A new approach to the 'hidden line' problem

C. B. Jones*

BEA Management Services, West London Air Terminal, Cromwell Road, London, SW7

This paper presents an approach to hidden line removal which relies on three-dimensional objects being described in terms of a series of inter-connected spatial cells. Avenues of sight through the openings between cells are explored in the process of generating display information for a perspective picture from a given viewpoint. An implementation on a small computer with a graphics display is described, and the performance of the program is illustrated by photographs of some views which were generated in a few seconds. Data preparation for the examples involved the definition of the space surrounding the objects by means of a special purpose data structure.

(Received June 1970)

In order to comprehend configurations of physical objects, it is frequently necessary to view them from a variety of angles and in many computer applications in which such objects have to be modelled there is a need to simulate this process of observation. Ideally this would be achieved by the display of tonally rendered images to each eye, matching those which would be seen from any selected viewpoint. An acceptable effect can nevertheless be conveyed by a series of perspective line drawings which are suited to output on a standard digital CRT display.

The critical problem is the detection and removal of 'hidden lines', which can involve so much computation that the time required to generate each view becomes excessive. This is particularly so in the case of objects composed of curved surfaces (Weiss, 1966). A number of solutions has been proposed to deal with the more manageable problem of objects bounded by plane surfaces.

The techniques they use may be classified as follows:†

1. More or less exhaustive comparison of each polygonal face with each line to determine overlap. (Roberts, 1963; Loutrel, 1967).
2. Scanning the objects with a cutting plane passing through the viewpoint, on which intersections are determined, and thus building up a raster image (Appel, 1968; Romney, Watkins, and Evans, 1968).
3. Repeatedly subdividing the picture into quadrants until a portion is found which may be processed simply (Warnock, 1969).

A different approach to the generation of line drawings of plane-surfaced objects will now be discussed, which is based on the way in which the objects are described.

The space they occupy is considered, not as a void containing a number of discrete objects, but as a set of cells which are bounded partly by faces of the original objects, and partly by artificially introduced transparent faces. This distinction is illustrated in **Fig. 1,** which for ease of representation is a 2D case.

Cell faces may be opaque, or may have any number of transparent openings. Each cell is required to be a convex polyhedron, i.e., a line between any two points on its surface lies entirely within the cell, while each opening is required to be a convex polygon, and to be shared by two adjacent cells. This spatial compartmentation can be achieved in a considerable number of ways, any of which may be adopted, whatever objects are present. It is particularly easy to apply to the inside of a building, and this example will be used to introduce the action of the algorithm.

Fig. 2 is a small house plan: the rooms are equivalent to the cells mentioned above; wall thickness is neglected. An observer standing in the living room will see without obstruction the walls of this room and the two openings into the kitchen and dining room. Through the first opening he can see a corner of the kitchen, the remainder being masked off by the outline of the door. Similarly with the dining room, but here a further opening, the service hatch, gives another view into the kitchen, masked this time by the combined outlines of the two openings.

In a more complex case, an avenue of sight might extend through a large number of cells, the view of each successive

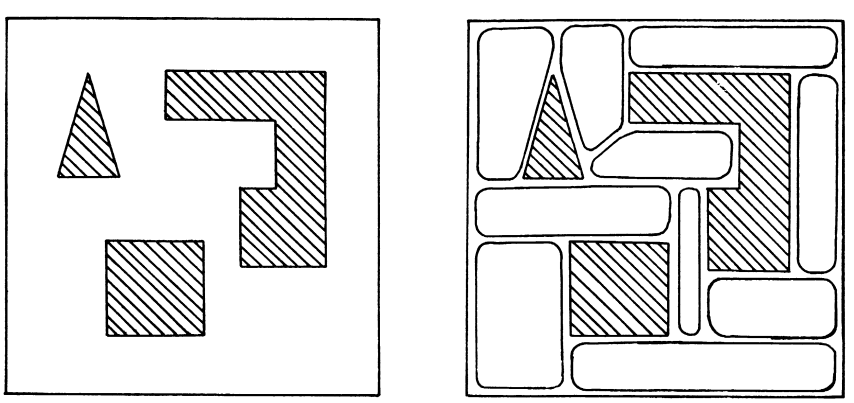

Fig. 1. Division of the space surrounding objects into cells

*While at the Centre for Computing and Automation, Imperial College, London. (The work described was supported by a Science Research Council Grant No. B/SR/2071, 'Computer Processing of Three Dimensional Shapes'.)

†Based on a survey by A. R. Forrest at the Cambridge Computer Laboratory (C.A.D. Group Document No. 19.).

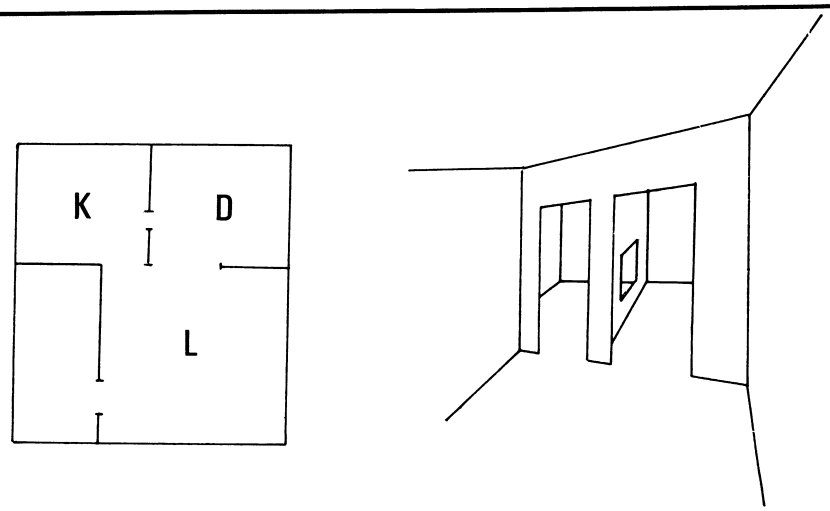

Fig. 2. House plan and perspective view. Kitchen (K), Dining Room (D), and Living Room (L).

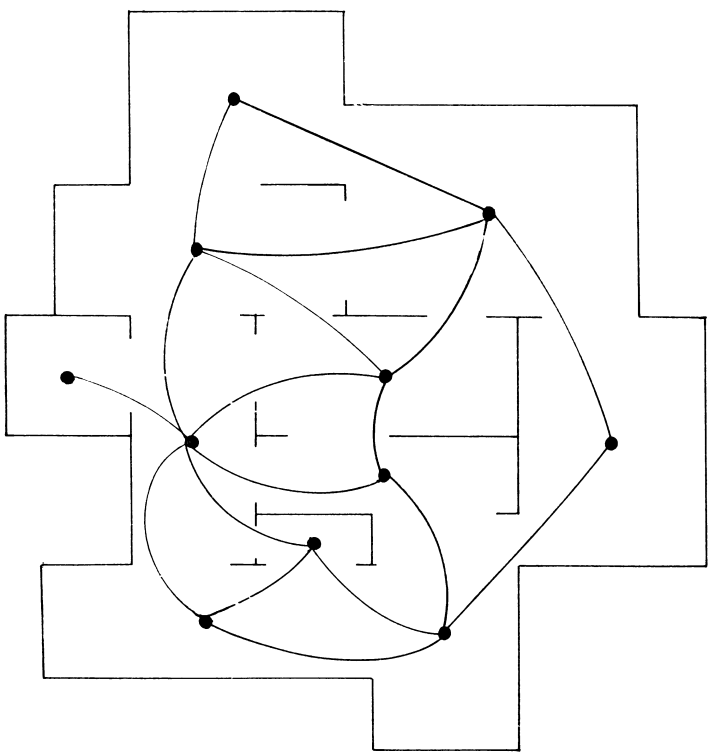

Fig. 3. The linear graph corresponding to a configuration of cells

cell being cumulatively masked by each opening encountered between it and the eye. An avenue of sight will terminate when a cell is encountered, none of whose openings is visible through the resultant mask. The structure of a configuration of cells may be represented by a linear graph, in which each node represents a cell and each arc an opening (**Fig. 3.**). The avenues of sight (**Fig. 4.**) then form a directed subgraph (**Fig. 5.**) in which a number of paths branch out from the initial node representing the cell occupied by the viewpoint. Each path to a node indicates that an area within the corresponding cell is visible through the sequence of openings corresponding to the arcs which lie on the path.

A recursive search procedure is used to explore the avenues of sight and thereby to determine the appearance of the picture. The algorithm pursues a path until it terminates, then backtracks to look for paths from an earlier node. On arriving at a node, (equivalent to gaining visual access to a cell) there will be a representation of the current mask, which is defined by the outline of the area of the picture plane through which the cell can still be viewed.

All the lines on the inner faces and edges of the cell which are not part of openings are dealt with by finding their projections on the picture plane, and then clipping them according to the mask. Since the cells are convex, there can be no obscuration of lines apart from that due to masking, and the clipped lines may be added to the display file.

Next, the first opening of the cell is examined, and its bounding lines treated as above. A new mask is formed from the intersection of the current mask and the boundary of the opening projected on to the picture plane. If the intersection is empty, then this opening is not visible, and needs no further attention. Otherwise the opening will give access into an adjacent cell. In this case the current mask is stored in a stack together with some information identifying the current cell and opening. The algorithm now descends a level: the new mask becoming the current mask; the new cell associated with the opening becoming the current cell; and the above procedure is applied again.

The openings were required to be convex polygons, and the perspective transformation on to the picture plane preserves this convexity. Since the overlap of two convex polygons is also convex, it follows that a mask outline is always convex. This permits a straightforward clipping routine to be written: a candidate line can only intersect the mask outline twice, and the visible segment, if any, can be readily determined.

Eventually a cell will be encountered in which no openings are visible through the current mask. This becomes more likely as more openings are combined into the mask, reducing its available viewing area and reducing in consequence the likelihood of seeing further openings. Termination is guaranteed, however, by describing in the source data a large opaque box surrounding the objects under consideration. On termination, the algorithm backs up a level, and restores attention to the cell through an opening of which the terminal cell was accessed. This cell, and the associated mask will be found at the bottom of the stack. If it has openings yet to be examined, the algorithm proceeds as before, otherwise, unless the cell is the initial cell, it ascends levels until a cell is found which does have unexamined openings. The process is complete when all openings of the initial cell have been examined: the stack will then be empty.

The program strategy outlined above has been implemented on a DEC PDP-7 computer with 8192 18-bit words of core storage and a cycle time of 1·75 microseconds. Display file is generated to drive a DEC type 340 CRT display. With a view to attaining the most efficient code, programming was undertaken in the low level assembly language of the machine. Certain restrictions were imposed to facilitate programming and to conserve the limited core memory available, without sacrificing the essential features of the approach. The most

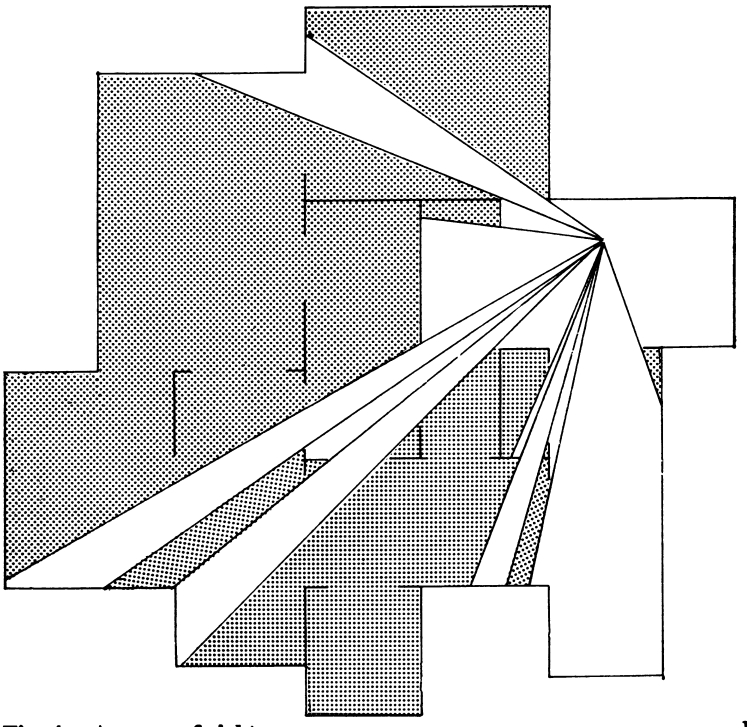

Fig. 4. Avenues of sight

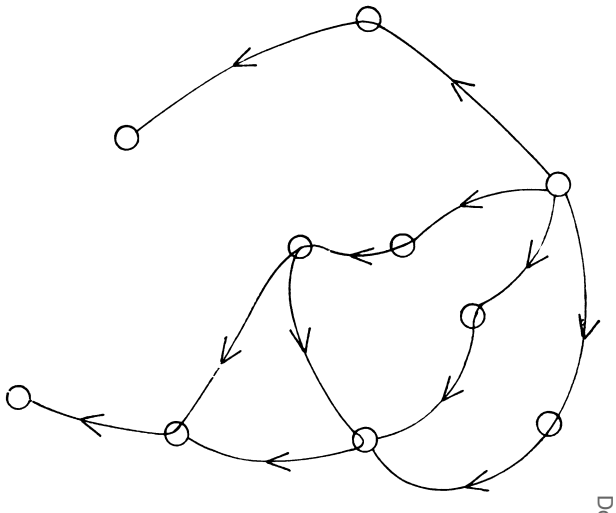

Fig. 5. Sub-graph representing avenues of sight

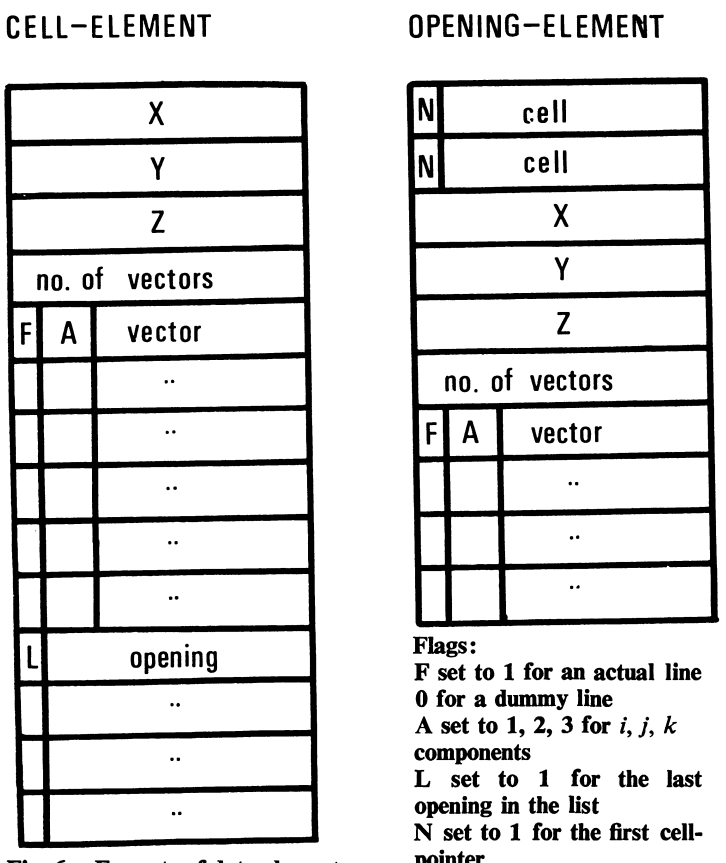

Fig. 6. Formats of data elements

serious of these prohibits the viewpoint being moved at run time out of one particular cell specified in the data as the initial cell. Ideally there would be complete freedom of movement of the viewpoint within the space surrounding the objects and the first task of the program would be to identify the cell occupied by the viewpoint. This facility was omitted in order to simplify data preparation by reducing the amount of information required to specify a cell: the coefficients in the equations of the bounding planes of the cell not then being necessary. A reasonable variety of views may still be obtained by making the initial cell an extensive zone in front of all the objects.

The viewpoint is initially assumed to be at the origin of the coordinate system used to locate the objects. Only orthogonal lines parallel to one of the axes may be included in object descriptions, the range of which is thereby limited to rectangular parallelepipeds. A translation may be applied at run time along each axis to change the position of the objects relative to the viewpoint. Rotation has been omitted: a rotation of the objects about the viewpoint would not alter the visibility of any line but would only change the proportions of the picture.

The perspective transformation is effected by dividing each of the x and y coordinates of a point by its z coordinate and multiplying by a scale factor. This relies on the z coordinate being positive, as it will be if the initial cell is correctly defined.

The data structure used to represent a configuration of objects in core employs two types of element: those specifying cells (cell-elements) and those specifying openings (openings-elements). The formats of these elements are illustrated in **Fig. 6.** The first three words of a cell-element contain the coordinates of an arbitrarily chosen origin point in the cell. There follows a sequence of vectors which describes all those lines in the cell which are not part of the boundary of an opening.

Each vector records the magnitude and direction of a line to be positioned at the end of the previous line, or, in the case of the first vector, at the cell origin. Vectors are flagged according as they represent actual lines in the cell or dummy lines which simply reposition the endpoint for the succeeding line.

Finally, the openings in the cell are enumerated by a sequence of addresses pointing to opening-elements. The first two words of an opening-element point back to the heads of the two cell-elements which share the opening. These are followed by the origin and the sequence of vectors describing the lines bounding the opening in the same manner as the lines in a cell.

The first step in coding an object description is to prepare diagrams depicting the individual cells and their openings (**Fig. 7**). A symbolic source tape prepared with the aid of the diagram is assembled and loaded into core together with the program.

The view which is displayed may be altered by using eight sense switches which, whilst depressed, regularly increment or

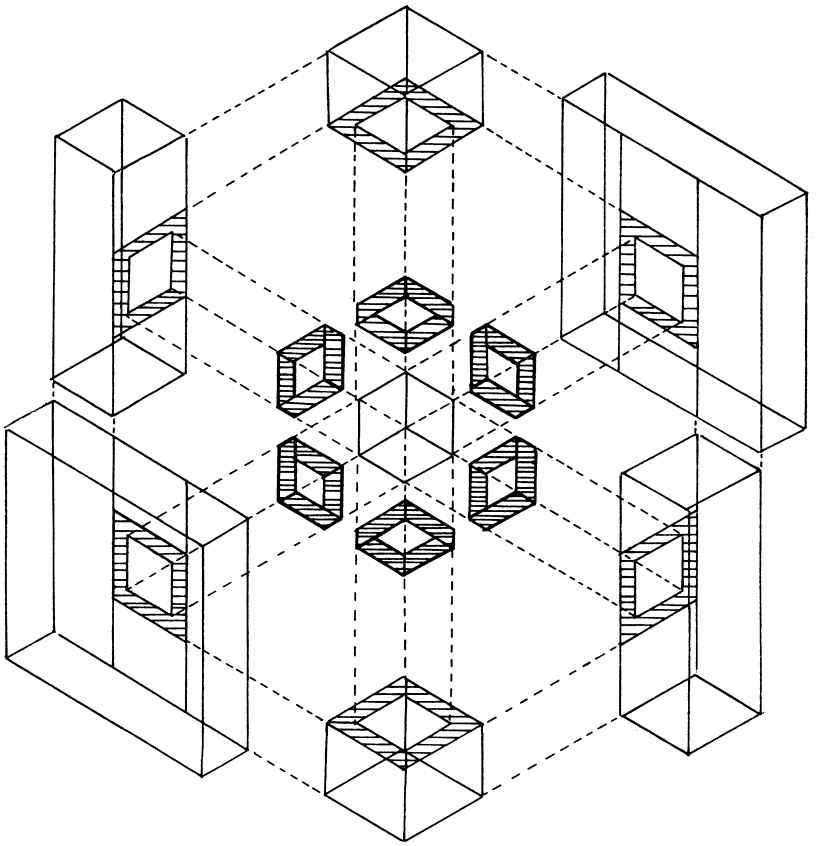

Fig. 7. Cells used in the spatial description of the object represented in Figs. 8 and 9

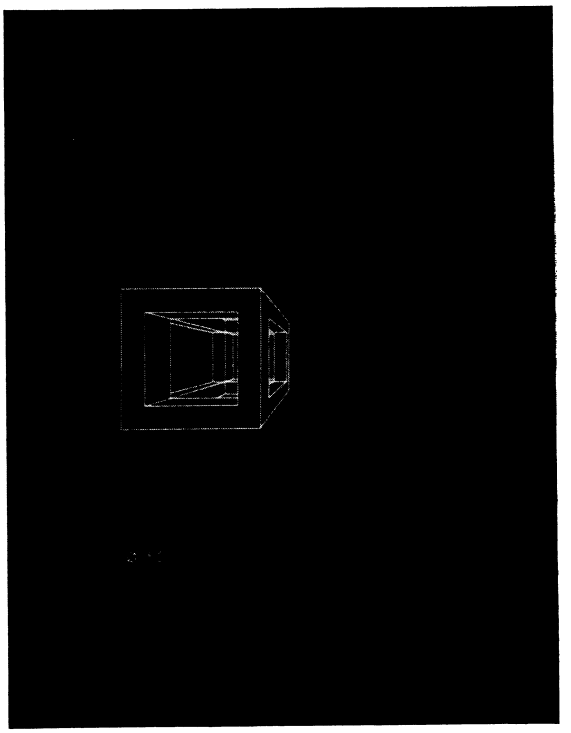

Fig. 8.

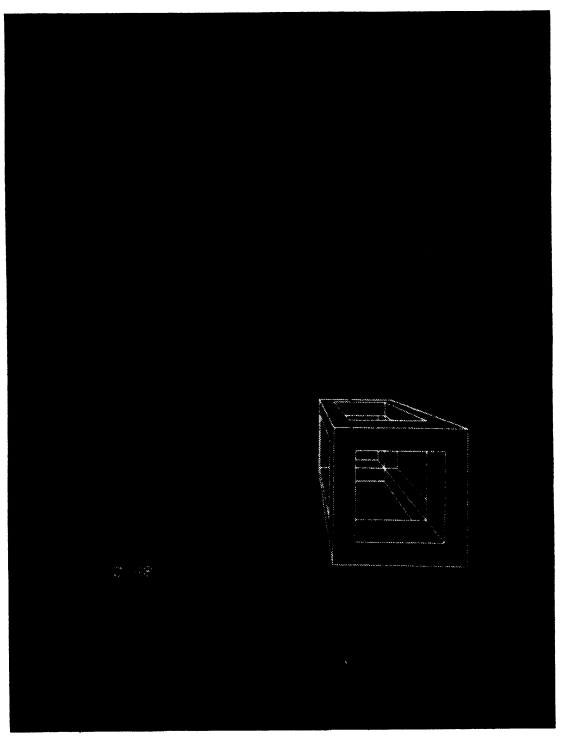

Fig. 9.

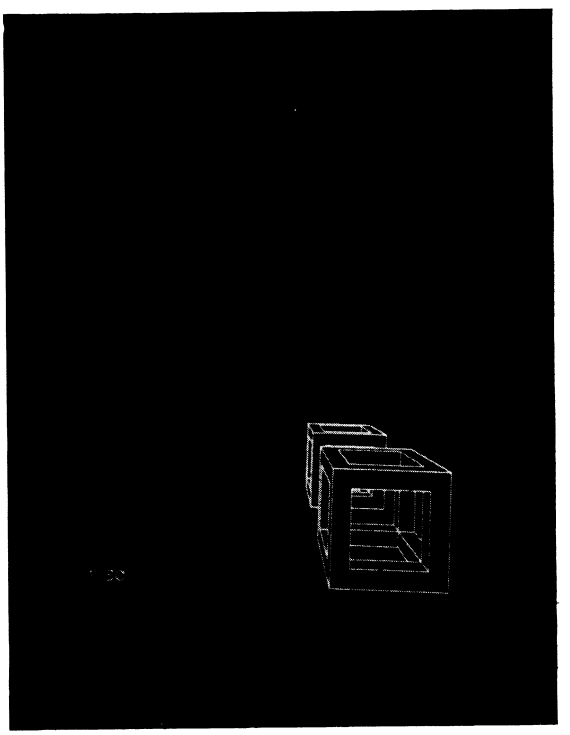

Fig. 10.

decrement registers containing translations to be applied to the objects along the x, y and z-axes, and a scale factor. These give effects of moving left, right, up, down, in and out, relative to the objects, and of increasing or decreasing the scale of the picture.

When the computation of a view is complete, the new display file replaces the current display file, the viewing parameters are updated according to the recorded use of the sense switches, and generation of the next view begins. Thus the appearance of the picture responds to the viewer's control after a delay which depends on the computing time for a view. This, in turn, depends on the complexity of the objects, and the particular viewpoint, but has been of the order of a few seconds for some objects which have been tried. **Figs. 8-13** are photographs of the display and show the computing time in seconds for each view. It has not been possible to achieve the rate of presentation of successive views which would give the illusion of continuous motion of the image: this would have to be about the cine-

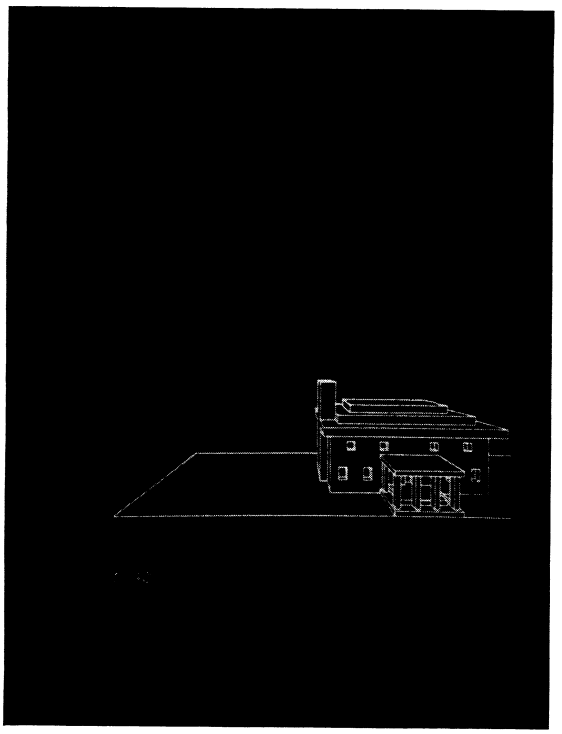

Fig. 11.

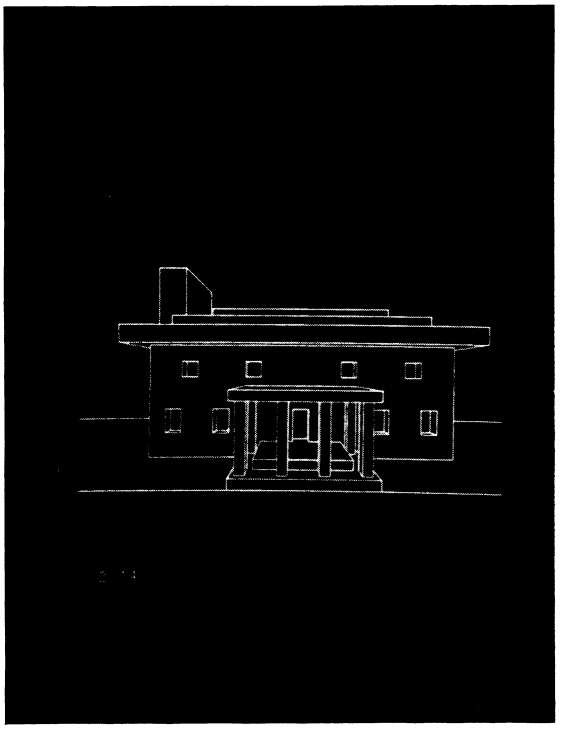

Fig. 12.

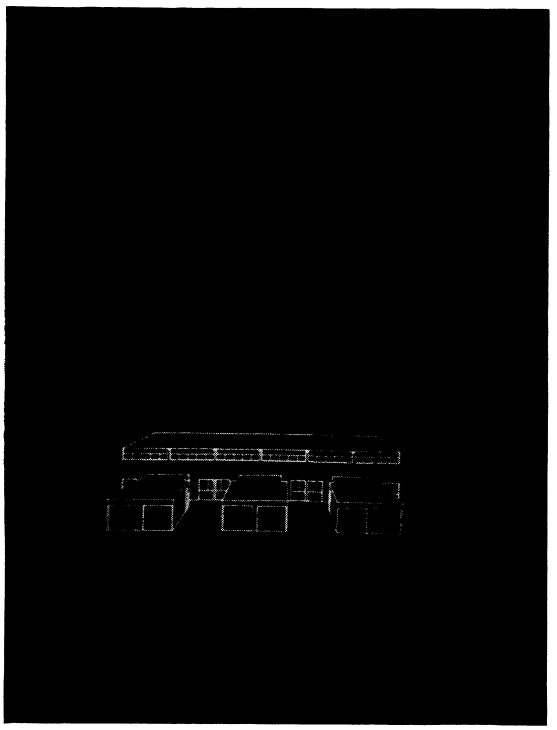

Fig. 13.

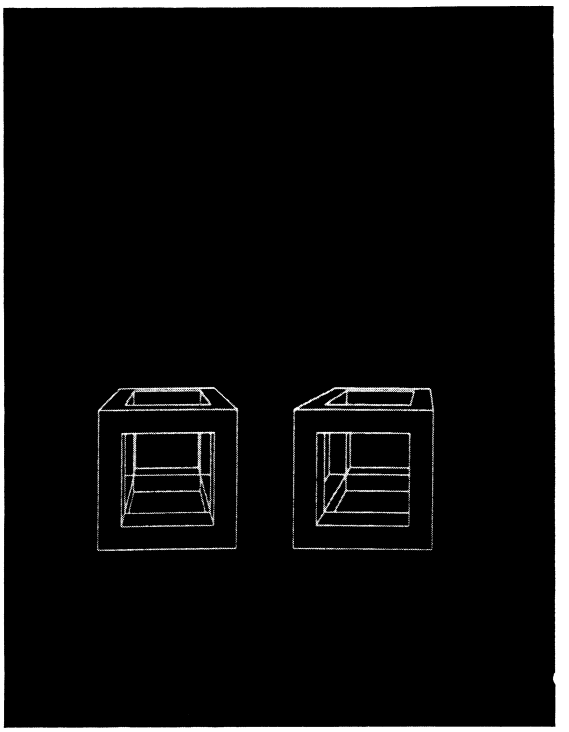

Fig. 14.

matographic frame rate (for silent films) of sixteen frames per second. An alternative display scheme was tried, in which new lines were added to the displayed picture immediately they were processed, rather than awaiting picture completion. Although illuminating the progress of the algorithm, this tended to distract attention from the actual appearance of the objects under consideration.

It is difficult to formulate any general relationship between computing time and object complexity, since an exhaustive set of tests for visibility of lines is avoided. The tests performed in a particular instance will depend on what can be seen from the viewpoint. The lines within a given cell are only examined if there is a line of sight from the viewpoint into that cell. Perhaps the most attractive feature of the program is that cells to which access is not gained in this way are never even considered. For this reason, best results are obtained when opaque planes predominate over transparent openings, and limit the extent of avenues of sight.

A conclusive comparison of different methods of hidden line removal by the criterion of speed would need to be based on their performance on a variety of objects from a variety of viewpoints, and would need to allow for machine dependent features. No such comparison will be undertaken here, in the absence of an accepted set of test objects. The object appearing

in Figs. 11 and 12 was selected, however, to permit some degree of comparison with another method which it has been used to illustrate (Roberts, 1963). It is believed to have taken several minutes to process on the then dedicated TX-2 machine at MIT Lincoln Laboratory. It should be remembered that the PDP-7 program, designed to test the feasibility of the general approach, omits features which are present in other methods, and that the data preparation was a quite time-consuming exercise.

The approach described above has achieved fairly rapid view generation by investing in some pre-processing of the objects which is valid for all views, namely the compartmentation of space into convex cells. As explained, this is currently the task of the user who encodes the object description, but it would be highly desirable for it to be performed by program.

The algorithm for doing this deserves further attention: it is not clear whether, in the general case of a set of unconnected objects, an exhaustive set of comparisons would be necessary, to establish the spatial cells.

It is felt that architectural subjects could be dealt with more straight-forwardly, since the space enclosed by buildings, and between them, is just as meaningful as the solid building fabric. Given a suitable input technique, a designer might be able to specify these spaces directly.

By a simple modification, the program can be made to create two views from slightly displaced viewpoints and to present these simultaneously on the display screen as a stereoscopic pair (**Fig. 14**). This can be fused into a three-dimensional image by using suitable viewing equipment: some methods which have been tried are discussed elsewhere (Ortony, 1971).

Hidden line elimination, perspective, stereoscopy, and motion of the viewpoint combine to offer a wealth of visual cues for the perception of depth, and go a considerable way towards providing the kind of man/machine interface outlined in the opening paragraph of this paper.

Acknowledgement

The author is grateful to Mr. A. Ortony for his many helpful suggestions during the preparation of the text of this paper.

References

APPEL, A. (1968). Some Techniques for Shading Machine Renderings of Solids, *AFIPS Conference Proceedings*, Vol. 32, pp. 37-45.

LOUTREL, P. P. (1967). A Solution to the 'Hidden-line' Problem for Computer-drawn Polyhedra, New York University, Dept. of Elec. Engg. Tech. Rept. 400-167.

ORTONY, A. (1971). A System for Stereo Viewing, *The Computer Journal*, Vol. 14, No. 2, pp. 140-144.

ROBERTS, L. G. (1963). Machine Perception of Three-dimensional Solids, M.I.T. Lincoln Lab. Tech. Rept. 315.

ROMNEY, G. W., WATKINS, G. S., and EVANS, D. C. (1968). Real-time Display of Computer Generated Half-tone Perceptive Pictures, Proc. IFIP 1968, E72-78.

WARNOCK, J. E. (1969). A Hidden Surface Algorithm for Computer Generated Halftone Pictures, Univ. of Utah, Tech. Rept. RADC-TR-69-249.

WEISS, R. A. (1966). BE VISION, *JACM*, Vol. 13, No. 2, pp. 194-204.