

THE HALOED LINE EFFECT FOR HIDDEN LINE ELIMINATION.

Arthur Appel
Computing Systems Department
IBM Thomas J. Watson Research Center
Yorktown Heights, New York 10598

F. James Rohlf
Department of Ecology and Evolution
State University of New York at Stony Brook
Stony Brook, Long Island, New York 11794

Arthur J. Stein
Computing Systems Department
IBM Thomas J. Watson Research Center
Yorktown Heights, New York 10598

ABSTRACT: The haloed line effect is a technique where when a line in three-dimensional space passes in front of another line, a gap is produced in the projection of the more distant line. The gap is produced as if an opaque halo surrounded the closer line. This method for approximate hidden-line-elimination is advantageous because explicit surface equations are not necessary. The relative depth of lines, axes, curves and lettering is easily perceived. This technique is especially suitable for the display of finite element grids, three-dimensional contour maps and ruled surfaces. When the lines or curves on a surface are closer than the gap size, the gaps produced close up to produce a complete hidden-line-elimination. A simple but efficient implementation is described which can be used in the rendering of a variety of three-dimensional situations.

C. R. Categories: 3.69, 8.1, 8.2

Key words: Haloed line effect, three-dimensional space, hidden line elimination, finite element, contour maps, ruled surfaces.

PRIOR ART:

Determining which lines should not be drawn in a three dimensional picture is called hidden line elimination. This has been of continuing interest (1-5). The essential task of hidden line elimination is to determine whether a bounded surface lies between the viewpoint and the line or point being considered for drawing, and when this occurs the point or line segment is invisible and is not drawn. Most techniques for hidden line elimination assume some surface topology. The most common assumption is to consider the scene being rendered as being approximated by triangular or rectangular patches (6, 7) or bounded planes (8, 9). There has been some work with quadric surfaces (10, 11). The floating horizon technique (12) and variants (13, 14) are popular because the format of data storage is the same as that used for other calculations and is usually appropriate for most experimental or statistical presentations. Some effort has been made toward hidden line elimination where the data is stored in two or three-dimensional raster arrays (15, 16).

BASIC TECHNIQUE:

The haloed line effect is illustrated in Figure 1 and is a common device in hand drawn technical illustration. For this effect, an imaginary region, a halo, is assumed to surround

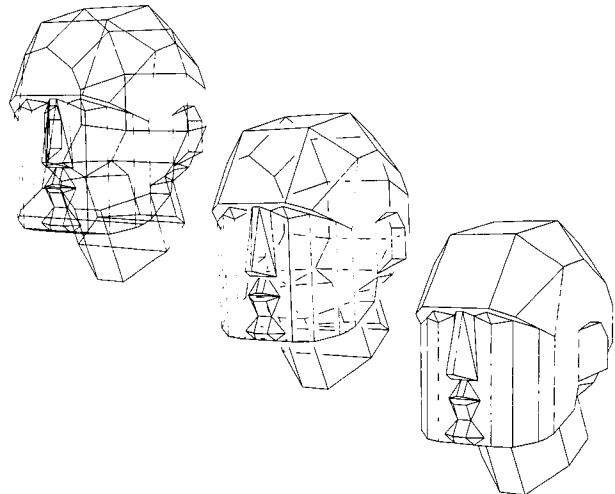


Figure 1 - These are three computer renderings of an approximate human head. Each in a different style, respectively, wire frame, haloed line effect and the usual surface dependent hidden line elimination. The bottom picture is the more realistic presentation but the middle picture which appears more like a crystal gives some knowledge of all the lines on the object. This style may be preferable for depictions of structures or finite element networks.

three-dimensional lines which can obscure any lines that pass behind the halo.

When the lines used to describe a surface form a three dimensional grid and the grid spacing is smaller than the size of the halo, then a complete hidden line' elimination results. This is illustrated in Figure 2. The haloed line effect technique offers several advantages:

- a - It is easily programmed.
- b - Variation of the halo size enables control of transparency.
- c - There are virtually no rules of topology or surface description. Real or implied surfaces are not used and need not be specified.
- d - The input to the haloed line effect processor is almost the same as might be used to draw a wire frame picture, hence this technique is easily adapted to programs or systems already making wire frame pictures.
- e - The haloing tends to accentuate surface contours and the dark and the light pattern of ruled surfaces.
- f - This method is tolerant of modelling and computational error in the original data.
- g - Memory requirements are minimized because the space usually required to store surface parameters and boundaries is not necessary.

For satisfactory results, the haloed line effect can not be rigorously implemented. For example lines parallel or nearly parallel to a line with a halo ought not to be obscured by this halo otherwise the apparent contour of objects will be destroyed. The result of this compromise is that small line segments may, almost at random, be left on a rendering as illustrated in Figure 3. These may be eliminated by additional processing but this extra work may subvert the advantages of the haloed line effect. Another compromise with rigorous implementation is that an error tolerance must be used when one line passes behind another line. Accumulated calculation errors may cause two lines that intersect in three-dimensional space to halo erroneously. Also, if a rigorous determination of halo gapping based upon imaginary haloes is attempted, artificial haloes would have to be created which would be a major undertaking and would not be very useful. It is the approximate solution to this problem which is of value.

DETAILS OF IMPLEMENTATION:

There are many possible programming tactics suitable for the haloed line effect. The optimum code would have to take into account the display technology, the preferred application language, the available storage space and the preferred data storage media. The procedure we have devised is satisfactory as implemented in FORTRAN, operating on a VM/168 computer with pictures being drawn on varied storage scopes, printer plotters and a photo composer.

The program was written to be used where the location of

points on the display and their relative or absolute depth could be determined by any three-dimensional projection scheme. There are three entry points. The first initializes the line count to zero; the second stores the end points of a line; and the third signals that the last line has been stored and starts the haloing processing and generation of drawing

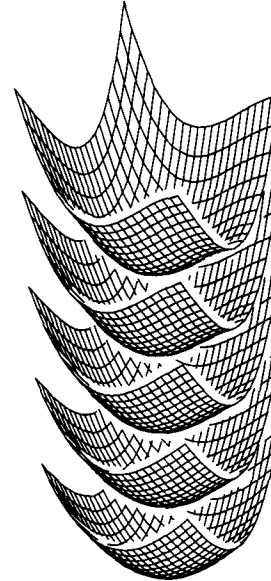


Figure 2 - Five nested algebraic surfaces rendered with the haloed line effect. While some haloing is inconsistent, the overall effect is vivid. This picture demonstrates that the haloed line effect probably cannot be perfectly programmed. Efforts to reduce penetration of haloes by short lines cause other errors or will require topological knowledge which obviate the value of this technique.

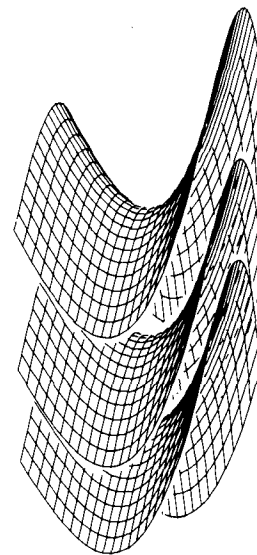


Figure 3 - A picture of three nested algebraic surfaces. Lines parallel to closer lines can penetrate through the haloes of the closer lines.

commands. The best way to store lines is in picture plane display coordinates $DX(K,L)$, $DY(K,L)$, and some measure of depth $DT(K,L)$. Where L is the line count and $DX(1,L)$, $DY(1,L)$, and $DT(1,L)$ are the coordinates of the end point with the greater DY value. K is 2 for the other end of the line. Some typical lines as stored are shown in Figure 4. The halving procedure is as follows:

1. Assume some halo gap size G . Assume some depth tolerance TOL .
2. Extend each line at both ends a distance G on the picture plane. This is illustrated in Figure 5, and will allow a halo gap at the ends of the lines.
3. Take each line one at a time. This is line I .
4. Set a counter NE to zero. NE is the number of halo edges the line I crosses. This must be an even number equal to twice the number of lines that pass in front of line I that have haloes that influence the visibility of line I .
5. Take each other line in the scene one at a time. This is line J . For intersection testing on the picture plane use the extended length of line J . This will help preserve the continuity of haloes around a surface. Some errors may be caused in open structural drawings such as in Figure 1, but these will not be serious.
6. If the $DY(2,J)$ coordinate of line J are greater than $DY(1,I)$ go back to step 5.
7. If the $DY(1,J)$ coordinate of line J is less than $DY(2,I)$ go back to step 5.
8. If both DX coordinates of line J are smaller than the DX coordinates of line I go back to step 5.
9. If both DX coordinates of line J are greater than the DX coordinates of line I go back to step 5.
10. If both DT coordinates of line J are greater than the DT coordinates of line I go back to step 5.
11. Test to determine if the projection of line J crosses the projection of line I . If the line projections do not cross go back to step 5.
12. Calculate the exact location on each line, I and J , where they cross. Calculate the relative depth of lines I and J at this crossing point. If line J is not at least TOL distance in front of I go back to step 5. The filtering operations of steps 6 thru 12 which reduce the average effort required to determine whether two lines intersect are shown in Figure 6.
13. The halo of line J can influence the visibility of line I . Increment $NE=NE+2$. Store the distance from the starting point of line I to where it passes into the halo of line J as $TE(NE-1)$. Store the distance from the starting point of line I to where it passes from behind the halo of line J as $TE(NE)$. Set visibility state markers $IGQ(NE-1)=-1$ and $IGQ(NE)=1$. The halo around line J can be

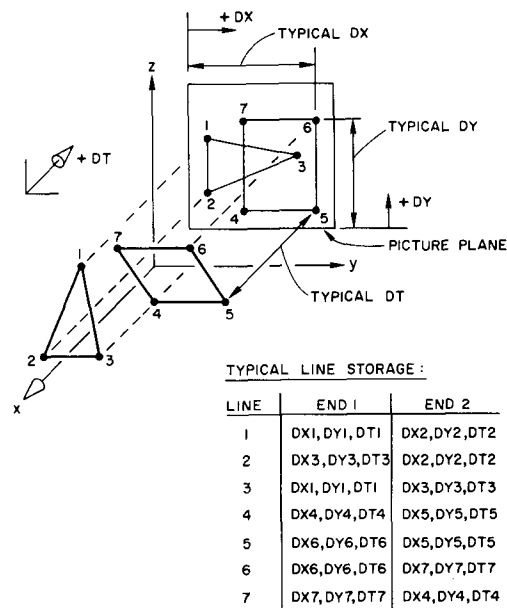


Figure 4 - An oblique drawing of the projection of a simple scene upon the picture plane and the preferred listing of data.

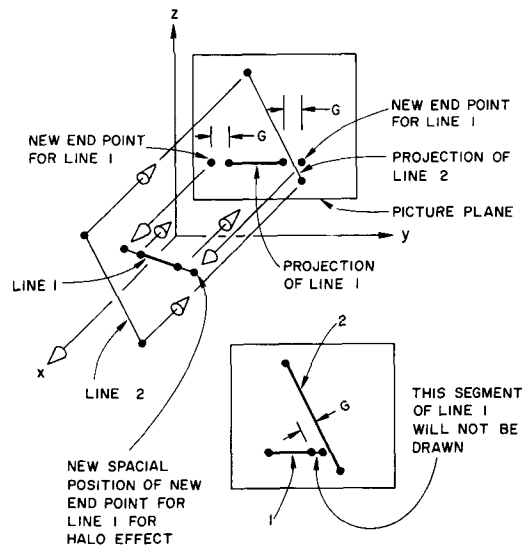


Figure 5 - An oblique drawing of the projection of two lines onto the picture plane. Line 1 which is being tested for gapping must be extended slightly as shown so as to detect that it penetrates the halo of line 2.

approximated economically by two lines parallel to line J and two circular arcs.

14. If NE for line I is zero, just draw line I and go back to step 2. If NE is larger than 2 sort $TE(NE)$ and the markers $IGQ(NE)$ on the basis of TE .
15. Sum up the running measure of how many haloes obscure the line I along its length by adding up the values of IGQ along its length. Only those sections of line I

where the sum of IGQ is zero are visible and should be drawn. Any segment of line I where the sum of IGQ is less than zero are hidden by one or more haloes and should not be drawn. This is illustrated in Figure 7.

16. Draw only the visible segments of line I that occur within its original length.
17. Go back to step 2 until all the lines in the picture have been processed.

The sequence of testing described above has been found to be optimal for pictures similar to Figure 2. For different types of pictures other techniques may be better. This procedure is simple and can be easily programmed with some simple plane geometry functions. Figure 8 demonstrates some particularly useful capabilities. Specifically the haloing of lettering and axis markings. Figure 9 thru 12 are additional demonstrations of the haloed line effect. All illustrations were produced with the same haloing program.

Some experimental pictures have been made where the gap size was made to vary with the difference in depth of lines, but these were confusing. Apparently the halo gap size should be constant.

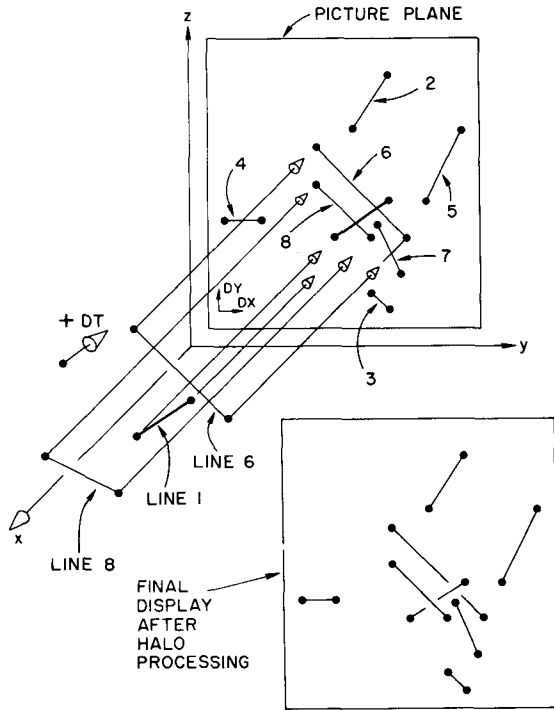


Figure 6 - The various filtering tests to determine when a line can halo another line. Line 1 is under test for haloing. For ease of explanation we can ignore the small extensions of line 1. Line 2 is obviously above line 1. Line 3 is obviously below line 1. Line 4 is obviously too far to the left. Line 5 is obviously too far to the right. Line 6 is behind line 1. Line 7 does not cross line 1. Only line 8 can cause a halo gap in line 1.

EXECUTION TIME

The execution time of a hidden line elimination program is determined by several factors. The primary influence is the geometrical strategy which takes into account the mathematical character of the objects or scene to be rendered and the style in which the picture is to be made. A secondary influence is the specific embodiment of the geometrical strategy in a computer program.

Hidden line elimination programs can also be tuned to take advantage of some geometric properties of objects being rendered and the characteristics of the processing computer or ancillary hardware. Most probably the most important factor in the speed of a hidden line elimination program is the sophistication and dedication of the programmer. For example, the use of optimal sorting algorithms, the measurement and minimization of program statement execution counts, and the use of optimizing compilers can improve speed by an order of magnitude. When directly measuring a hidden line elimination program, the design and orientation of the test shape can bias the results. This has been observed to be especially true when measuring the execution time of haloed line effect pictures. When generating a typical algebraic surface, similar to that shown in Figure 3, the execution time could vary considerably with a change in viewpoint. The gap size of the halo has an unpredictable effect upon execution time. Generally a larger gap size tends to increase the execution time, but some objects have been drawn when the reverse has been observed. The gap size should be decided upon aesthetically.

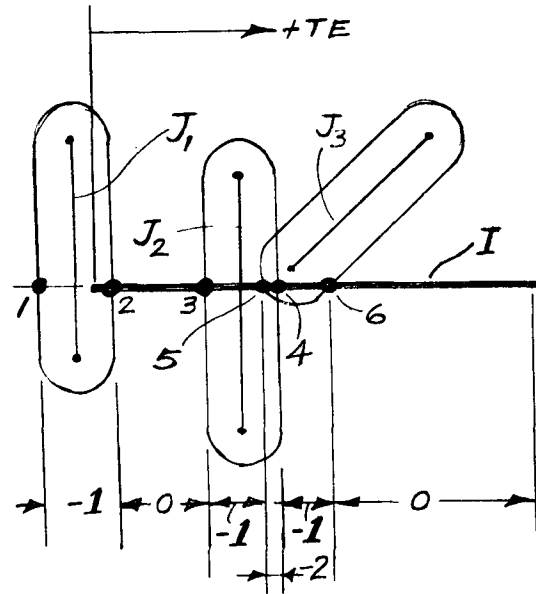
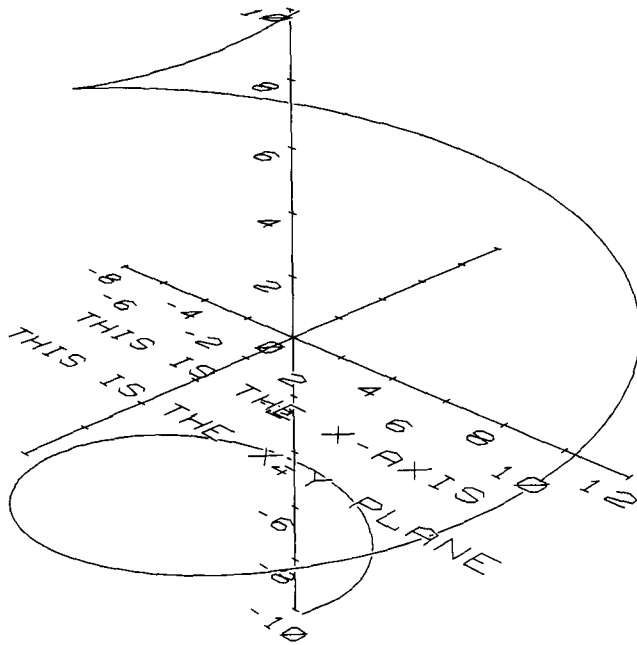


Figure 7 - A typical situation where the value of IGQ is used to determine the visible segments of line I. Notice that the halo gap of line J3 is subject to an end effect. The numbers along line I are the original TE index. The dimensions are the running IGQ counts which are how many haloes obscure a line on a particular segment.

Hidden line elimination programs are usually evaluated upon the dependence of execution time on picture complexity. The most common measures of picture complexity being the line count, or polygonal surface count. Obviously for the haloed line effect, only the line count, N , is meaningful. For algebraic surfaces such as Figures 2, 3, 10 and 11, the execution time has been observed to be proportional to $N^{3/2}$. Rendering a picture of an algebraic surface containing 180 lines required about 1.15 seconds and a similar surface with 1,740 lines required about 26.07 seconds.



SPIRAL ON A SPHERE

Figure 8 - A simple picture containing lettering and axis markings. The haloed line effect can be used economically with these graphic elements.

POTENTIAL ALGORITHMIC IMPROVEMENTS:

The techniques described in this paper for implementing the haloed line effect are early experimental results and are a compromise between complexity and an acceptable quality of graphics. The major calculation task is to detect which lines cross and considerable improvement in this regard is to be expected if the lines in the picture would be subdivided into smaller groups by clipping (17, 18).

The halo entrance and exit marker technique used in steps 13 and 14 are similar to the concept of quantitative invisibility described in a previous paper on polyhedral hidden line elimination (19). This should suggest that there may be many ways in which the haloed line effect can augment or benefit from the usual hidden line elimination techniques.

One assumption of the implementation is that all the lines used for halo obstruction will be drawn. Additional lines can be added to a scene, such as patterns on planes, which are not intended to be drawn but which can obscure other lines. The result can be an approximation to complete hidden line elimination.

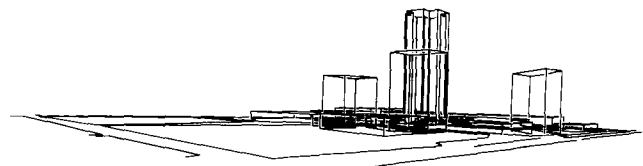
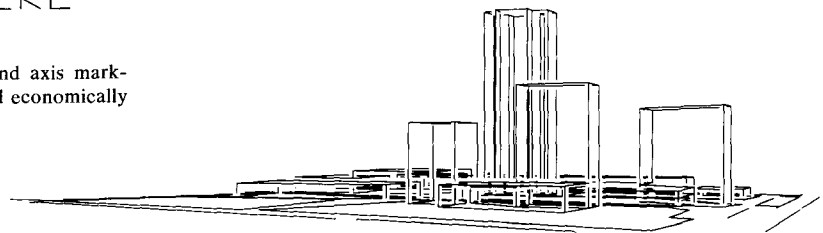
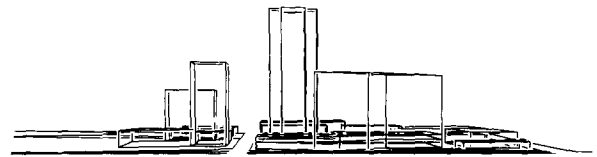


Figure 9 - A series of architectural drawings prepared using the haloed line effect. The data consisted only of lines in a three-dimensional space and were not organized in any other way.

It may be interesting to compare the haloed line effect with other hidden line elimination techniques, but this is like comparing apples with oranges as the character of pictures and processing are so different. Referring to Figure 1, the wire frame picture took about .2 seconds of CPU time, the haloed line effect picture required about .69 seconds and the complete hidden line elimination required about 2.57 seconds. In addition, the complete hidden line technique picture needed about 3 seconds set up time for the calculation of surface equations, line equations and the usual organizational data structuring which the haloed line effect does not need. While these measurements are typical of the programs used, this should not imply that the haloed line effect is particularly faster than the usual hidden line elimination techniques. Most probably the floating horizon methods of Wright (12) have the same performance characteristics for algebraic surfaces.

CONCLUDING REMARKS:

The most important aspect of the haloed line effect is the simplicity and geometrical independence of the technique. Scientists and engineers, inexperienced in the special techniques of three-dimensional computer graphics, are easily taught to use the haloing program because there are virtually no rules of topology or organization of the input line set.

We would like to thank Cliff Nass for integrating the haloed line effect into the United States Military Academy Graphics Compatibility System (GCS) and for many helpful ideas for improving the quality of the pictures. Thanks are due to Robert O'Hara for his architectural data used for Figure 9.

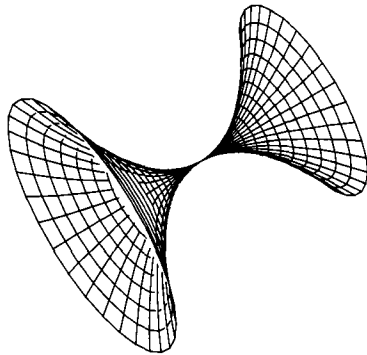


Figure 10 - An algebraic surface.

REFERENCES:

1. J. Encarnacao, "Survey of and New Solutions for the Hidden-Line Problem", *PROCEEDING OF SYMPOSIUM ON COMPUTER GRAPHICS*, Delft, Netherlands, October 1970, pp. 26-28.
2. I. E. Sutherland, R. F. Sproull and R. A. Schumacker, "A Characterization of Ten Hidden-Surface Algorithms", *COMPUTING SURVEYS*, Vol. 6, No. 1, March 1974, pp. 1-55.
3. Various techniques are discussed in Chapter 5 of Wolfgang K. Giloi, *INTERACTIVE COMPUTER GRAPHICS*, Prentice-Hall Inc, Englewood Cliffs, N. J., 1978.
4. Various techniques are discussed in Chapter 14 of William M. Newman and Robert F. Sproull, *PRINCIPLES OF INTERACTIVE COMPUTER GRAPHICS*, McGraw-Hill Book Company, New York, 1973.
5. J. G. Griffiths, "Bibliography of Hidden-Line and Hidden Surface Algorithms", *COMPUTER-AIDED DESIGN*, Vol. 10, No. 3, May 1978, pp. 203-206.
6. C. Wylie, G. W. Romney, D. C. Evans, and A. Erdahl, "Halftone Perspective Drawings by Computer", *PROC. AFIPS FJCC 1976*, Vol. 31, 49.
7. A. Appel, "Hidden Line Elimination for Complex Surfaces", *IBM TECHNICAL DISCLOSURE BULLETIN*, Vol. 18, No. 11, April 1976, pp. 3873-3876.
8. Bruce G. Baumgart, "A Polyhedron Representation for Computer Vision", *PROCEEDINGS AFIPS NATIONAL COMPUTER CONFERENCE 1975*, Vol. 44, pp. 589-596.
9. L. I. Lieberman, M. A. Wesley, M. A. Lavin, "A GEOMETRIC MODELLING SYSTEM FOR AUTOMATED MECHANICAL ASSEMBLY", IBM Research Report RC 7089, April 25, 1978.
10. Ruth A. Weiss, "BE VISION: a package of IBM 7090 Fortran programs to draw orthographic views of combinations of plane and quadratic surfaces", *JOURNAL OF THE ACM*, Vol. 13, No. 2, April 1966, pp. 194-204.
11. Peter Woon and Herbert Freeman, "A Procedure for Generating Visible Line Projections of Solids Bounded by Quadric Surfaces", *PROCEEDINGS IFIP CONGRESS INFORMATION PROCESSING*, North Holland, 1971, pp. 1120-1125.
12. Thomas J. Wright, "A Two-Spaced Solution to the Hidden Line Problem for Plotting Functions of Two Variables", *IEEE TRANS. ON COMPUTERS*, Vol. c-22, No. 1, January 1973, pp. 28-33.
13. A. Appel, S. W. Handelman and A. J. Stein, "Hidden Line Elimination for Semi-Convex Slices", *IBM TECHNICAL DISCLOSURE BULLETIN*, Vol. 20, No. 7, December 1977, pp. 2917-2920.
14. Wm. Randolph Franklin and Harry R. Lewis, "3-D Graphic Display of Discrete Spatial Data by Prism Maps", *Computer Graphics*, Vol. 12, No. 3, August 1978, Quarterly Report of SIGGRAPH-ACM. pp. 70-75.
15. Thomas Wright, "Visible Surface Plotting Program", *COMMUNICATIONS OF THE ACM*, Vol. 17, No. 3, March 1974, pp. 152-155.
16. A. Appel and A. J. Stein, "Cellular Automata for Hidden-Line Elimination", *IBM TECHNICAL DISCLOSURE BULLETIN*, Vol. 17, No. 2, July 1974, pp. 586-589.
17. I. E. Sutherland and R. F. Sproull, "A Clipping Divider," *Proc. AFIPS FJCC 1968 Conference*, Vol. 33, Part I, pp. 765-776.
18. James F. Blinn and Martin E. Newell, "Clipping using Homogeneous Coordinates", *Computer Graphics*, Quarterly Report of SIGGRAPH-ACM, Vol. 12, No. 3, August 1978, pp. 245-251.
19. A. Appel, "The Notion of Quantitative Invisibility and the Machine Rendering of Solids", *Proc. ACM National Conference 1967*, pp. 387-393.

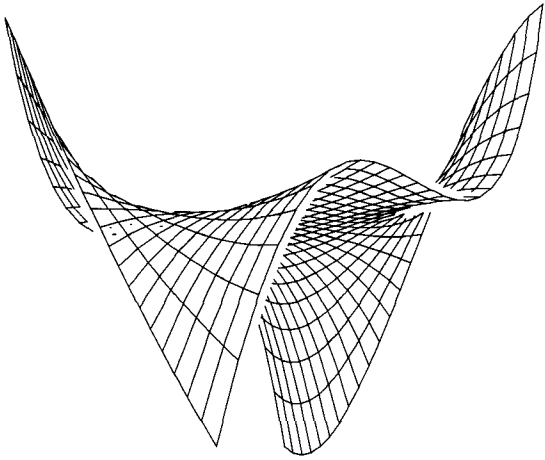


Figure 11 - An algebraic surface.

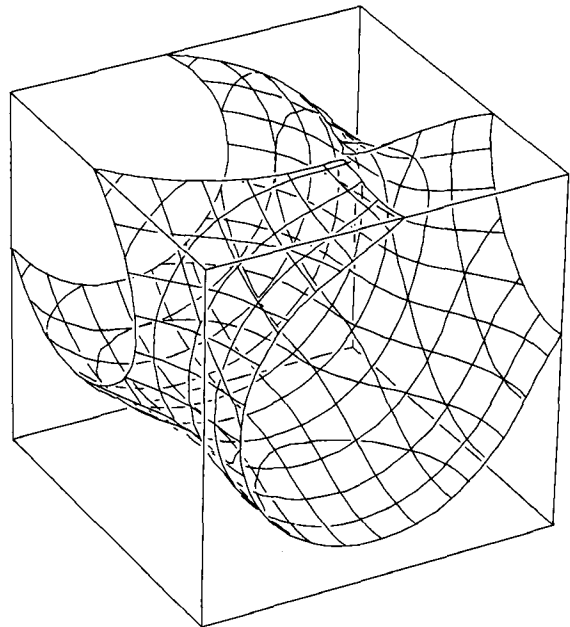
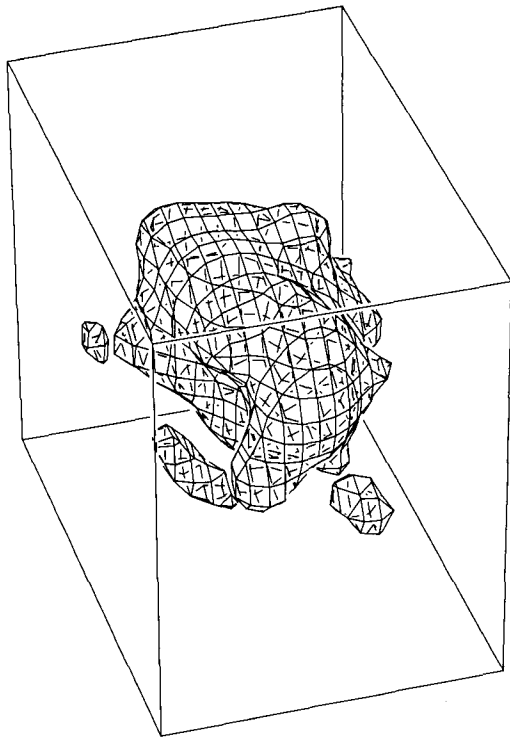
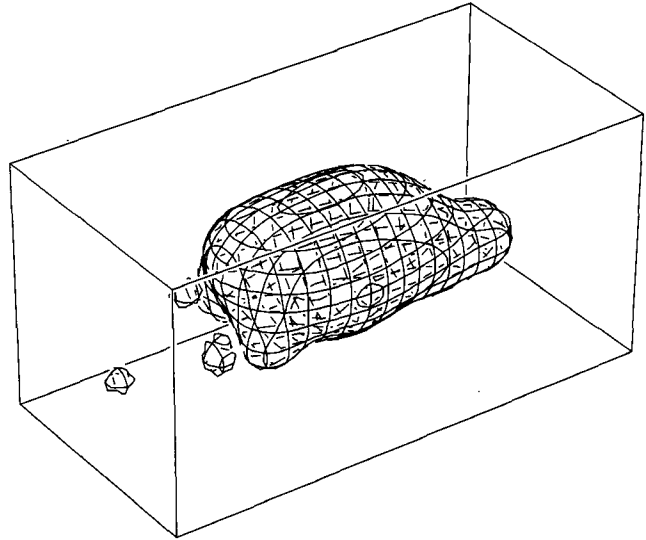


Figure 12 - A series of pictures based upon statistical data. All haloed line effect pictures in this paper were processed by the same program.