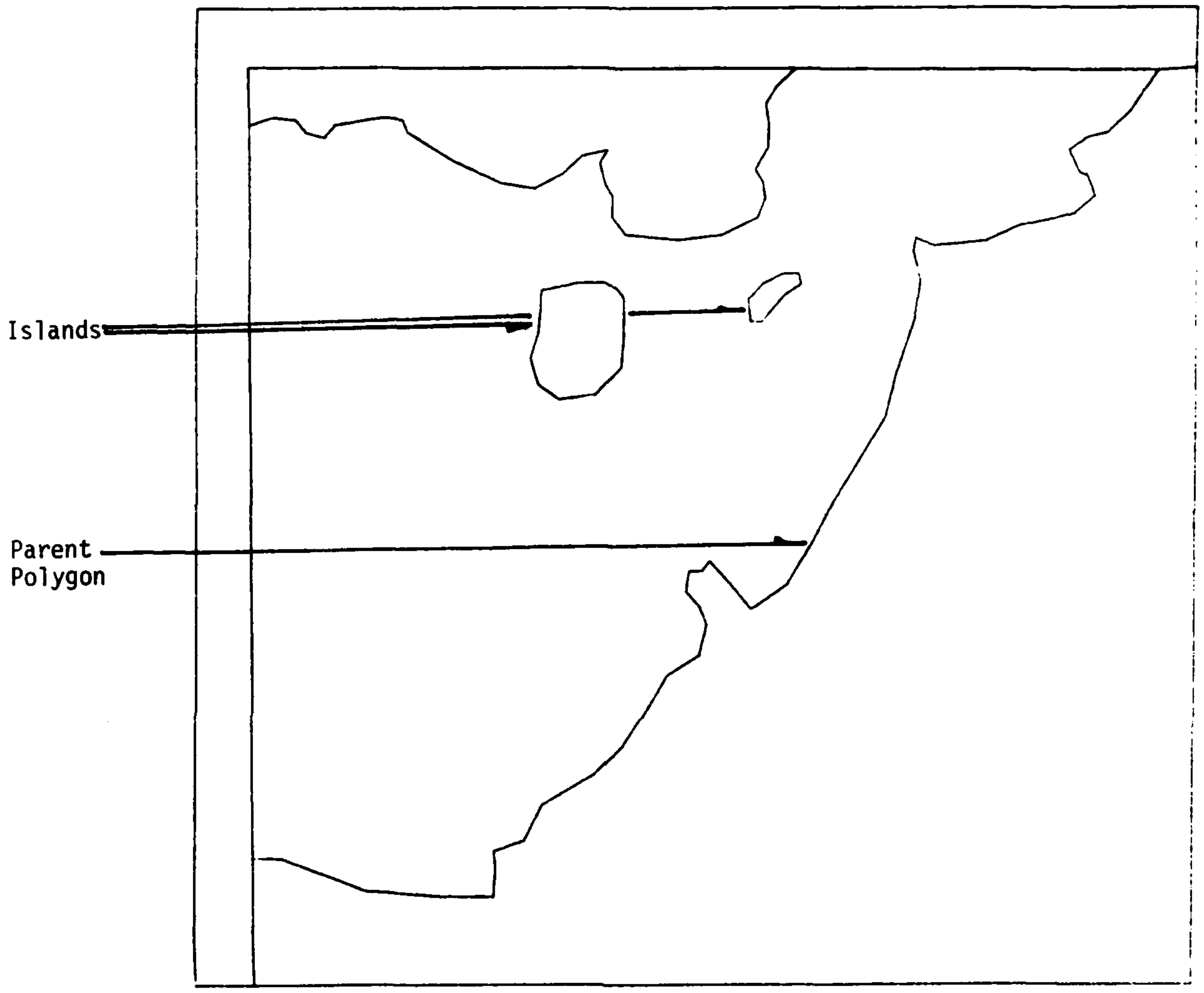


Figure 3.6 Example of a polygon with islands.

3.3.1 Identification Functions

The operator is provided with the capability to display the attributes of an individual polygon. This is accomplished by prompting the operator to select an interior point of a polygon via a data tablet input. A search is performed of the spatial database in an area about the target point looking for the nearest line node within the same overlay as the polygons currently being displayed on the graphics terminal. The seed address of the data tablet hit is either on the left or right side of the line segments which emanate from the nearest line node. The appropriate polygon identifier is extracted from the user application characteristics and used as a key into the RIM attribute database. The RIM record for the polygon is retrieved and the contents of the record are displayed to the operator on the alphanumeric terminal. Figure 3.7 shows an example of the identification function.

In this example, the seed address is used to find the nearest node in data which happens to be the node which is common to the four polygons 200, 100, 543, and 813. The vector from the nearest node to the seed address is used to determine that the line segment separating polygons 100 and 813 is the line segment which is to be used for identifying the polygon 813. The polygon identifier associated with the right side of the line segment (relative from the nearest node) is extracted and used as a key into the RIM data base.

3.3.2 Query Functions

A variety of query and display capabilities are available to the operator in terms of displaying map data. These include display line styles, restoring previously saved areas of interest, changing the display center point and radius, and qualifying the overlay selected for display. Figure 3.8 shows a sample menu containing several display options. In addition to these mapping display functions, a method of display is provided whereby the operator can qualify the polygonal data to be displayed by responding to the prompts with keyboard input in RIM syntax. The RIM attribute database is searched for those polygons which qualify. As each RIM record is retrieved, the directed edge contained within the record is used to initiate a polygon tracing algorithm using data within the spatial database. As the polygons are traced, a copy of the nodal structure is placed in a temporary work file. When the operation has been completed, the resultant polygons are displayed on the graphics terminal. This method of display is only available for one overlay at a time and the attribute records can be qualified with up to five "where" clauses. The following example shows how a display of all land use polygons with code equal to ACC would be initiated:

```
OVERLAY NAME: LANDUSE
POLYGON NAME: ACC
```

Figure 3.9 shows the results of such a request.

3.3.3 Polygon Intersection

Special purpose software was written to provide the ability to produce a data set in response to a request for selective retrieval from the database. The form of the request is such that an operator can perform a selection of a given overlay with up to five different conditions being true before the data is qualified. The Boolean operators of AND and OR can be used between the conditions with an implied left to right expression evaluation.

The Boolean operators of AND and OR (intersection and union, respectively) can be utilized between individual overlay selections to produce a compound Boolean expression between different overlays. The NOT operator is not specifically allowed between overlay expressions. The equivalent functionality of the NOT operator can be accomplished within the overlay expression itself.

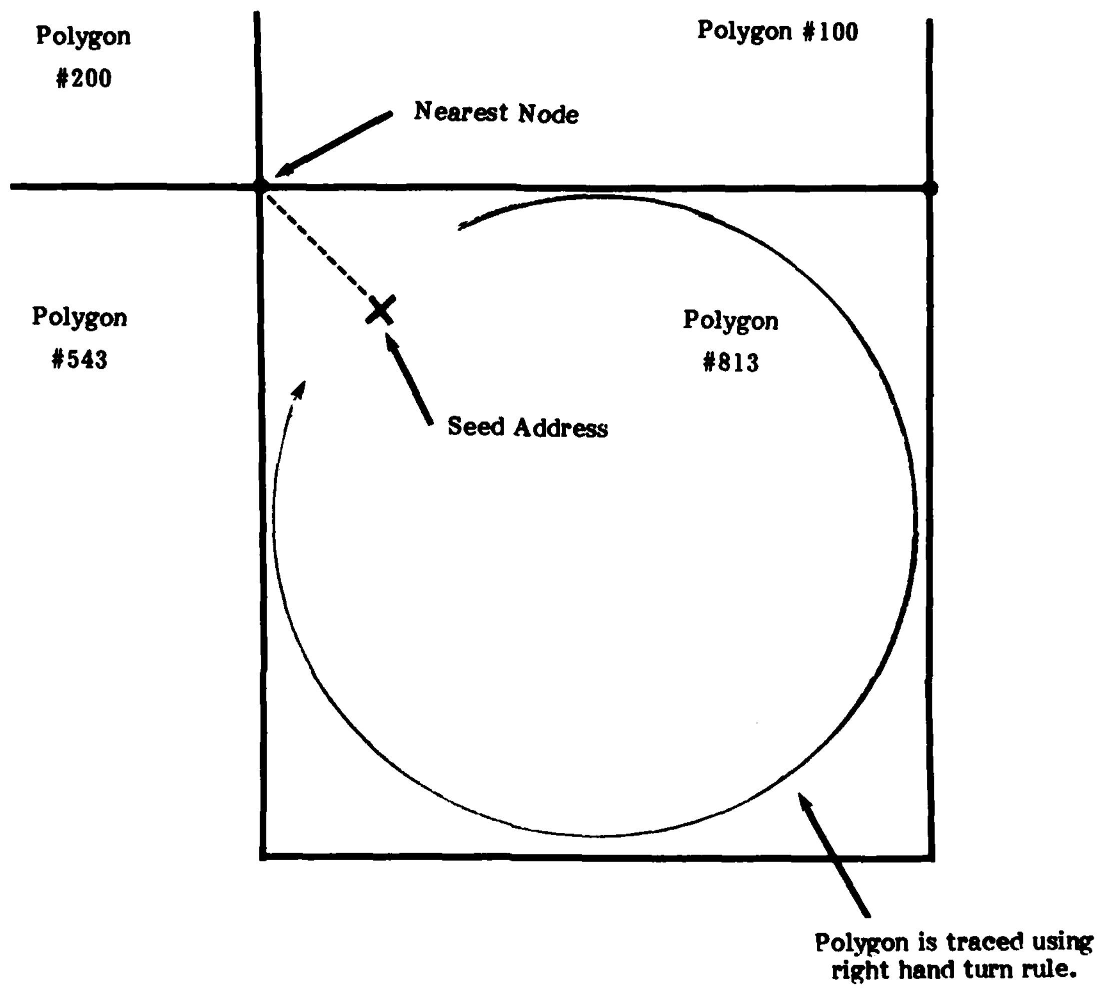


Figure 3.7 Example of identification function.

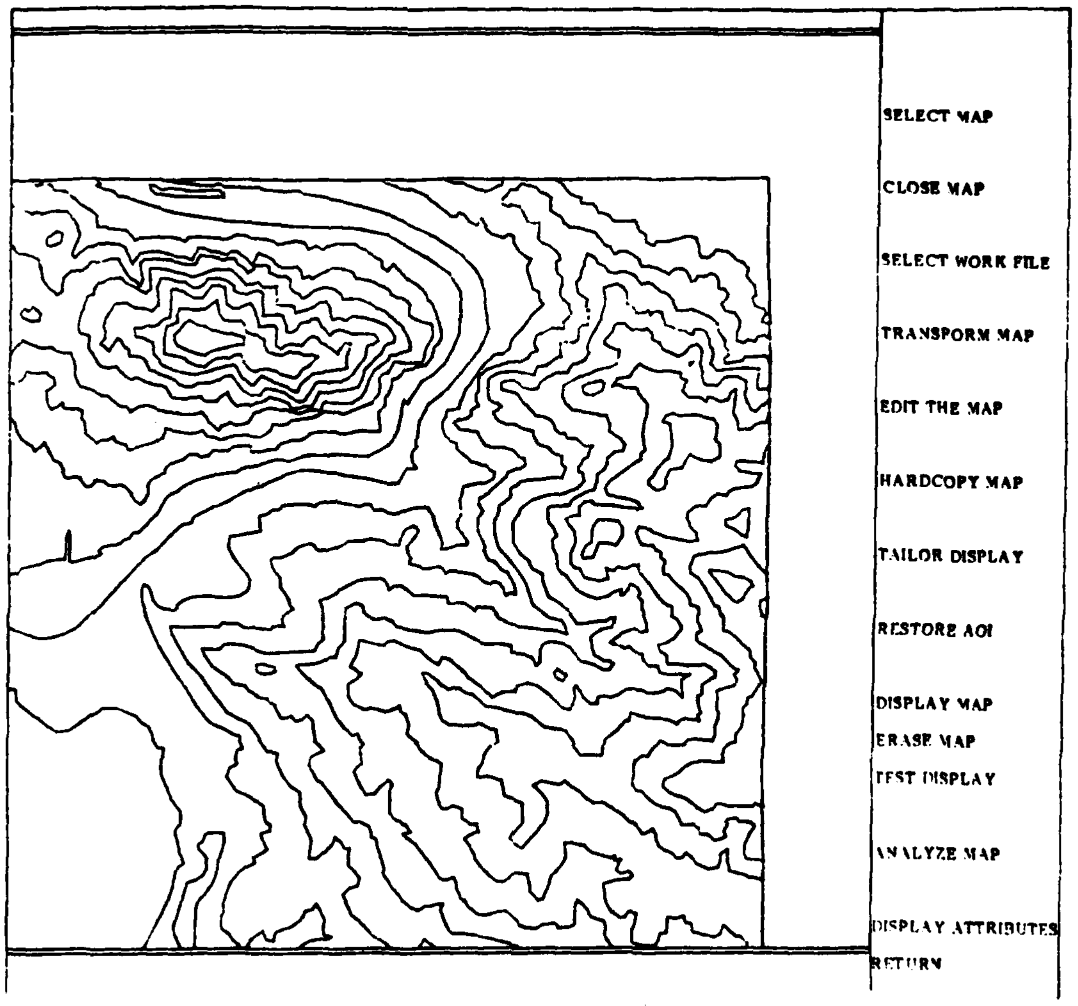


Figure 3.8 A menu tree.

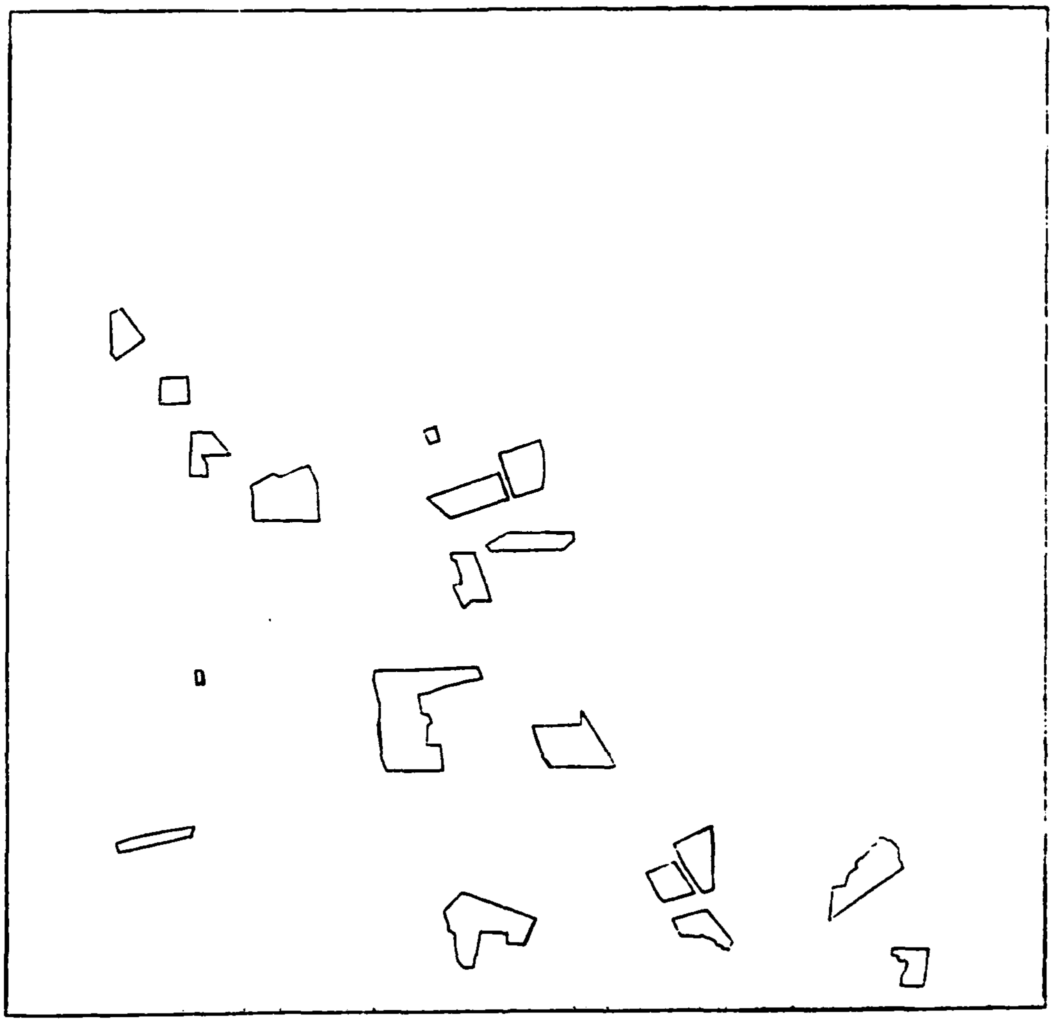


Figure 3.9 Land use areas with code ACC.

The qualifying expression as entered by the operator is parsed and broken up into commands sent to RIM for a single overlay at a time. Therefore, a compound Boolean expression is broken up into a series of binary operations. The following is an example of a compound Boolean expression used to qualify the data:

OVERLAY NAME:	LANDUSE
POLYGON NAME:	
ANDOR:	AND
OVERLAY NAME:	FLOODP
POLYGON NAME:	EQ INSIDE
ANDOR:	AND
OVERLAY NAME:	CONTOUR
POLYGON NAME:	LE "100"
ANDOR:	

Figure 3.10 shows the results of such a request. The actual algorithm to compute the resultant polygons from a compound Boolean expression is described in the following steps:

1) Individual polygons qualified by a single overlay expression are copied from the master map file into a temporary work file. After all the polygons have been copied, a merge operation is performed to remove the duplicate line segments that were produced as a result of copying adjacent polygons on an individual basis.

2) Points of intersection between polygons of different overlays are located. This is done by searching the entire work file for nodes of a given overlay and as each line node is retrieved, a small area search is performed about that node looking for line nodes of the other overlay. If any nodes are detected in the proximity search, then the line segments involved are individually checked to determine if there is any intersection. If a point of intersection is found, then a special intersection record is written to the work file which includes enough information to determine which line segments were involved to produce the intersection point.

3) A search of the entire work file is initiated looking for all the intersection records. As each intersection record is retrieved, a new line node is formed by adding line segments from the intersection point out to the nodes which were involved in producing the intersection record. The new node is also tagged with enough information to determine which branches are to be kept as part of the resultant polygon. It is at this point in time that the Boolean operator between overlay expressions plays an important role. The branches of the intersection node to be kept or thrown away depend on whether a polygon intersection or union operation is currently being performed. The new intersection node is added to the work file.

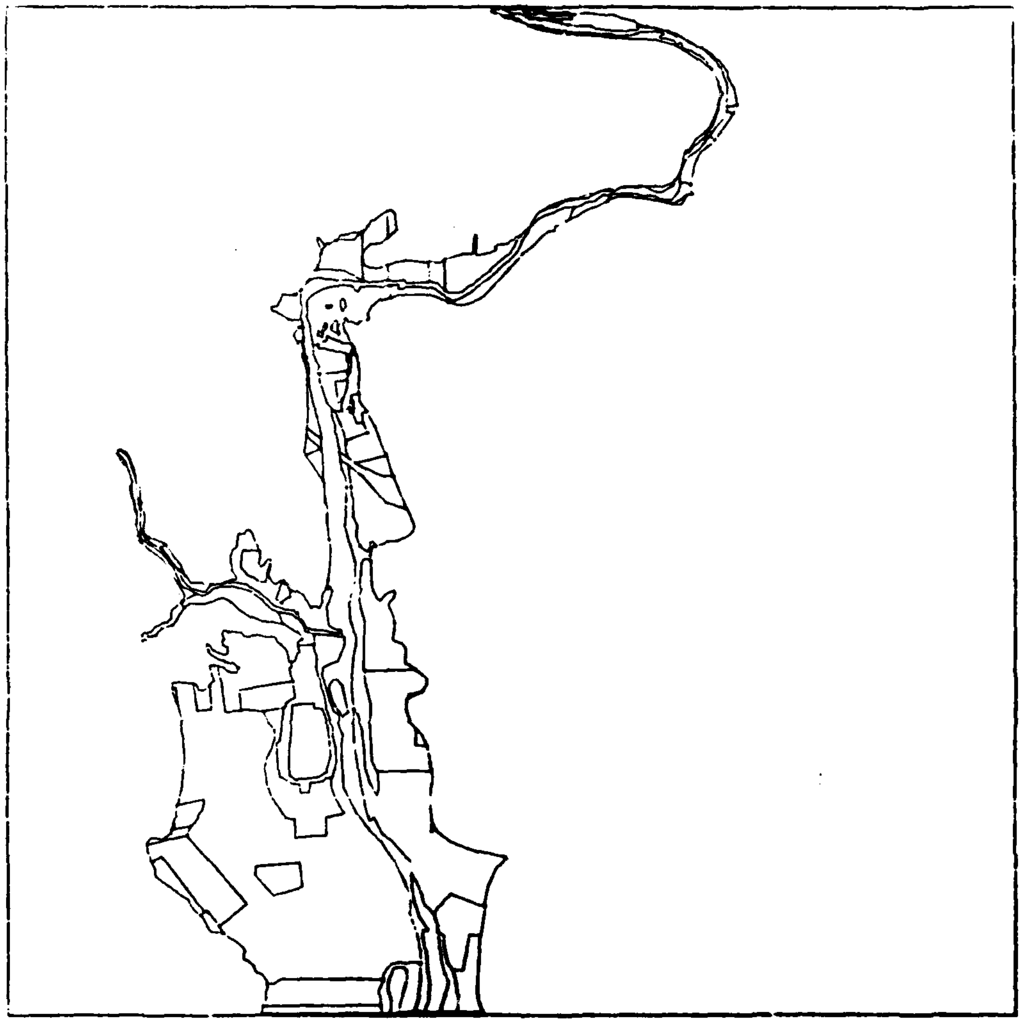


Figure 3.10 All landuse areas within the floodplain and below 100 feet.

4) The intersection nodes are then used as seed points from which the polygons are traced. A unique polygon identifier is obtained and as each resultant polygon is traced, two modifications are made to the line nodes. The first is the replacement of the polygon identifier. The second is to change the overlay bit encoding to signify that the line nodes are now the result of a binary operation (either intersection or union) instead of their original encoding (either LANDUSE, CONTOUR, or FLOODPLAIN). As each resultant polygon is closed, a RIM record is created by extracting information from the RIM records of the polygons involved in the intersection. This record is then added to the RIM database as part of the work relation.

5) The final step is to check for the case in which a polygon within the first overlay completely surrounds a polygon within the second overlay. The process used to perform this check is to obtain a line node from any of the polygons within the second overlay which was not involved in any previous intersections. A search is preformed around this node looking for the nearest line node in overlay one. A simple check can then be made to determine if the node in overlay two is located inside the line segments which emanate from the overlay one node. If the overlay two node is inside, then the polygon which it is a part is traced to mark the line nodes to be kept as part of the resultant data set.

3.3.4 Tabulation Function

A tabulation function was built to compute the area in acres of all resultant polygons found in a temporary work file. The work file is produced by using one of the functions described above. Every RIM record from the work relation is extracted in a sequential manner. The directed edge contained within each record is used as a key in the temporary map file and each polygon is traced to compute the area in square meters. In the case of polygons which contain islands, the area of the entire polygon is calculated and then each island is traced and the island area is subtracted from the parent polygon. An output routine sorts the polygon names in alphabetical order and converts the square meter measurement into an acre equivalent.

3.3.5 Hardcopy Output

A stand-alone program was written to produce hardcopy output at a variety of scales including the same scale as the input data. The inputs to the program are: 1) the map file name, 2) the name of a previously stored area of interest, and 3) the desired number of inches for the longest axis (x or y). The plotting program then traces the map data. As the map data is traced, the output is directed to the appropriate hardcopy device.

3.3.6 Map Sheet Merge Function

Data which is digitized from separate input map sheets will not line up exactly on the common edge. Portions of the data may overlap or contain gaps between nodes that should connect. Small artifacts are almost always present due to the fact that the operator must re-register the map sheet for every digitizing session. For this reason, a special purpose function was built to perform the automatic merging of polygons which lie on adjacent map sheets. The map sheet border is an artificial boundary and the true polygon is the composite of the polygons on each side of the border. The approach taken to perform map sheet merging begins with tagging the line segment that represents the border as being a border segment. This implies that the line segments along the border will be tagged border segments in addition to being tagged with the overlay of the polygons which they define. There are four different border data types (TOP, BOTTOM, LEFT, and RIGHT). This is done so that adjacent map sheets would have complementary border data types along their common edges. The border line of an adjacent map sheet is tagged with complement border data type. The algorithm to merge polygons then searches the spatial database for those line nodes with the border data type. As each node is retrieved, a search is performed looking for the nearest line node with the complement border data type. These line nodes are connected and replaced in the database. After all the nodes have been connected, all line segments with a border data type are deleted. This removes the artifical border between map sheets. The final step is to search each polygon that previously lay along the border and to remove the RIM records for it. Each of these polygons is then retraced to add a new RIM record which represents the composite of the polygons which were on opposing sides of a map sheet boundary. At this time, the algorithm as implemented does not check to determine if the polygon on one side of the border has the same name as the polygon on the on the other side. A simple enhancement to the software would be to flag those areas when the polygons do not match across a common border.

CHAPTER 4

RESULTS AND LESSONS LEARNED

4.1 What Was Learned

The work performed under this study can be analyzed at three levels. The most basic is the AGIS software, including the GRAM hexagonal data manager, which was the foundation for the polygon software. At the next level is the polygon handling concept and the software that was developed for this study. Finally there are the statistics which resulted when the system was exercised on the Healdsburg data set.

4.1.1 AGIS

Generally the AGIS software performed well in handling the requirements of this study. AGIS data entry and editing features made the database construction straight forward. The one area of concern within AGIS was the strong interdependence between GRAM and RIM that was required to handle polygons. This caused minor problems in database integrity.

Both data managers have recovery capability. However, they do not recover in parallel. After a power interruption during digitizing halted the system, the recovery operation left GRAM records pointing to nonexistant RIM records. Enlightened editing was required to make things right.

A related problem can occur when editing. The existing AGIS editor allows independent editing of GRAM and RIM records. Therefore it is the responsibility of the person performing the edit to make sure that both data managers are informed of changes. As an example, suppose it is necessary to remove the common boundary between two adjacent polygons. Simply erasing that boundary segment from the GRAM file does not complete the process. The RIM records for the two polygons must also be accessed and merged.

Aside from these problems, the AGIS software performed perfectly in support of this study.

4.1.2 Polygons

During the course of the study a concept for handling polygonal data was developed and implemented. Details of that concept were presented earlier in this report. Two aspects of the concept appeared to have been less than optimal. One was the trade off between batch and real-time computation of intersections. The other was the handling of islands.

The concept that was followed called for the different overlays to be stored completely independently, both in GRAM and RIM. To respond to any given query involving multiple overlays, the software would assemble all needed data and compute from scratch whatever was required. It now seems that this process involved more time than can be tolerated in real-time operations. Restructuring of the software developed for this study contract would improve the results substantially. However, changes in the underlying concepts are required to make the polygon processing truly interactive. Intersecting all polygons in all overlays and storing the resulting pieces with multiple attributes would lead to the creation of a massive data set which will be very infrequently used. Our feeling is that a good intermediate position is to compute polygon intersection points in batch mode and store them in GRAM while leaving the RIM records unmodified. This would result in additional data but would avoid some of the most time consuming aspects of responding to real-time queries.

Under the concept used in this study, the presence of islands within a polygon was stored in the RIM record for that polygon, and was not directly observable from any data in GRAM. One consequence of this was that islands always presented a special case that required separate logic and code. Another is that RIM's slowness relative to GRAM made the processing of islands time consumming. It now seems that the concept would have been superior if the relation of an island to its surrounding polygon had been directly present in GRAM. Such a change, if made, should result in simpler code and faster execution.

4.1.3 Results

The tabular listings required by the Statement of Work are provided in Appendix A of this report. The graphic products required are included in Appendix C under separate cover.

Table 5 describes the data element breakdown and the file sizes for the database. The number of line nodes and the file sizes are shown to be significantly reduced for the packed node format. Table 6 shows the elapsed operator time taken to perform each major process. It should be noted that the operator time to manually input the data is extremely low for two reasons. The first is that no data preparation was necessary. The input map sheets were placed on the digitizing table, registered, and digitized. The second is that editing functions were provided to the operator at the digitizing station. At the end of the digitizing session, the spatial data was practically error free. The edit times were very small because of the quality of the input from the digitizing software. Several edit functions in an interactive graphics environment allowed the operator to quickly perform the necessary modifications. The times listed for display and analysis of selected queries to the database are larger than they should be. As stated above, an attempt was made to produce on-line intersections of all polygons of interest between overlay data types.

Figure 4.1 shows a small portion of the landuse overlay that was utilitized for testing the polygon merge operation. During the input of the two tést areas, a map registration was performed for each side. Significant map edge overlap was incurred deliberately in order to highlight the problem of adjacent map sheet merging. Figure 4.2 shows the result of the automatic polygon merge function. It can be seen that several artifacts are produced using the merge function with its simple connection rule.

The quality of the connections could be improved by examining more than just the connection nodes during the merge process. By utilizing the line segments leading up to connection nodes, a less severe connection between map sheets would result.

Figure 4.3 shows the area involved in the landuse revisions. The procedure to perform the land use revision was to disassociate the original polygons, perform the necessary graphical edits, and then reassociate the resultant polygons. Figure 6.4 shows the resultant polygons.

DATABASE STORAGE REQUIREMENTS

	# OF POLYGONS	# OF UNPACKED NODES	# OF PACKED NODES
LANDUSE	223	3223	1245
ELEVATION	75	5427	1155
FLOODPLAIN	4	660	257

FILE SIZES FOR THE DATABASE

	UNPACKED	PACKED
SPATIAL DATABASE	879 BLOCKS *	319 BLOCKS
ATTRIBUTE DATABASE	190 BLOCKS	190 BLOCKS

* 1 BLOCK EQUALS 512 BYTES

Table 5. Database storage and file sizes.

	CPU TIME	MEMORY (512 BYTE PAGES)	OPERATOR TIME
DATABASE CREATION AND DIGITIZING			
- Landuse	17.9 min	1409 (peak)	3.5 hour
- Elevation	34.8 min	1240 (peak)	3.3 hour
- Floodplain	2.5 min	1124 (peak)	.5 hour
NODE EDITING	4.8 min	1153 (peak)	.5 hour
ASSOCIATING POLYGON ATTRIBUTES	46.5 min	1536 (peak)	2.4 hour
ASSOCIATING ISLANDS	15.0 min	1523 (peak)	.9 hour
VERIFICATION AND FINAL EDITING	8.5 min	1536 (peak)	.8 hour
DISPLAY AND ANALYSIS PER QUERY			
- EASY	1.75 min	1024 (peak)	3 min
- AVERAGE	15 min	1024 (peak)	25 min
- DIFFICULT	45 min	1024 (peak)	1.5 hour

Table 6. Execution time for all major processes.

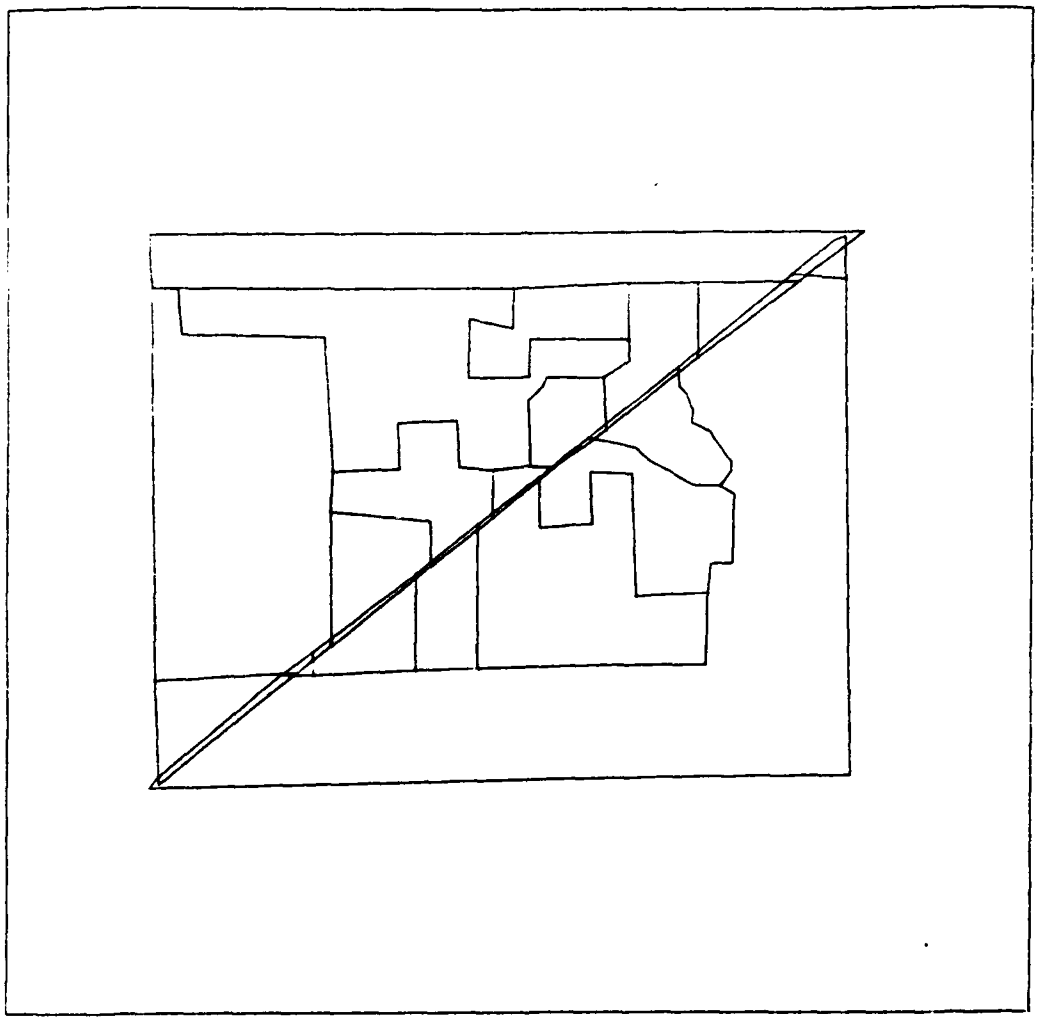


Figure 4.1 Merge function test area.

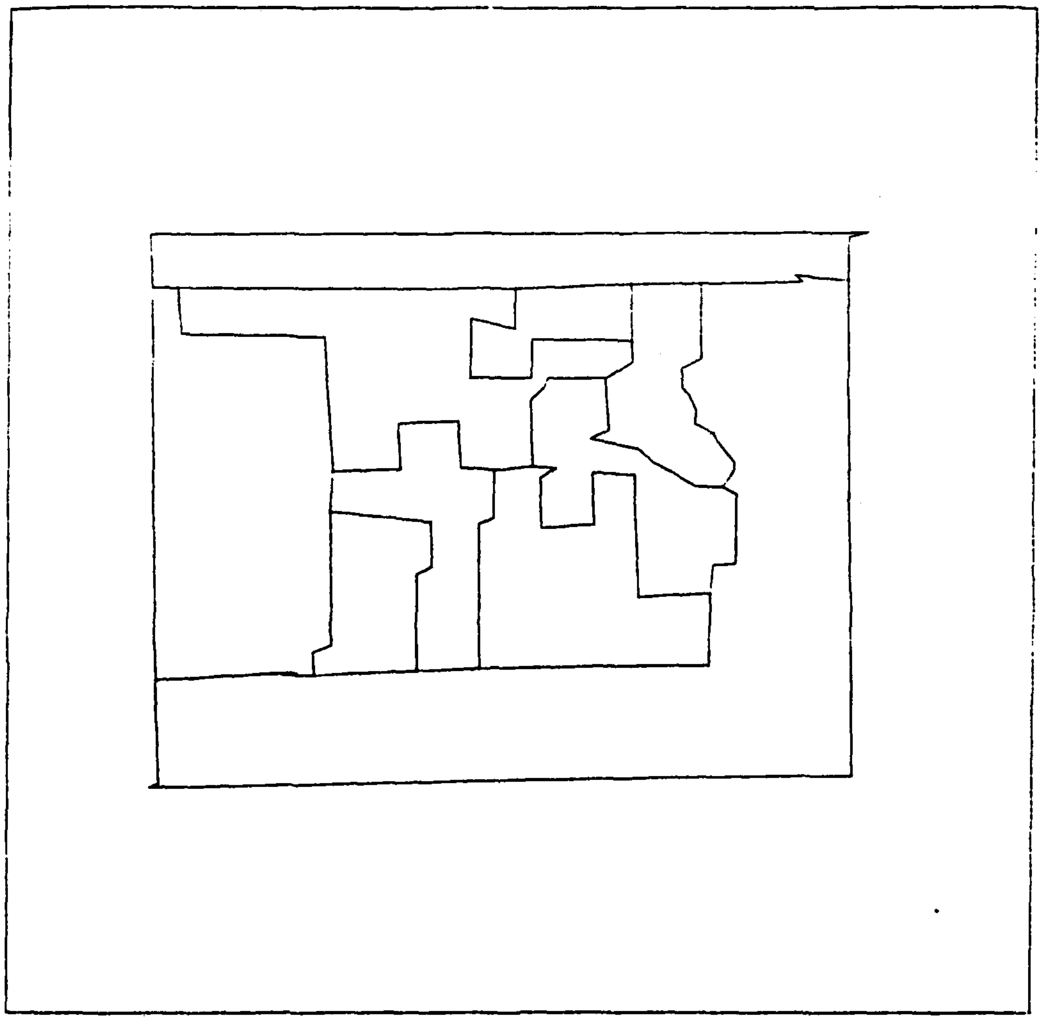


Figure 4.2 The results of a merge operation.

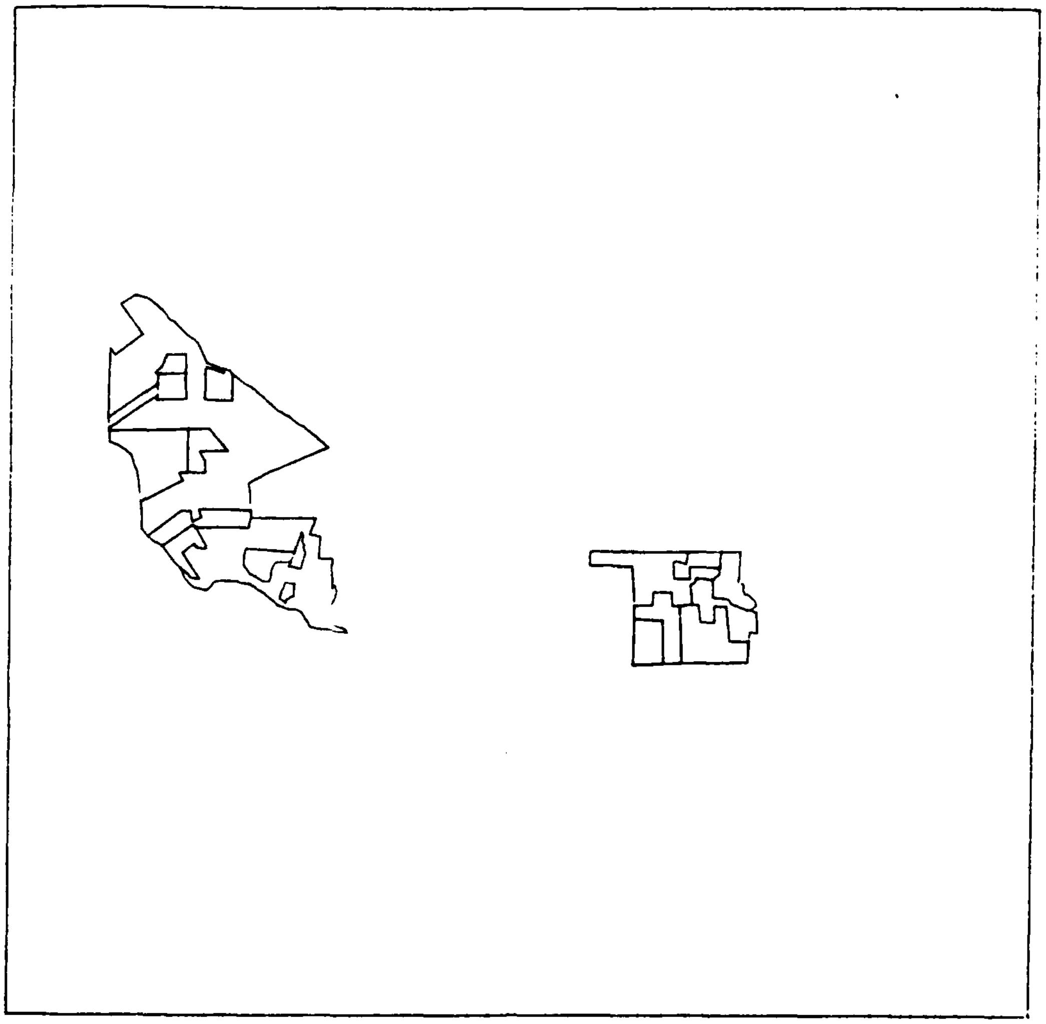


Figure 4.3 Land use areas affected by the update.

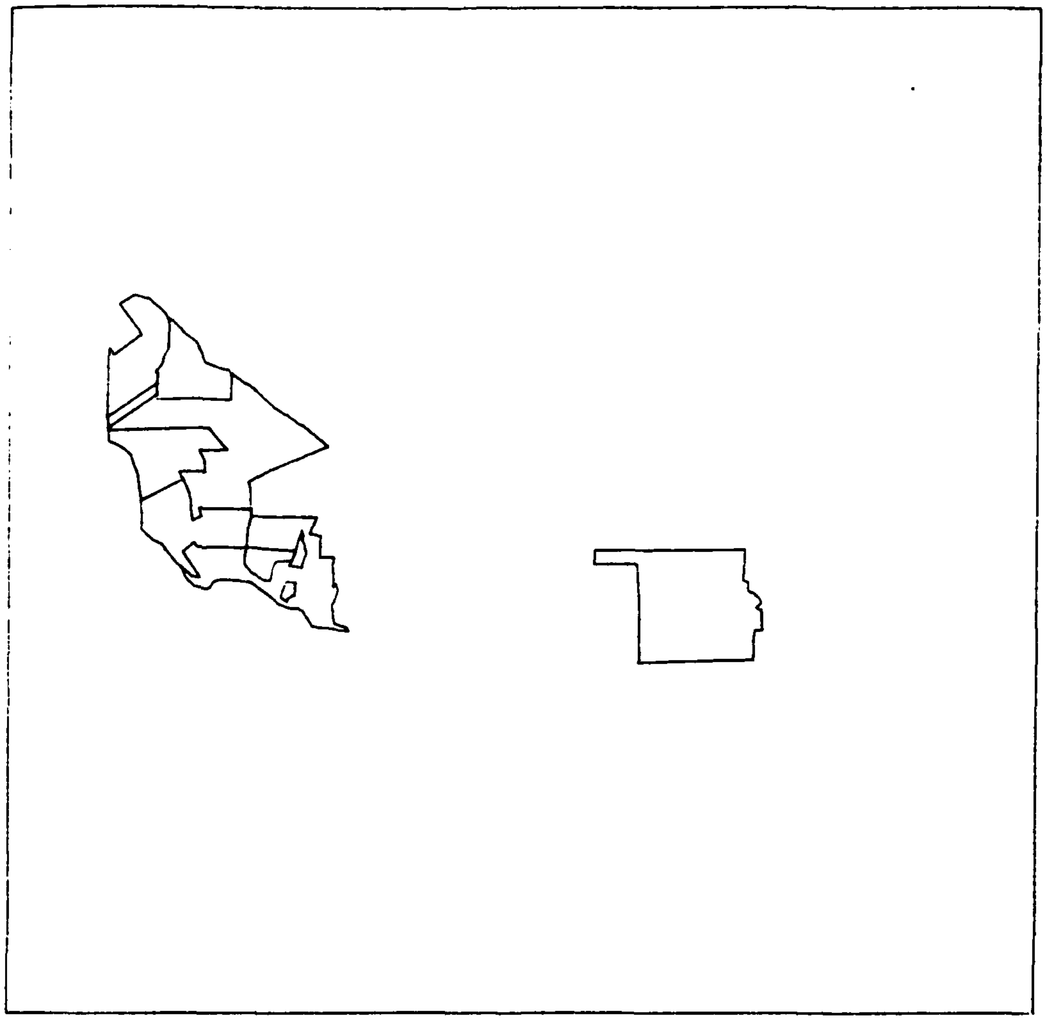


Figure 4.4 Land use area after update.

4.2 Recommendations

The present work has shown the ability of hexagonal data structures and associated software to manipulate a particular type of data (polygons) in a relatively small region (the Healdsburg area). To find the applicability of this type of spatial data manager to the actual ETL/DMA data management problem more work is required. The following tasks are extentions of the present work that would be beneficial in understanding the operational utility of haxagonal data structures.

a. Rework the polygon handling concept along the lines suggested in section 4.1.2. Recode the new concept and compare the results with the present work.

b. Intergrate point and line data into the existing Healdsburg polygon database. Provide algorithms for analyzing interrelated data types.

c. Extend the coverage to a significantly large area, e.g. the United States or the World. Explore the question of performance degradation as files grow.

d. Intergrate the existing polygon with pixel based imagery in such a way that the polygon boundaries can be displayed over the image background. Modify the editing software to allow an operator to easily modify boundaries to coincide with the image.

These tasks if accomplished would permit a more through understanding of the utility of a hexagonal data management system and related software in handling operational map databases.

APPENDIX A

A.1 LANDUSE AREA TABULATIONS

The required area measurements for the study contract were produced by invoking the special purpose software to compute polygon areas. Table A-1 contains the results for all the landuse regions. Table A-2 lists the area of the landuse polygons affected by the landuse revision. Table A-3 lists the area of the landuse polygons after performing the landuse revisions. Table A-4 lists the landuse areas for the region utilized in the polygon merge test region. Table A-5 lists the landuse areas after invoking the map sheet merge function. Table A-6 tabulates the area of the landuse polygons that are within the floodplain and below 100 feet.

The graphics plots of the study area are provided under a separate cover (Appendix C). These plots were produced at the same scale as the input documents. The following is a list of the plots provided.

C-1. Landuse Polygons.
C-2. Landuse Polygons with ID's.
C-3. 100 year Floodplain Polygons.
C-4. 100 year Floodplain Polygons.
C-5. Elevation Contour Polygons.
C-6. Elevation Contour Polygons.
C-7. Intersection of all Landuse areas within the Floodplain and below 100'.
C-8. Intersection of all Landuse areas within the Floodplain and below 100' with ID's.
C-9. Polygons before Landuse revision.
C-10. Polygons before Landuse revision with ID's.
C-11. Polygons after Landuse revision.
C-12. Polygons after Landuse revision with ID's.
C-13. Polygons before merge operation.
C-14. Polygons before merge operation with ID's.
C-15. Polygons after merge operation.
C-16. Polygons after merge operation with ID's.
C-17. Landuse Polygons with code ACC.

LANDUSE CODE	POLYGON ID	AREA (ACRES)
ACC	14	8.59
	22	0.95
	39	33.39
	41	11.36
	46	1.71
	48	19.98
	49	19.45
	51	15.00
	55	14.19
	58	66.67
	64	30.32
	77	38.17
	92	12.59
	103	10.95
	110	26.66
	114	17.01
	143	8.16
	146	11.55
	224	11.70
		358.40
ACP	5	37.87
	10	13.56
	15	7.57
	16	19.55
	38	10.40
	52	789.88
	61	19.95
	67	5.10
	83	84.61
	95	15.89
	100	8.72
	101	56.68
	108	19.35
	133	32.40
	137	15.33
	179	342.30
		1479.15
AR	59	7.48
	63	24.89
	81	11.80
	140	7.94
	207	11.46
		63.57

Table A-1. Acreage for all landuse areas.

AVF	3	60.83
	12	47.57
	13	12.71
	18	2.71
	19	12.95
	21	8.87
	23	6.09
	24	49.96
	25	92.96
	29	40.38
	53	52.92
	57	2.16
	68	4.29
	87	43.39
	96	58.41
	106	6.67
	112	7.72
	116	4.40
	120	4.86
	128	17.87
	135	39.04
	147	214.15
	150	7.41
	177	277.40
	181	35.59
	184	15.77
	188	17.46
	197	6.46
	210	5.37
	214	62.93
	223	98.45
		1317.74
AVV	2	9.51
	4	18.60
	9	60.93
	27	7.39
	31	654.15
	34	2.17
	35	3.30
	37	18.94
	42	39.89
	50	18.54
	56	184.35
	65	16.72
	74	6.49
	76	253.36

Table A-1 (Continued). Acreage for all landuse areas.

	80	30.32
	82	4.26
	84	9.17
	85	19.44
	88	18.01
	98	2.39
	102	12.30
	104	11.67
	105	2.31
	107	0.68
	111	17.60
	113	3.78
	117	12.99
	126	102.42
	132	18.99
	134	20.77
	136	8.71
	138	19.68
	141	12.32
	142	5.70
	144	4.97
	173	13.39
	178	5.90
	182	20.99
	190	3.78
	217	1.91
	219	10.44
	307	6.53
		1695.55
BBR	32	4.48
	78	21.36
		25.84
BEQ	8	14.12
		14.12
BES	170	9.05
		9.05

Table A-1 (Continued). Acreage for all landuse areas.

BT	91	7.36
	109	29.45
	123	67.07
	180	14.41
	201	9.31
		127.60
FO	69	1.23
	205	146.51
	209	478.04
	213	147.44
	218	37.63
		810.85
ISC1	305	12.66
		12.66
ISC2	306	4.19
		4.19
LR	43	38.19
	79	7.67
	168	3.30
	183	6.54
		55.70
OUV	193	7.67
		7.67
R	121	2.24
	124	30.45
	125	583.08
	206	50.31
	208	66.76
	212	127.16
	215	57.68
	308	7.79
		925.48

Table A-1 (Continued). Acreage for all landuse areas.

UCB	159	5.29
	202	6.91
		12.19
UCC	71	9.13
	158	5.86
	186	7.25
	189	7.29
	200	24.72
		54.25
UCR	157	77.52
	164	5.61
	196	2.98
		86.11
UCW	152	2.30
	154	4.98
	155	10.13
		17.41
UES	28	36.02
	175	54.38
		90.41
UIL	45	10.47
	161	12.39
		22.86
UIS	62	15.18
	89	4.11
	145	9.00
	163	4.79
	172	6.96
	185	16.13
	195	0.93
	198	1.56
		58.66
UIW	54	7.81
		7.81

Table A-1 (Continued). Acreage for all landuse areas.

UOC	191	16.35
	192	3.73
		20.08
UOG	187	64.69
		64.69
UOO	47	5.15
	153	16.01
	165	7.83
		28.99
UOP	156	1.47
	171	10.70
		12.17
UOR	199	2.41
		2.41
UOV	194	7.73
		7.73
URH	60	2.62
	167	7.99
		10.61
USR	6	11.32
	11	2.19
	17	11.99
	20	3.02
	36	10.84
	40	23.24
	66	6.40
	72	22.02
	73	64.29
	75	3.82
	86	29.44
	90	49.53
	93	10.88

Table A-1 (Continued). Acreage for all landuse areas.

	97	3.21
	99	14.28
	127	6.41
	131	14.79
	149	9.05
	151	14.44
	162	21.46
	166	21.44
	174	41.62
	176	3.55
	203	490.39
	211	27.26
	216	15.96
	220	590.08
		1522.90
UIW	139	3.39
		3.39
UUS	26	14.34
	70	0.93
		15.26
UUT	44	65.34
	148	34.95
	160	10.70
		110.99
VV	169	6.24
		6.24
WO	7	7.20
	30	60.88
		68.08
WS	33	193.65

Table A-1 (Continued). Acreage for all landuse areas.

WWP	1	0.42
	115	4.79
	118	6.63
	119	5.65
	122	1.82
		19.31
		9343.77

Table A-1 (Continued). Acreage for all landuse areas.

LANDUSE CODE	POLYGON ID	AREA (ACRES)
ACC	142	2.81
	148	26.19
	181	56.74
		85.74
ACP	144	6.61
	171	1.89
		8.50
AVF	110	0.47
	114	17.94
	115	2.16
	123	9.41
	130	15.46
	133	32.92
	136	0.03
	138	42.23
	140	0.32
	141	3.68
	146	41.37
	153	34.08
	154	1.79
	159	27.86
	161	16.44
	162	2.23
	169	1.32
	170	2.78
	175	0.00
	176	12.12
	188	0.00
	193	40.38
	199	12.71
		317.70
AVV	116	7.15
	129	4.40

Table A-2 Acreage for all intersection areas.

	137	445.46
	139	0.10
	143	0.06
	145	8.33
	147	108.78
	150	15.83
	152	11.42
	158	2.04
	160	5.34
	165	34.80
	168	1.44
	172	3.50
	173	4.00
	174	0.96
	182	0.36
	184	1.27
		655.26
BBR	195	4.48
	196	21.36
		25.84
BEQ	191	14.12
		14.12
BES	122	7.09
		7.09
BT	124	4.19
		4.19
FO	102	2.32
	105	13.79
	109	4.38
	134	0.08
		20.58
LR	164	37.17

Table A-2 (Continued). Acreage for all intersection areas.

	179	4.51
	180	0.04
	197	7.67
	198	3.30
		52.69
R	99	0.58
	113	3.31
		3.89
UCC	149	0.00
		0.00
UES	121	33.17
	185	34.81
		67.98
UIL	187	0.00
		0.00
UIS	119	6.03
		6.03
UOO	189	4.89
		4.89
UOP	120	8.47
		8.47
URH	177	0.73
		0.73
URS	98	3.24
	100	1.29

Table A-2 (Continued). Acerage for all intersection areas.

	103	0.31
	104	5.15
	106	8.03
	107	0.04
	108	0.06
	111	2.95
	112	20.35
	118	2.72
	125	5.04
	126	1.42
	128	6.46
	131	0.61
	132	1.29
	135	0.19
	151	2.47
	156	3.09
	157	4.33
	163	7.14
	166	0.09
	167	0.01
	178	0.06
		76.32
UUS	183	0.26
	192	14.34
		14.60
UUT	127	0.01
	155	1.48
	186	11.30
		12.80
VV	117	6.12
		6.12
WO	190	7.20
	194	30.44
		37.64
WS	101	184.36
		184.36
		1615.52

Table A-1 (Continued). Acreage for all intersection areas.

LANDUSE CODE	POLYGON ID	AREA (ACRES)
ACC	41	11.36
	143	8.16
		19.52
ACP	38	10.40
	133	32.40
	137	15.33
		58.13
AVF	135	39.04
	147	214.81
	223	98.45
		352.30
AVV	34	4.35
	35	6.61
	37	18.94
	42	39.89
	132	18.99
	134	20.77
	136	8.71
	142	6.12
	144	4.97
		129.35
USI	145	9.00
		9.00
USR	36	21.67
	131	14.79
		36.47
		604.77

Table A-3. Acreage for landuse areas affected by update.

LANDUSE CODE	POLYGON ID	AREA (ACRES)
ACC	38	51.25
		51.25
AVF	36	48.70
	37	130.88
	133	87.77
		267.35
AVV	35	4.35
	41	56.96
	42	3.30
	142	6.12
		70.73
UIS	34	40.42
		40.42
URS	131	150.04
	132	10.84
		160.88
		590.63

Table A-4. Acreage for landuse areas after update.

LANDUSE CODE	POLYGON ID	AREA (ACRES)
ACP	3	6.95
	8	1,45
	10	7.97
	11	3.64
	13	115.98
	14	7.23
	16	29.71
		172.94
AVF	6	38.74
		38.74
AVV	2	57.83
	5	11.40
	7	8.84
	9	5.36
	15	15.35
	17	7.13
	19	0.36
		106.27
R	1	34.51
	12	1.22
		35.73
URS	4	8.56
	18	5.87
		14.43
		368.11

Table A-5. Acreage of landuse polygons in triangular regions to be merged.

LANDUSE CODE	POLYGON ID	AREA (ACRES)
ACP	12	30.94
	16	124.49
	18	14.72
		170.16
AVF	6	38.74
		38.74
AVV	7	8.84
	13	18.25
	17	57.92
	19	20.53
		105.54
R	21	35.02
		35.02
URS	14	13.84
		13.84
		363.30

Table 6. Acreage of landuse polygons in triangular regions after merging.

APPENDIX B

INTRODUCTION TO GBT

Dean Lucas and Laurie Gibson

WHAT IS GBT?

GBT is a method of representing a two dimensional surface that enables a computer to work easily with data distributed on that surface. When a person looks at data in two dimensions, a map sheet or a photographic image, for example, he quickly sees the general content of that map or photograph, and can examine certain aspects of the content in detail if he chooses to focus his attention. Computers which use conventional representations for planar data have difficulty performing this simple chore. Such systems permit easy examination of the smallest components of the data, be they pixels, bits, bytes, or vectors, but have no efficient mechanisms for examining the data in aggregate form. As the saying goes, they cannot see the forest for the trees.

The GBT method of structuring a surface permits data to be aggregated so that an automated system can examine the general content of the data without looking at the detail. Because of this capability, an algorithm can determine at a high level if the generalized information suffices for the algorithm's purpose or whether finer information is needed. In the latter case, the GBT structure permits the layered accessing of finer and finer data until the finest granularity is reached. This capability for selective access to successive levels of detail is taken for granted in human perception, but it has presented a significant problem in designing systems for machine perception. It is fundamental, for example, to the notions of scene analysis, feature extraction, and pattern recognition. The GBT system solves this problem.

GBT is not only a method for representing space, it is also an addressing scheme that allows access to that representation. Further, it contains an algebraic system which operates on the addressing scheme. These aspects of GBT will be described below. A GBT structure can be implemented in any dimension, however, a discusson of dimensions other than the second is beyond the scope of this paper. GBT stands for Generalized Balanced Ternary, a term signifying that GBT is the higher dimensional analogue of the one dimensional system known as balanced ternary.

THE STRUCTURE AND ADDRESSES

The GBT structure is one of a hierarchy of cells. At each level, the cells are constructed of cells from the previous level according to a rule of aggregation. The basic cells of this structure are hexagons. Figure 1 shows the hexagonal covering of the plane. This covering has the uniform adjacency property, that is, each element of the covering is adjacent to exactly six other elements and shares with each exactly one-sixth of its boundary. In contrast, a covering of the plane with squares does not have uniform adjacencies. Some squares are adjacent at a point while others share a side.

A first level aggregate is formed by taking a single hexagon and its six neighbors (see Figure 2a). The first level aggregates also cover the plane and have the uniform adjacency property. In general, an aggregate at level n is formed by taking a level n-1 aggregate and its six neighbors. It can be shown that the planar covering and uniform adjacency properties hold at each level. Figures 2b and 2c show second and third level aggregates.

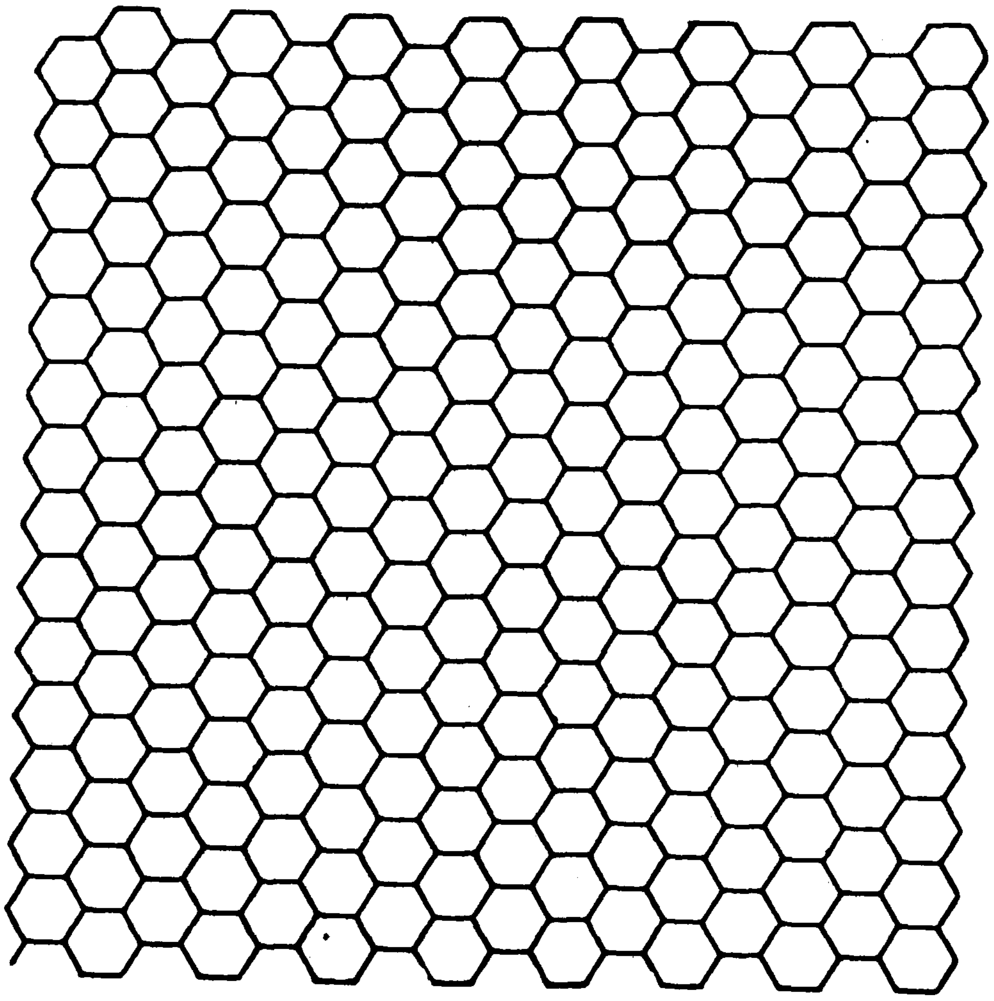
The GBT addressing system is based on the following scheme. In an aggregate, the center cell is labeled 0 and the outer six cells are labeled, in clockwise order, 1, 3, 2, 6, 4, 5 (see Figure 2). Each hexagon in the plane has a unique GBT address, a sequence of digits corresponding to the labels of the cells above that hexagon.

Each digit of the address corresponds to an aggregate level. For example, the address 536 labels the hexagon in the 6 position of the first level aggregate, which is in the 3 position of the second level aggregate, which is in the 5 position of the third level aggregate, which is at the 0 or center position at all higher levels. The hexagon 536 is shaded in Figure 3.

The digit 7 is used to address entire aggregates rather than hexagons. Thus, the first level aggregate shaded in Figure 4 has address 117. The second level aggregate outlined in the same figure has address 677.

The symbols 0, 1, 2, 3, 4, 5, 6, and 7, used as GBT digits, are also used in octal and decimal notation to express integers. Using them in this new context allows GBT addresses to be handled directly by computer hardware and software. Care must be taken, however, not to confuse the addresses with integers.

FIGURE 1: The Hexagonal Covering of the Plane



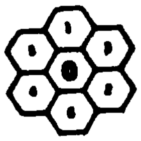


Figure 2A: First Level Aggregate

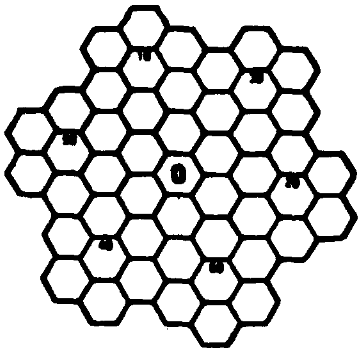


Figure 2B: Second Level Aggregate

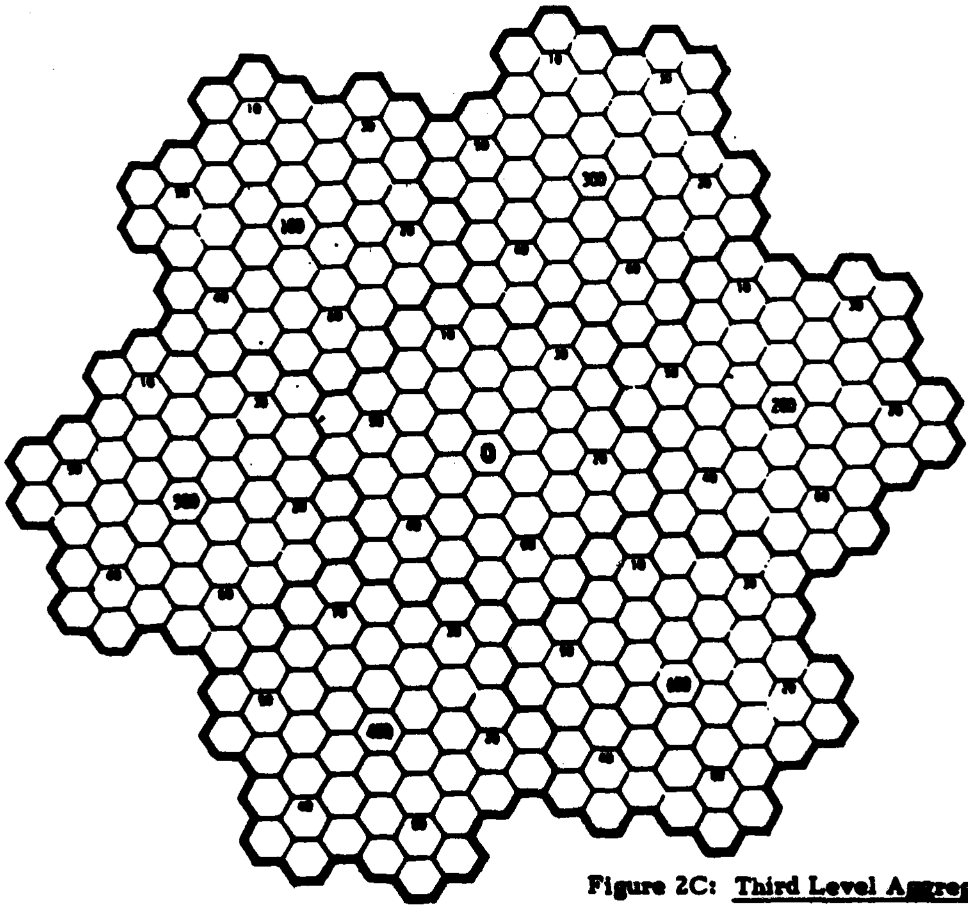


Figure 2C: Third Level Aggregate

FIGURE 2: The Aggregate Structure

FIGURE 3: The Location of the Hexagon

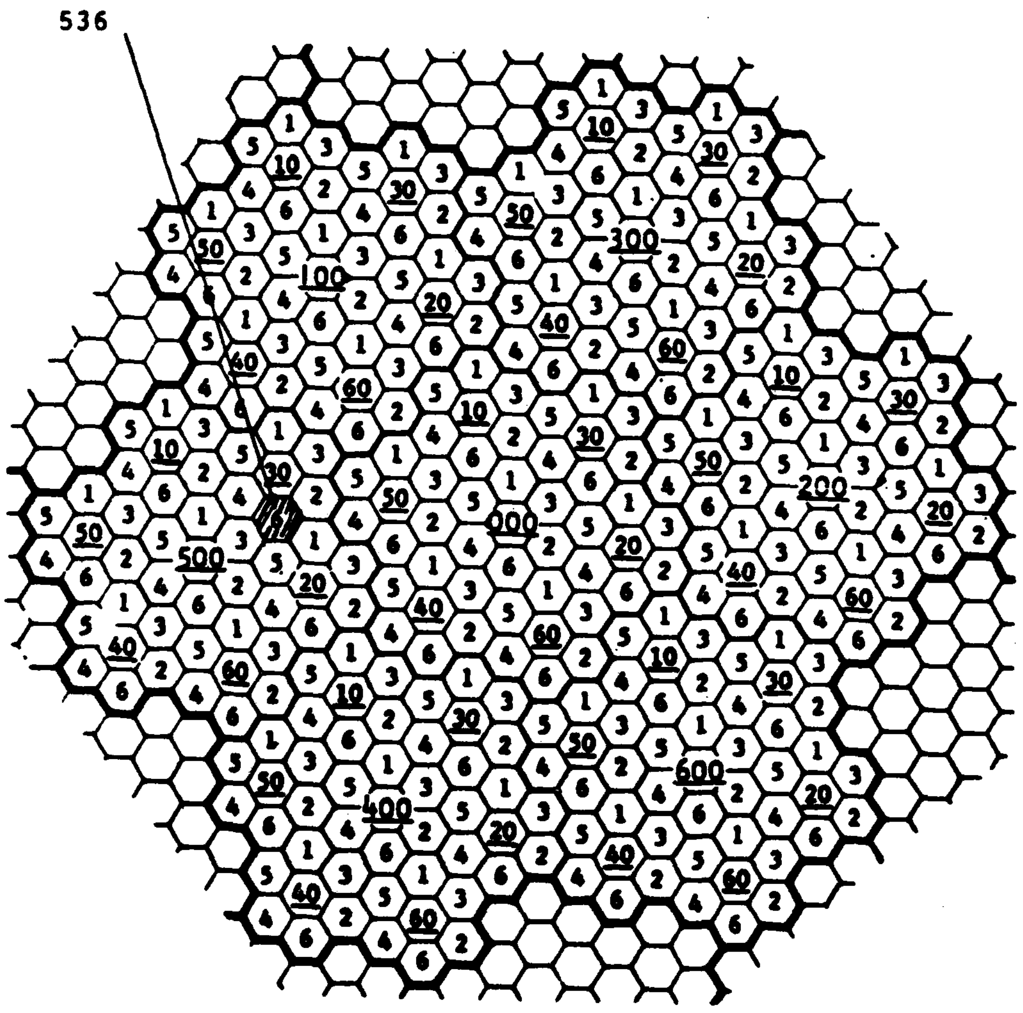
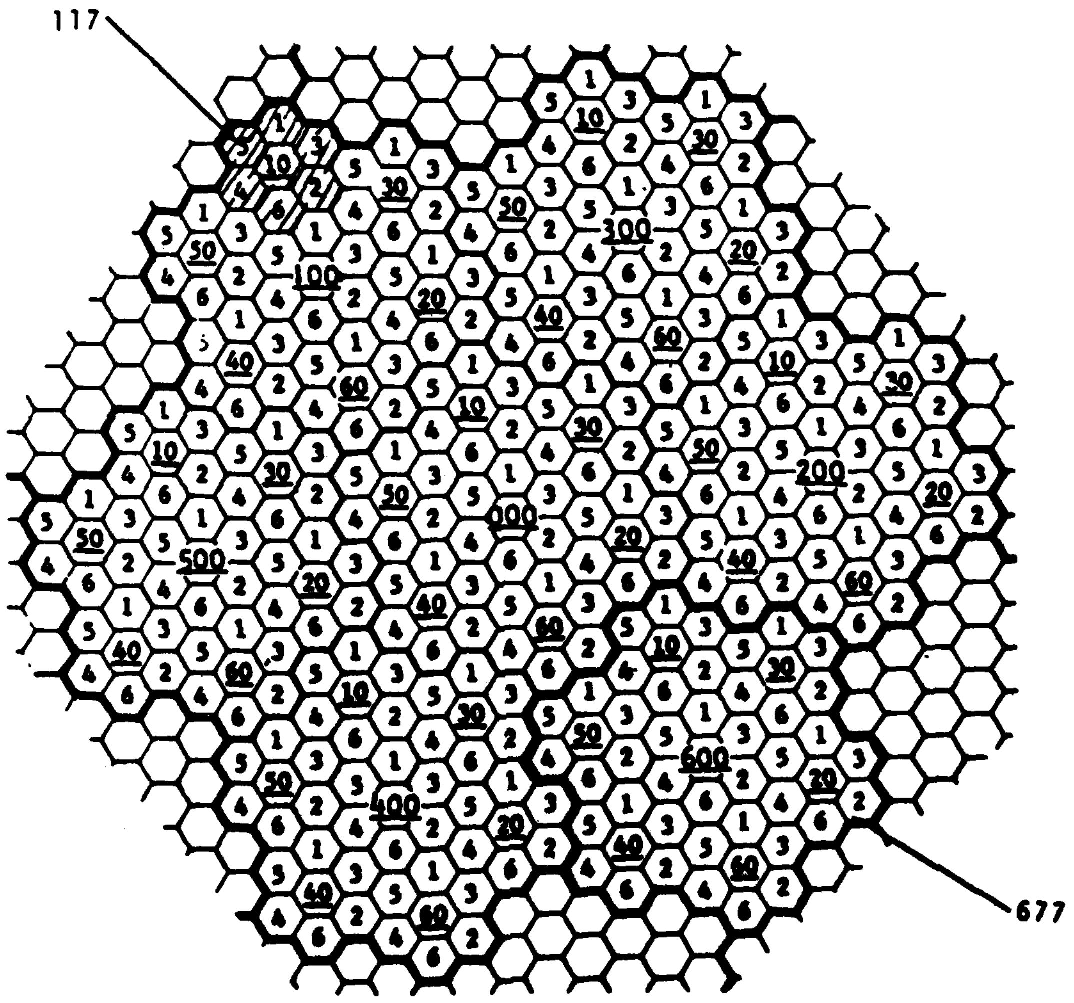


FIGURE 4:

Two Examples of the Use of the Failing Sevens Notation to Address Higher Aggregates



THE ARITHMETIC

In order for the GBT addressing system to be useful, there must be efficient methods for doing planar addition, subtraction, and multiplication entirely in terms of GBT addresses. These methods are discussed below:

1. Addition: Addition of GBT addresses parallels integer addition in that if an addition table for the seven digits is given, any two addresses can be added. From Figure 5, we can derive the basic GBT addition table. Using standard planar parallellogram addition, we see, for example, that 1+2=3, 3+6=2, and 5+6=4. If 3 and 2 are added, the sum is outside the central first level aggregate, 3+2=25. Thus, Table 1 is obtained. Addition of multidigit addresses is very much like adding multidigit integers. For example (see Figure 6), to add 153 and 45, add 3+5=1, then 5+4=52, and carrying the 5 to the next column, 5+1=16. Thus, 153+45=1621. Figure 6B shows these vectors in the plane.

2. Subtraction: Subtraction is accomplished by the process of complementing and adding. The complement of a GBT address is its digitwise sevens complement. The complement of 61542 is 16235, for example. In Figure 5, we see that 3 and 4 are complements, as are 1 and 6, and 53 and 24.

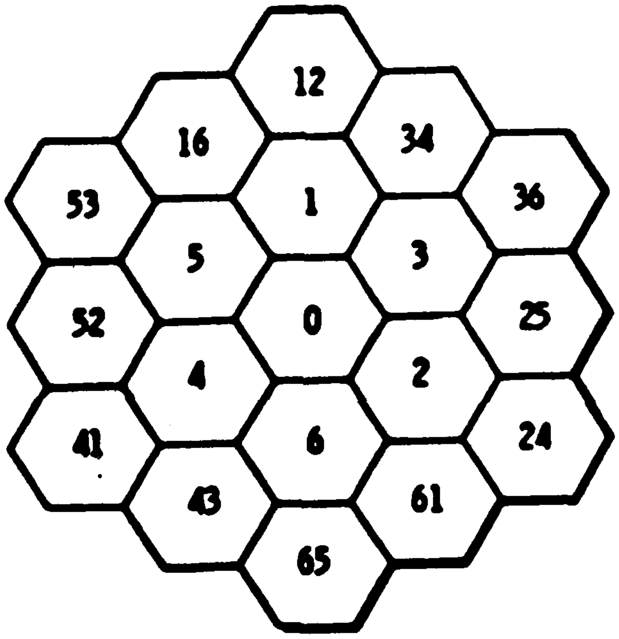
3. Multiplication: GBT multiplication is similar to addition in that it is a digitwise operation. The multiplication table (Table 2) shows that the GBT product of two digits is just their integer product modulo 7. An example of multidigit multiplication is:

```
254 x 62:

   254
 x  62
   431    (= 2 x 254)
  523     (= 6 x 254)
  5261    (= the GBT sum)
```

Figure 7 illustrates this product geometrically.

FIGURE 5: Key to Planar Addition Table



+	0	1	2	3	4	5	6
0	0	1	2	3	4	5	6
1	1	12	3	34	5	16	0
2	2	3	24	25	6	0	61
3	3	34	25	36	0	1	2
4	4	5	6	0	41	52	43
5	5	16	0	1	52	53	4
6	6	0	61	2	43	4	65

TABLE 1: **GBT Addition**

Figure 6A

Add the Addresses 153 and 45:

```
(1)(5)          ( ) Denotes a Carry
   153
    45
  1621
```

Figure 6B

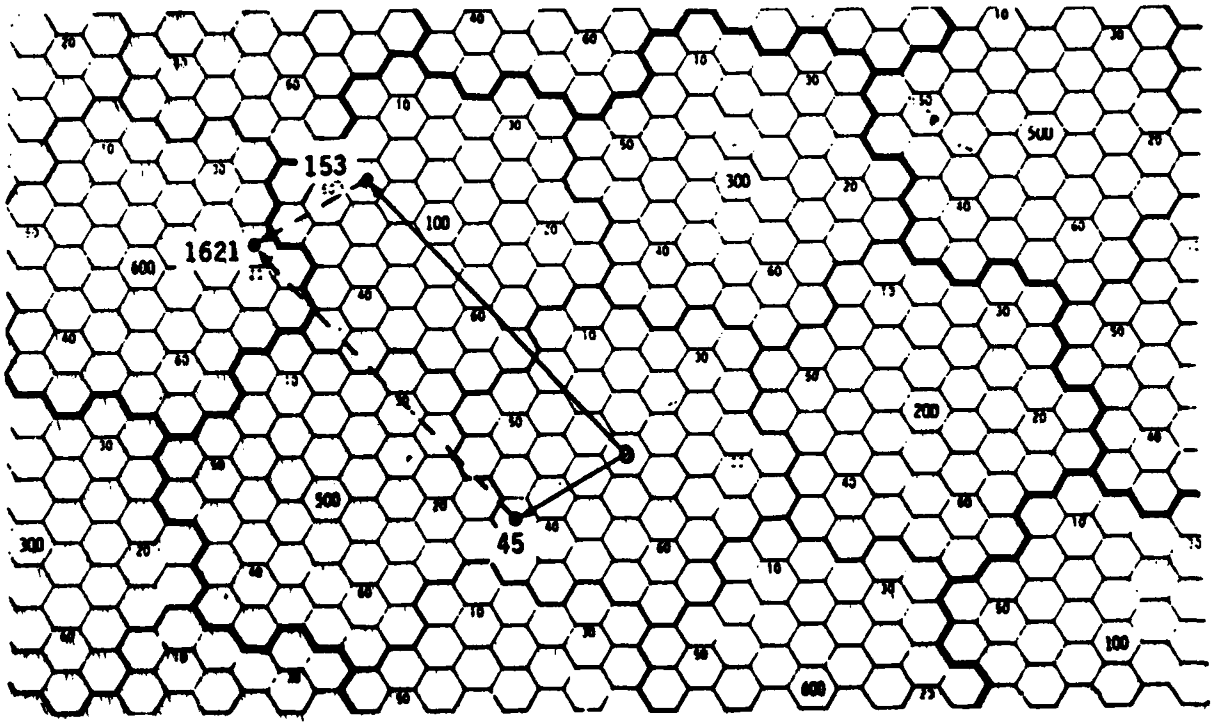
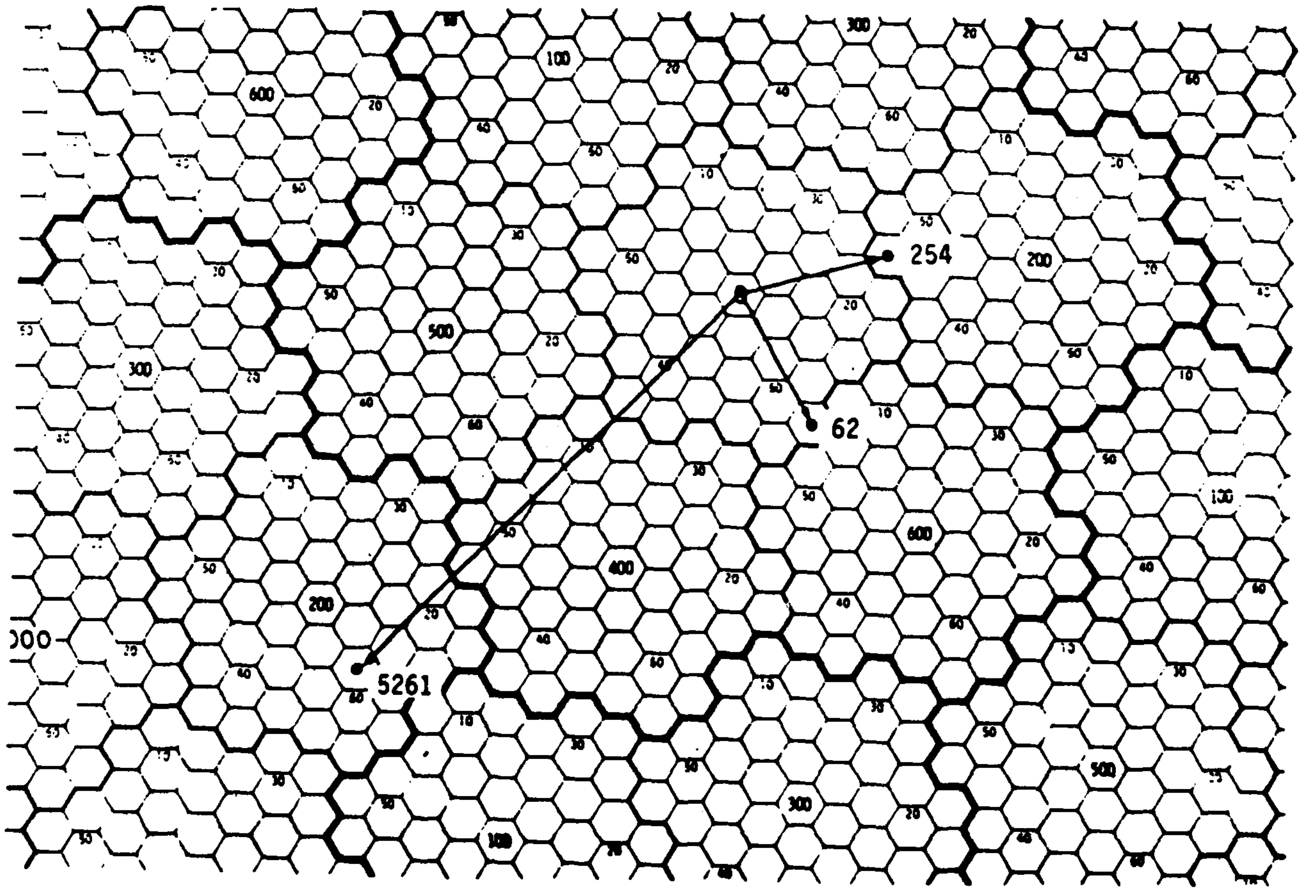


FIGURE 6: The GBT Sum of 153 and 45

•	0	1	2	3	4	5	6
0	0	0	0	0	0	0	0
1	0	1	2	3	4	5	6
2	0	2	4	6	1	3	5
3	0	3	6	2	5	1	4
4	0	4	1	5	2	6	3
5	0	5	3	1	6	4	2
6	0	6	5	4	3	2	1

TABLE 2: <u>GBT Multiplication</u>

FIGURE 7: The GBT Product: 254 x 62



APPENDIX C

The statement of work for contract DAAK 70-82-C-0133 specifies that the graphics plots required were to be produced at the same scale as the input map sheets. The following plots were produced at that scale for the study area.

C-1. Landuse Polygons.
C-2. Landuse Polygons with ID's.
C-3. 100 year Floodplain polygons.
C-4. 100 year Floodplain polygons.
C-5. Elevation Contour Polygons.
C-6. Elevation Contour Polygons.
C-7. Intersection of all Landuse areas within the Floodplain and below 100'.
C-8. Intersection of all Landuse areas within the Floodplain and below 100' with ID's.
C-9. Polygons before Landuse revision.
C-10. Polygons before Landuse revision with ID's.
C-11. Polygons after Landuse revision.
C-12. Polygons after Landuse revision with ID's.
C-13. Polygons before merge operation.
C-14. Polygons before merge operation with ID's.
C-15. Polygons after merge operation.
C-16. Polygons after merge operation with ID's.
C-17. 300' Contour polygons.
C-18. Landuse polygons with code ACC.

REFERENCES

1. D. Lucas and L. Gibson, Automated Analysis of Imagery, Final Report on Contract Number F49620-82-C-0070, November 1982.

2. D. Lucas, "A Multiplication in N-Space," Proceedings of the American Mathematical Society, Volume 74, Number 1, April 1979

3. D. Lucas and L. Gibson, A System for Hierarchical Addressing in Euclidean Space, Martin Marietta Corporation, January 1980.

4. L. Gibson and D. Lucas, "Spatial Data Processing Using Generalized Balanced Ternary," Proceedings of the IEEE Computer Society Conference on Pattern Recognition and Image Processing, June 1982.

5. D.E. Knuth, "The Art of Computer Programming," Vol. 2, Addison - Wesley, Reading Mass., 1969.