REPRESENTATIONS OF SURFACES
AN ALGORITHM FOR A THREE-DIMENSIONAL HISTOGRAM WITH HIDDEN-LINE REMOVAL

Lisa Welbourn* and Susan Laflin
University of Birmingham

Abstract

The first program for representation of survey data was presented by N Sutton at the 1983 Computer Applications in Archaeology Conference. This work has continued and sections in contouring, shaded mosaic, projection drawings of the surface and three-dimensional histograms are now available, to varying extents, on the two mainframes at Birmingham University. They are written in Fortran using the GINO-F package and this imposes some limitations. The latest of these is described in detail in the remainder of this paper.

The program is designed to display data that would typically be produced by a geophysical survey or that which could be represented by a rectangular array of spot height values. Such data would be generated by, for instance, magnetometer surveys, pottery counts or the simple height of the ground above the sea level studies.

Each point in the array is represented by a three-dimensional pillar of the appropriate height, positioned at that point. Initially, this leads to a most confusing picture, with no means of depth perception. Therefore, a hidden-line algorithm is used to remove those surfaces and edges of the pillars that would be obscured from view by other pillars. The use of such an algorithm has the advantages that the user can obtain hard copy of the results from a plotter, for use in the report; that the viewing position can be selected by the user; and that the results can be obtained with a reasonable amount of speed.

The hidden-line algorithm designed and implemented will be described and some examples of the results obtained will be shown.

* now at Department of Computing, Trent Polytechnic, Nottingham.

Introduction

The SURVEY program for presentation of survey data recorded as a rectangular array of spot heights has been extended by several student projects over the past four years. The first project by N Sutton in 1982/83 set up the framework and produced output on the Sigma colour terminals for mosaic and shaded representation and a rather slow version of the three-dimensional histogram using the painters' algorithm. These representations in a variety of colours, together with some facilities from the GINO-F package remain as the implementation on the DEC-20 system.

Later projects transferred to the Honeywell mainframe running under the Multics operating system. In 1984/85, two undergraduate projects were completed. One, by A Hilton studied various contouring methods and produced a suitable section. The other, by M Parker, studied shading methods to simulate grey-scale output on a black and white terminal and produce a version of mosaic output. In the summer of 1985, two MSc students completed the set of output versions. M Perez-Casanova studied the problem of producing a projection drawing of a surface with hidden-lines correctly removed and so produced output on both plotter and terminal. L Welbourn studied the problem of algorithms for hidden-line removal and produced the software for fast and accurate production of three-dimensional histograms. Unlike the earlier version, her software can be used on a plotter as well as terminals and is very much faster than the painters' algorithm.

Table 1 shows which versions of software are available under which system at Birmingham. You will note that the choice of colour table is not available under Multics. In the remainder of this paper, one of us, L Welbourn, describes the algorithms included in her project. No further work on this program will be attempted, but future projects supervised by S Laflin will relate to similar work based on the Graphical Kernal System rather than on GINO-F.

The Hidden-Line Algorithm

The algorithm described here has been designed to work with large matrices of spot height values, only using a small amount of storage and capitalizing on the fact that all the objects are of the same type. Most hidden-line/surface algorithms are not designed to cope with large numbers of objects as they need to access so much information about each object. Many need the x, y and z components for each vertex of each object. The SURVEY package was designed for matrices of up to 100 x 100 elements in size ie 10,000 objects and therefore

$$3 \times 8 \times 10{,}000 = 240{,}000$$

separate pieces of data are required.

In this algorithm, each pillar is considered individually, the coordinates being calculated and used before being replaced by a new set belonging to the next pillar. The algorithm works in image space, that is, the three-dimensional coordinates of the vertices are calculated and transformed into the two-dimensional plane in which they are to be viewed before any hidden-line algorithm is applied.

The histogram is constructed from an m column by n row matrix of spot height values (Figure 1) that is considered to line in the xz plane with heights extending in the y direction. The viewer is also assumed to be sited in the xz plane and as such would only be able to view one face of each pillar. A transformation is performed to move the histogram and allow views of three faces, the top and two of the sides. The transformation consists of two rotations, one about the y vertical axis affecting the views of the side face, and the other about the x horizontal axis causing more or less of the top face to be seen. Once the coordinates have been calculated and transformed, an orthographic projection is performed on them so that they can be drawn in two dimensions. The one used is quite simple and just involves ignoring the z component of the coordinates.

Any pillar being drawn in this manner will have at least one of its vertices obscured from view by its own volume (Figure 2). Also hidden by the volume of the pillar will be at least three of the edges. These are the edges that connect with the hidden vertex, shown as dashed lines in the diagram. These points and edges need never be considered by the algorithm.

The (m,1)th pillar is the first to be drawn, and can be drawn in full without any calculation as there is nothing to obscure any of it from view. This pillar is at the front of the histogram and may, therefore, hide parts of other pillars that lie partly or wholly behind it. The silhouette of this pillar can be used to form a mask as no further drawing should take place inside it.

The remaining pillars on the first, front, row may now be constructed. Vertices 1, 2, 5 and 6 of each of these pillars will always be visible, but vertices 3, 7 and 8 must be tested against the mask for their visibility. If these points lie outside of the mask or on its edge, then they are considered to be visible. Any edge that joins two visible vertices can be drawn in full. Any edge that joins two hidden points may be totally omitted. But some further calculation must be done if one vertex is visible and one invisible. The intersection point of the edge and the mask edge must be calculated. This is done simply using the formula for the intersection of two straight lines. The edge is then drawn from the visible point to the point of intersection (Figure 3). As each pillar is added to the histogram, the mask must be updated to include it, because the new pillar is likely to block the view of other pillars lying behind it.

Once the whole of the first row has been constructed, the other rows may be added, one by one, pillar by pillar. Not as many constraints may be placed on the drawing of each pillar ie vertices 1, 2, 5 and 6 will not always be visible. As the visibility of the vertices heavily depends on the contents of previous rows. Each vertex must be tested for visibility and edges only drawn as far as the mask if the second point lies beneath it. Again, the mask must be updated as each new pillar is drawn (Figure 4).

Implementation

As well as the routines dealing with the hidden-line algorithm, there are two major parts to the program: the selection of the viewing position and the scaling of the output to fit the device.

The algorithm draws a histogram from a matrix of m x n elements in size. The user may select any of the sides to be displayed as the front face. A default operates, but a small subroutine exists to reposition the elements of the array (Figure 5a). For instance, if the histogram is to be viewed with the face marked (Figure 5b) as the front face, then element (1,1) must become element (m,1) of the new matrix and element (1,n) must become element (1,1) etc.

The user may also select the angle at which the front face is viewed. This, combined with the selected front face, allows a view of the histogram from any position around it. An angle of elevation for the viewing position may be chosen, to allow a viewer to look into a histogram from above. This is useful for identifying hollows in the data and seeing pillars that would otherwise be hidden from view by much higher surrounding pillars.

The position of each vertex is calculated and then transformed using the angles and view chosen, so that they can be drawn on the device.

The rotational transformations can be described by the matrices

$$(x'\ y'\ z'\ 1) = (x\ y\ z\ 1)\begin{pmatrix} \cos\theta 1 & 0 & \sin\theta 1 & 0 \\ 0 & 1 & 0 & 0 \\ -\sin\theta 1 & 0 & \cos\theta 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

for rotations of theta1 about the y axis

$$(x'\ y'\ z'\ 1) = (x\ y\ z\ 1) \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\theta 2 & -\sin\theta 2 & 0 \\ 0 & \sin\theta 2 & \cos\theta 2 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

for rotations through theta1 about the x axis

Therefore, a point (x, y, z) is moved to

$$\begin{bmatrix} x.\cos\theta 1 - z.\sin\theta 1 \\ y.\cos\theta 2 + (x.\sin\theta 1 + z.\cos\theta 1)\ .\sin\theta 2 \\ -y.\sin\theta 2 + (x.\sin\theta 1 + z.\cos\theta 1)\ .\cos\theta 2 \\ 1 \end{bmatrix}$$

The parallel projection into the z = 0, xy plane for viewing is simply a matter of ignoring the z coordinate and, therefore,

$(x, y, z) \rightarrow (x\cos\theta 1 - z\sin\theta 1,\ y\cos\theta 2 + (x\sin\theta 1 + z\cos\theta 1)\sin\theta 2)$

These transformations are also used in the scaling routines. This is an important step in the program since, without it, the user would gain very little information about the data that would be of use. For example, if a 5 x 5 histogram was output using a scale suitable for a maximum 100 x 100 size histogram, the result would be 25 pillars with very tiny cross sections squashed up to the left-hand side of the screen. The vertical height of the pillars must also be scaled, so that histograms with vertical heights of 200 or more units will occupy the same vertical height as those with heights in the range 0 to 1, say. Four points are needed for the scaling process (Figure 6). A is the transformation of the point (m.xstep, 0, n.zstep) where xstep is the unit length in the x direction and zstep, the unit length in the z direction. B, the transformation of (m.xstep, 0, 0). C - (0, 0, n.zstep) and D (0, amaz.ystep, 0) where amaz is the height of the tallest pillar in the histogram. This must be used just in case the highest pillar occurs at point C. The horizontal distance 0A is equated to the length of the device and the vertical distance BD' to the width. These two simultaneous equations are then solved for xstep and ystep, assuming that xstep = zstep to keep the cross section of a pillar square.

When the routines mentioned above have been executed, the program will have gathered sufficient information to be able to begin the drawing stage. The coordinates of each vertex of the pillar can be calculated and the transformations implemented. The points may then be tested for their visibility against the mask. If the y coordinate component of the point is less than the y component of the mask edge at position x, then point (x,y) is hidden. The edges of the pillar are drawn, once any necessary intersection calculations have been carried out, using two simple GINO-F routines 'linto2' and 'movto2'.

The mask is the silhouette of the pillars. The line segments that form the edge of the mask are the edges of some of the pillars, and the vertices of the pillars are the ends of these segments. Therefore, the mask may be defined using the vertices. There is no need to define a bottom edge to the mask, as nothing will ever be drawn below it and no line will ever intersect with it.

Acknowledgements

One author, L Welbourn, would like to acknowledge the financial support given by the Science and Engineering Research Council, during the undertaking of an MSc, including a project on the topic included in this paper.

Bibliography

Interactive Computer Graphics:
Data Structures, Algorithms and Languages
W K Giloi
Prentice Hall 1978

Computer Graphics: A Programming Approach
S Harrington
McGraw-Hill 1983

Computational Geometry for Design and Manufacture
I D Faux, M J Pratt
Ellis Horwood 1981

Fundamentals of Interactive Computer Graphics
J D Foley, A Van Damm
Addison Wesley 1983

Principles of Interactive Computer Graphics
W M Newman, R F Sproull
McGraw-Hill 1983

TABLE 1

	DEC20	Honeywell
Mosaic	Sutton's project colour terminals Parker's project monochrome terminals	Parker's project monochrome terminals only.
Contouring	GINOSURF	Hilton's project
Parallel Projections	GINOSURF Isometric only	Perez-Casanova's project
3D Histogram	Sutton's project Colour terminals only	Welbourn's project

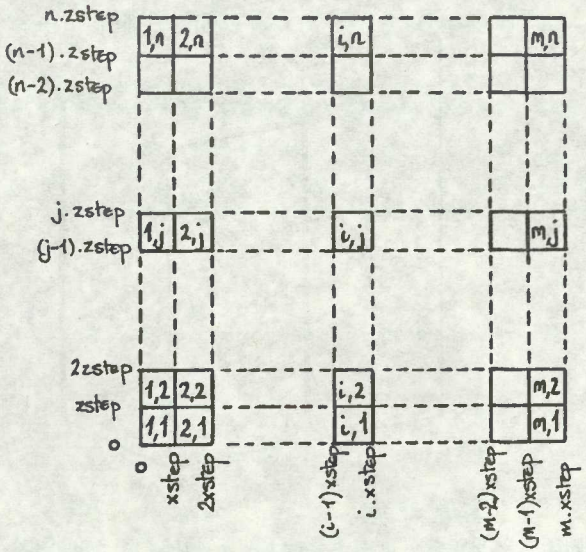


Figure 1

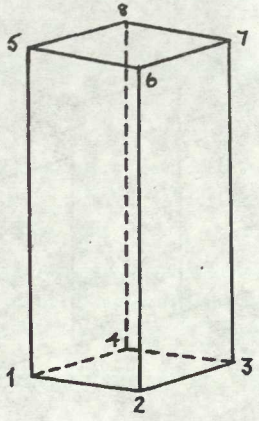


Figure 2

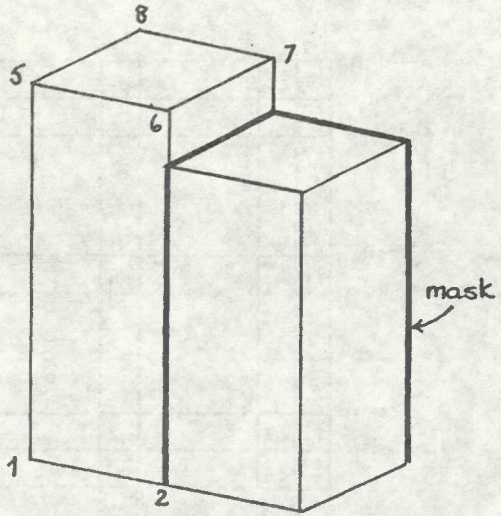


Figure 3

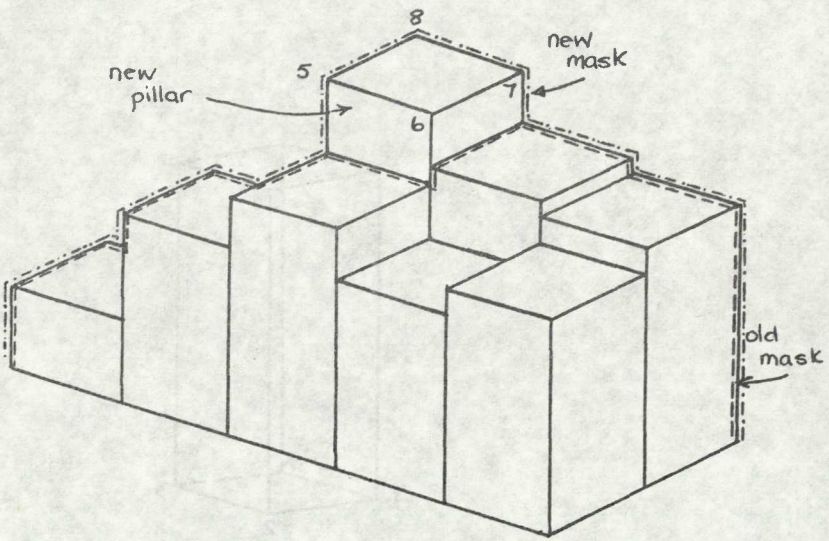


Figure 4

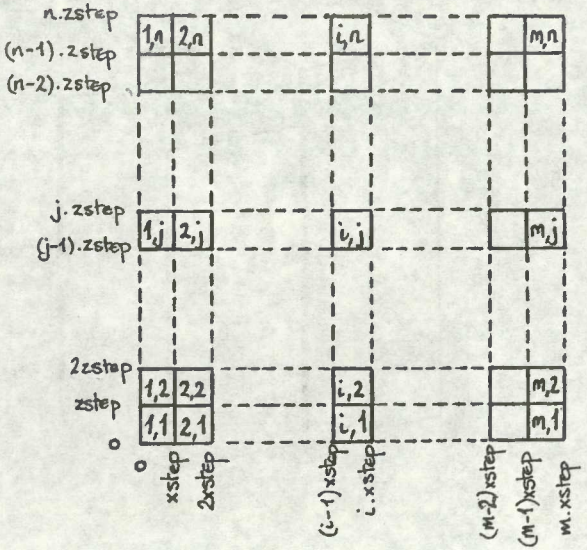


Figure 1

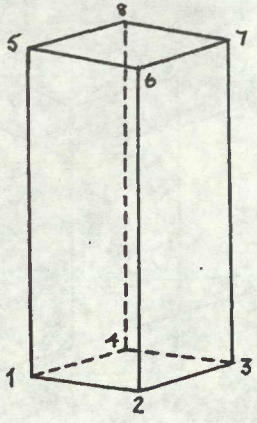


Figure 2

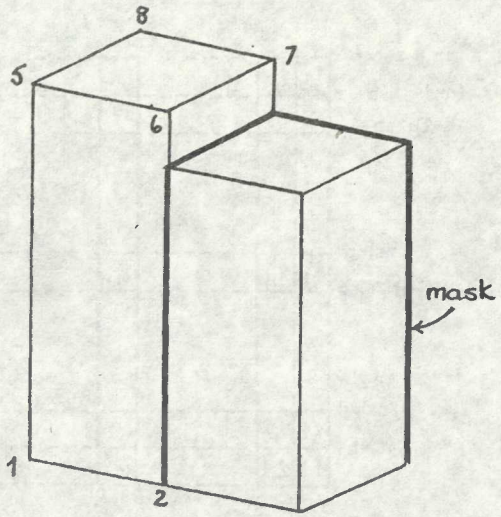


Figure 3

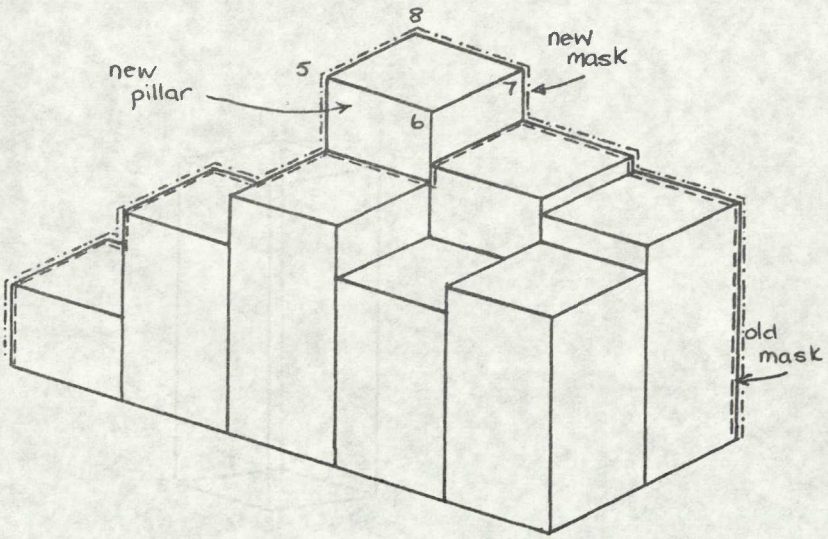


Figure 4

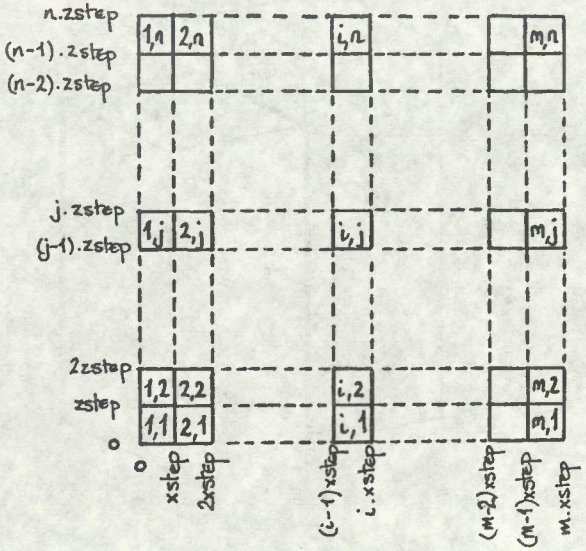


Figure 1

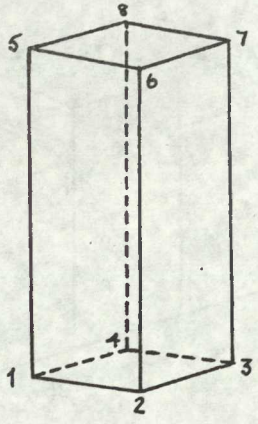


Figure 2

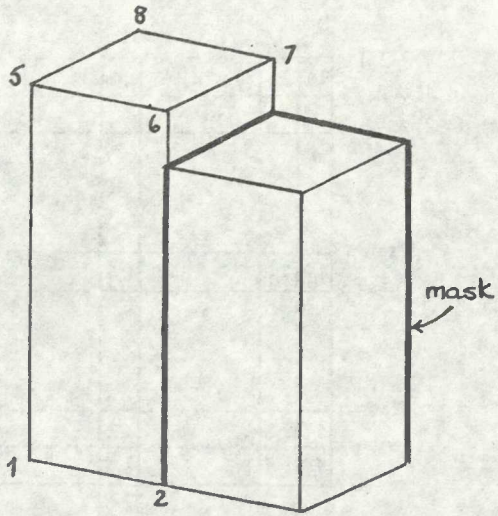


Figure 3

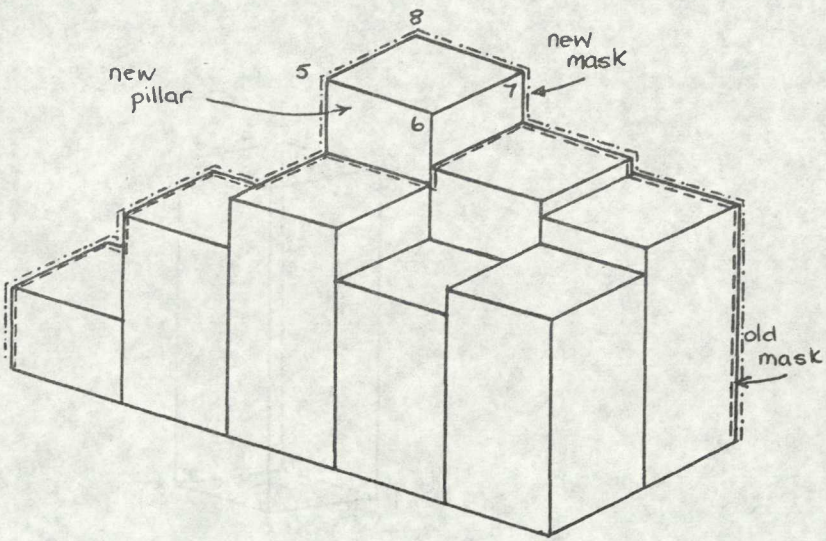


Figure 4

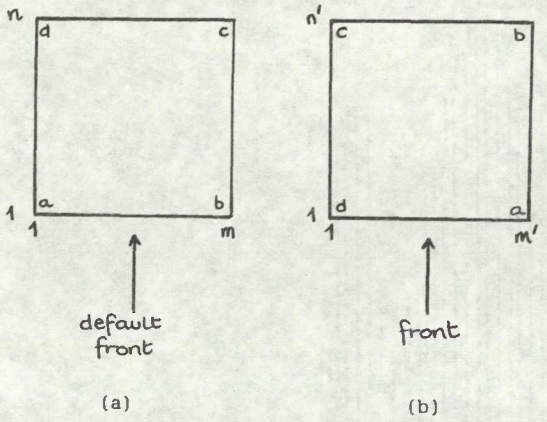


Figure 5

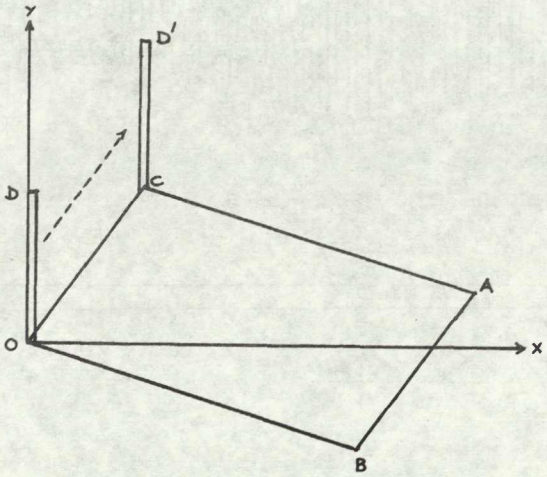
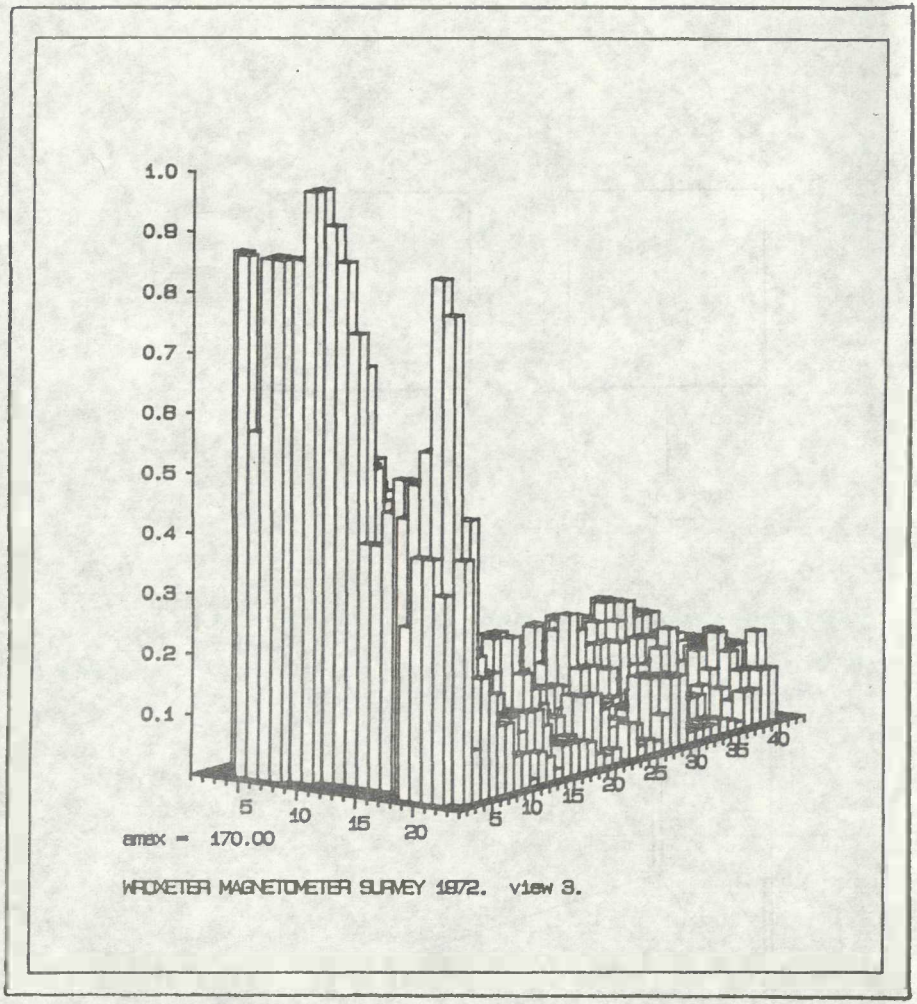


Figure 6



Example Output.