A real time hidden surface technique

P. J. Willis

School of Engineering and Applied Sciences, University of Sussex, Falmer, Brighton BN1 9QT

Most hidden surface algorithms require a considerable amount of computation for all but the simplest images. This prevents their use in real time systems where new frames may be calculated at a rate of 25 per second. The paper presents an approach suitable for fixed models, such as those used in flight simulators, where most of the time consuming computation may be performed when the model is first created. The technique is to test for surface proximity in a well defined manner, 'well spaced' surfaces requiring a simple measure of distance to determine priority, and 'closely spaced' surfaces being modified until they are 'well spaced'. This modification is only in the representation of the surface and does not affect its final appearance in the picture.

The work to be described is part of a project financed by the Science Research Council, to which grateful acknowledgement is made.

(Received June 1976)

Real time image generation presents the hardware with a considerable computational load for all but the simplest of cases, one of the most time consuming tasks being the elimination of hidden surfaces. Typical approaches to the problem rely on comparing surfaces to decide which are obscured by others, but the number of comparisons required is a rapidly increasing function of the number N of surfaces, $N \log_2 N$ or N^2 being usual. This immediately limits either the picture complexity, or the update rate of the display which results in image flicker or jerking motion.

One of the most demanding application areas of computer graphics is that of simulators for aircraft or other rapidly moving vehicles. The image to be generated is that of a realistic view from the aircraft cockpit. On the one hand the requirement for realism necessitates a detailed display, while on the other hand the display will need to be updated at least 25 times a second if an illusion of continuous motion is to be maintained. In addition the aircraft is free to move in three dimensions and the image must respond quickly to any change to the controls made by the pilot. For these and other reasons present commercial computer generated flight simulator displays are rather limited, most systems relying on a model board traversed by a television camera.

The present paper will therefore be concerned with the hidden surface problem applied to the real time generation of images representing rapidly changing views of a static model of a complexity which implies that the number of visible surfaces will be in the hundreds. Only plane surfaces will be considered.

In essence the technique is to associate with each surface to be displayed a number, called the priority of that surface, such that a suitable raster display can produce the correct image by selecting the highest priority surface at each scan point. Low priority surfaces are hidden by higher priority surfaces. Distance from the viewpoint to the object provides a guide to the priority value, but the fact that a surface extends over a range of distances introduces a certain ambiguity which the technique to be discussed resolves.

The application area is that of a flight simulation visual display system being developed at the University of Sussex. Various aspects of the project have been discussed elsewhere (Willis, 1976) and a new colour display system, well suited to the algorithm to be discussed, has been patented (Grimsdale *et al.*, 1976).

Earlier approaches

Much of the published work in the field relates to hidden line rather than hidden surface removal. Historically this relates to the use of calligraphic displays based on vector generators and cathode ray tubes in preference to scanned displays. This is adequate for many purposes such as line drawings of architectural features, but is a hindrance when image realism is a requirement. This will be discussed shortly.

Jones (1971) recognises three classifications of published techniques and adds a fourth of his own. Following his approach the list is as follows:

1. Exhaustive comparison of each surface with every line to determine overlap.
2. Scanning the model with a plane passing through the view-point. Calculation of intersections with this plane allow a raster image to be created. The method has been used by Appel (1968), Romney, Watkins and Evans (1968), and Bouknight (1970).
3. Repeatedly subdividing the picture into quadrants until a portion is found which may be processed simply. This is known as Warnock's Algorithm and is adequately discussed by Sutherland (1970).
4. Jones' own method is to define objects by the spaces between them. The interior of a house, for example, may be defined as a series of convex space cells loosely corresponding to the rooms or convex sections of rooms. A data structure is used to link cells visible from each other and a suitable traverse along lines of sight using a recursive search determines the nature of the final image.

All of these methods suffer, in varying degrees, from the amount of computation required at high image complexity, and this precludes them from real time working. In particular it is too optimistic to hope that developments in the near future will produce faster machines thereby upgrading the performance into the real time region, for if a technique to generate an image takes a few seconds then a speed increase of two orders of magnitude is required to attain 25 frames per second. Therefore rather than explore further online techniques it would seem to be more helpful to approach the problem by precomputing as much information as possible, so relying on minimising the real time computational load at the expense of, if necessary, a considerable amount of precomputation. This latter point is justifiable when the model to be represented is fixed since the precomputation only needs to be performed once, even though many real time sequences will subsequently be generated.

Since the nature of many hidden surface algorithms depends to some extent on the type of display utilised, it is appropriate to digress slightly to discuss the type of display system being developed at Sussex University.

The Sussex approach

The requirement of realism implies that a colour display capable of area, rather than line generation, be employed. As Noll (1971) has pointed out, the use of a raster removes the processing load associated with shading a calligraphic image. Further, techniques to minimise delays when moving the beam of a calligraphic display perform no better than a raster at high image complexity. It is also worth mentioning that full colour raster displays are available cheaply in the form of conventional television receivers.

Having decided to use a raster display it is natural to ask in what way surfaces will be represented especially with regard to the hidden surface problem. An attractive approach is to assign a priority value to each surface being displayed so that where there is an overlap on the screen the surface with highest priority is displayed, and the other inhibited. Although this raises design problems associated with the display hardware these will not be dealt with here. The present concern is how to assign priority values to the various surfaces in the model in such a way that the image appears correctly.

Use of distance as a measure of priority

Fig. 1 shows a plan view of two rectangular surfaces standing vertically on the ground, the shape and orientation being chosen for convenience of representation. From the viewpoint shown distance is an unambiguous measure of priority because all points on surface *B* are further from the viewpoint than any point on surface *A*. However in **Fig. 2** this is no longer the case and it becomes necessary to decide from which point on each surface distance should be measured. The correct points are those corresponding to the intersection of a line from the eye, moving in sympathy with the raster scan, with the surfaces in question. Although not especially difficult, the calculation required to do this is time consuming, since it involves solving simultaneously the equation to each planar surface with the equation of the line from the eye. Further, it is necessary to obtain the equations to the planar surfaces either by run time calculation, at a cost of time, or by precomputing at a cost of storage. The equations are clearly viewpoint dependent.

To avoid this, one of the vertices can be taken to be typical of the position of the surface and distance from eye to vertex calculated, this being a much simpler calculation. Unfortunately, as Fig. 2 illustrates, the vertex must be selected with care. This would still be acceptable if the choice is independent of viewpoint: this is not the case. Since distance is sometimes an unambiguous measure of priority (as in Fig. 1) it is desirable to be able to specify *when* it is unambiguous. One further point should be made. Whatever the priority calculation performed, the resulting priority values only need to be unambiguous for surfaces which do actually overlap on the display screen, since only then is the priority calculation utilised. With the foregoing in mind, is it possible to remove such ambiguity from the model without materially affecting the final image?

Clustering

Before the previous question can be answered, it is necessary to introduce the concept of a cluster. The term has been used before in the context of hidden surface removal, by Sutherland, Sproull and Schumaker (1973) and so will be retained here even though used for a different algorithm. They define a cluster as 'a collection of faces that can be treated as a group for some special reason'. In the present case the reason is that the surfaces (faces) are so close together that distance is an ambiguous measure of priority for some viewpoints. It is worth noting that Sutherland, Sproull and Schumaker also consider precomputation and they use spatial partitioning to generate a tree which is traversed according to viewpoint to decide on surface

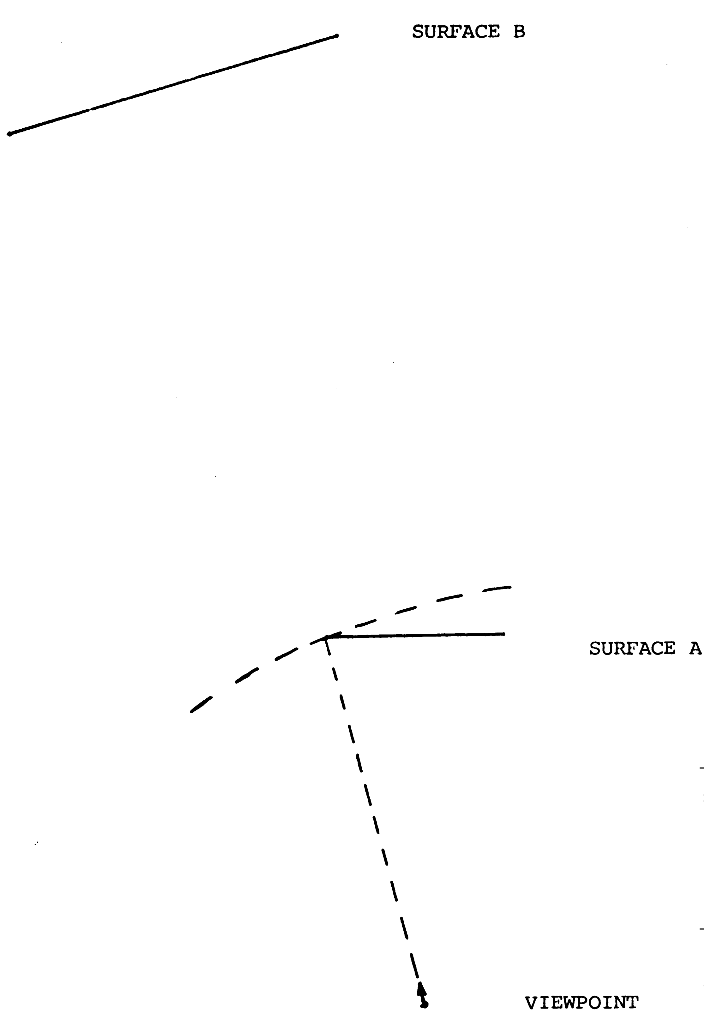

Fig. 1 Distance as an unambiguous measure of priority

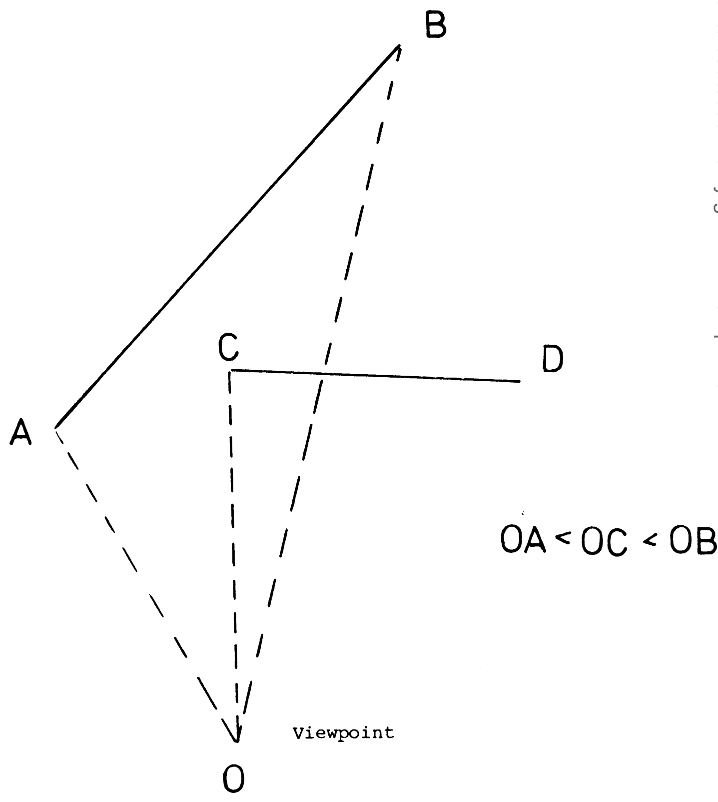

Fig. 2 Distance as an ambiguous measure of priority

priorities. However the scanning of trees, especially elaborate ones, is still rather time consuming.

In Fig. 1 the two surfaces shown are 'well spaced' in the sense that distance is an unambiguous measure of priority, *regardless of viewpoint*. The term 'well spaced' evokes the idea of an invisible boundary surrounding each surface: if the boundary contains no other surfaces then the term 'unclustered' may be applied to that surface, and distance may be used for priority evaluation. Conversely if a boundary contains part or all of another surface it means that the original surface is 'clustered' and so action must be taken before distance can be utilised. As will be seen, this may all be precomputed with the obvious exception of the distance to the surface from a viewpoint.

Determining the boundary

Consider one surface obscuring a second when priority is not well defined as a function of distance. Fig. 2 showed one such case and **Fig. 3** generalises this in plan view. The point A is any point on one of the surfaces, or may be considered to be a very small surface. The coordinate axes have been chosen so that the viewer is at the origin and A is on the positive y axis. This choice is arbitrary and does not affect the final result. The four lines, f, g, h, i arranged about A represent the four distinct positions in which a surface can be placed while satisfying the following pair of conditions.

1. A either obscures (g, h) or is obscured by (f, i) the surface, as seen from the origin O.
2. One end of the surface is nearer to O than A (inside the circle at O through A) and the other end is further from O (outside the circle).

Condition (1) shows that there is a need to resolve priority, while condition (2) shows that distance alone is ambiguous.

The diagram is symmetric about the y axis, so only two cases (g and i for example) need consideration. Note that the first condition is equivalent to the statement that the second surface must cut the y axis between O and A, or must cut it beyond A, depending on the case.

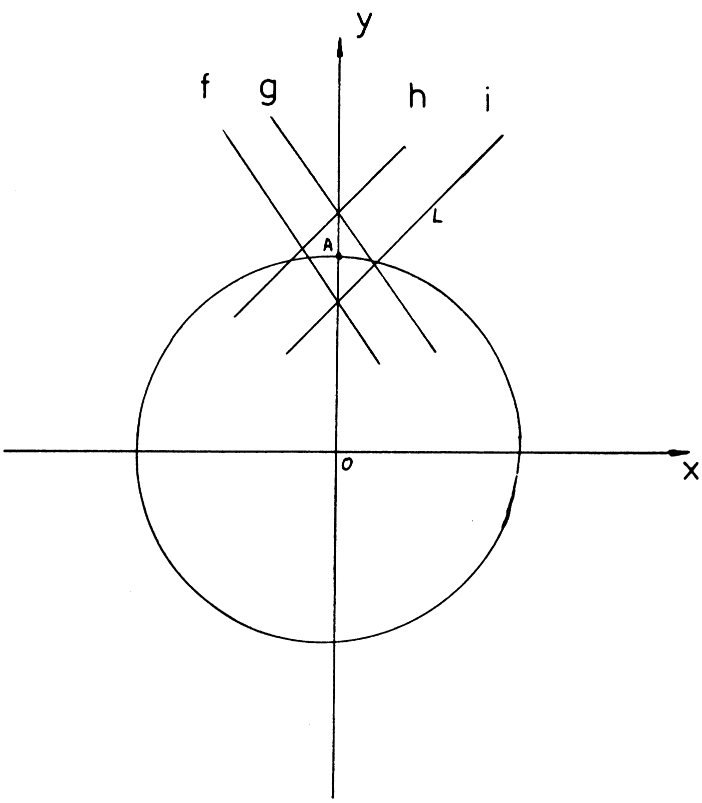

Fig. 3 **Possible orientations of a plane about a point**

Proposition

Let L be the length of the second plane, as shown. Then if conditions (1) and (2) are enforced, the point A always lies within a distance L of the second plane.

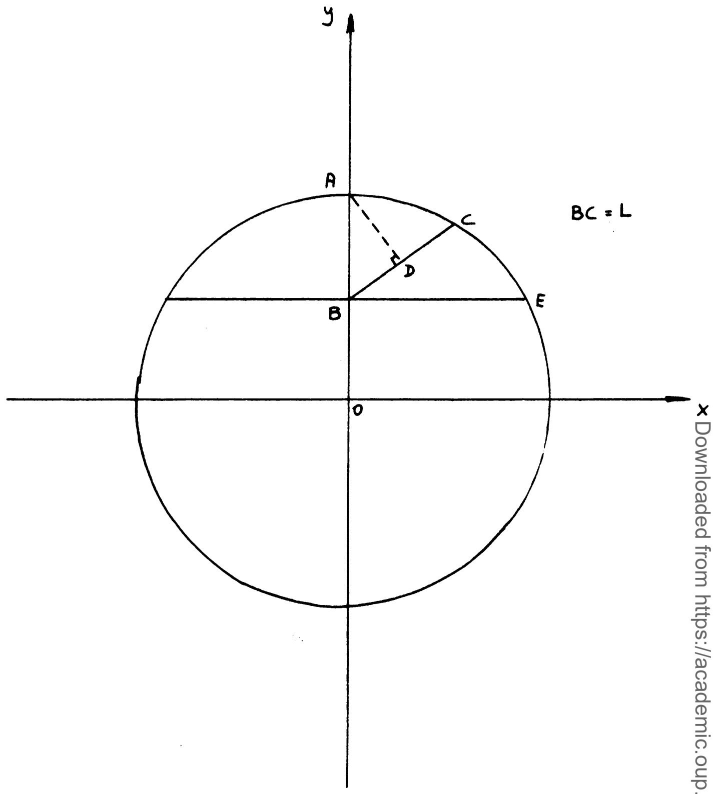

Fig. 4 **First limiting case**

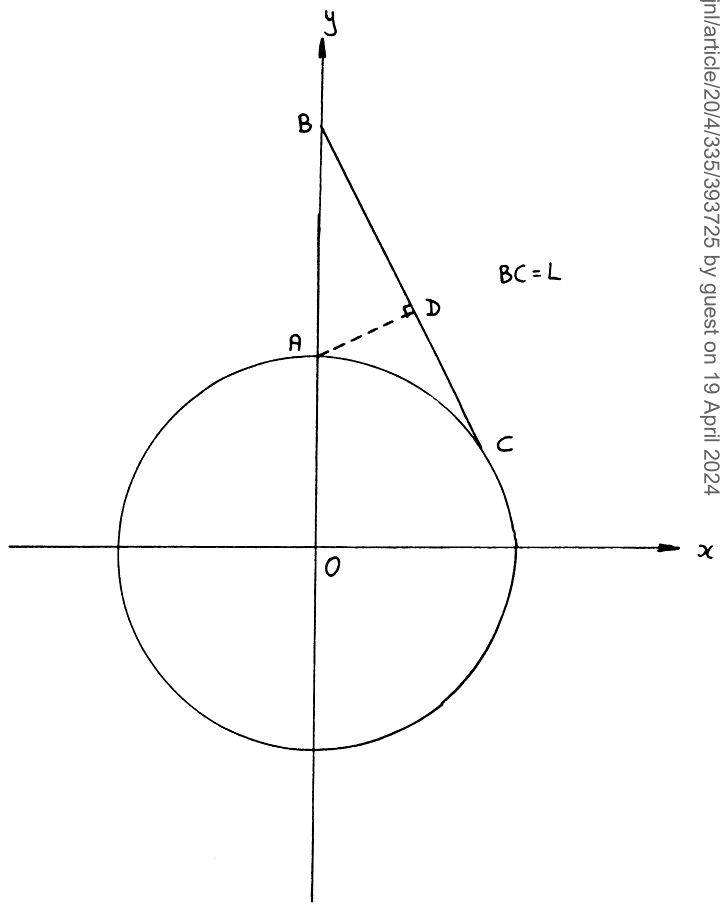

Fig. 5 **Second limiting case**

The proposition can be established by taking the two cases suggested, assuming the conditions stated.

Case 1

Fig. 4 shows the arrangement when the second plane cuts the y-axis at B, a point internal to OA. The limiting case is shown: BC just obscures A and OC is actually equal to OA.

(a) If C is above E, construct the perpendicular AD to BC, such that D is internal to BC and is the point of BC nearest to A.

Now $BC \geqslant AB$

$AB \geqslant AD$ (hypotenuse is longest side)

$\therefore$ $BC \geqslant AD$

i.e. $AD \leqslant L$ as required.

(b) If C is below E, then B is the nearest point of BC to A.

But $BC \geqslant AB$

i.e. $AB \leqslant L$ as required.

Case 2

Fig. 5 shows the arrangement when B is above OA, so that A obscures B. As before, the limiting case is shown.

The perpendicular can now always be constructed so that D is internal to BC, so that it is not necessary to further divide the case.

Now $BC \geqslant AB$

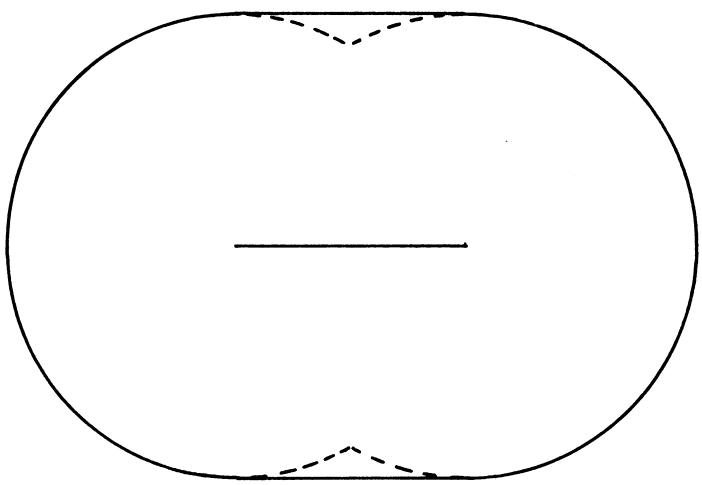

Fig. 6 Boundary shape for clustering

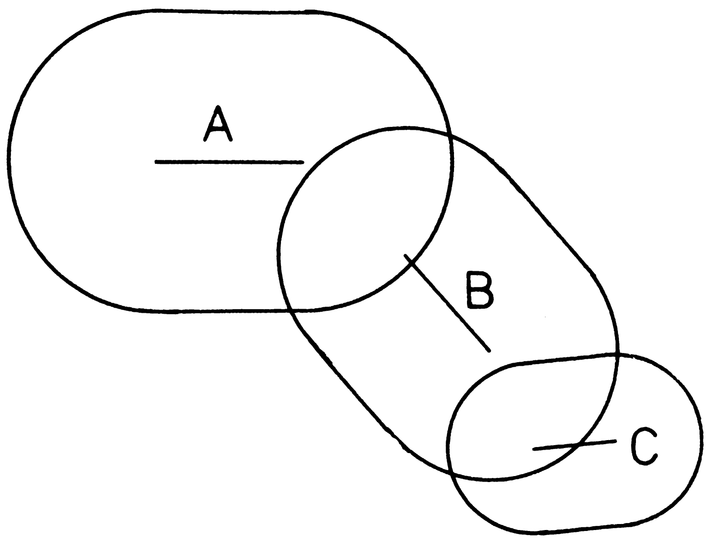

Fig. 7 Clustering is neither associative nor commutative

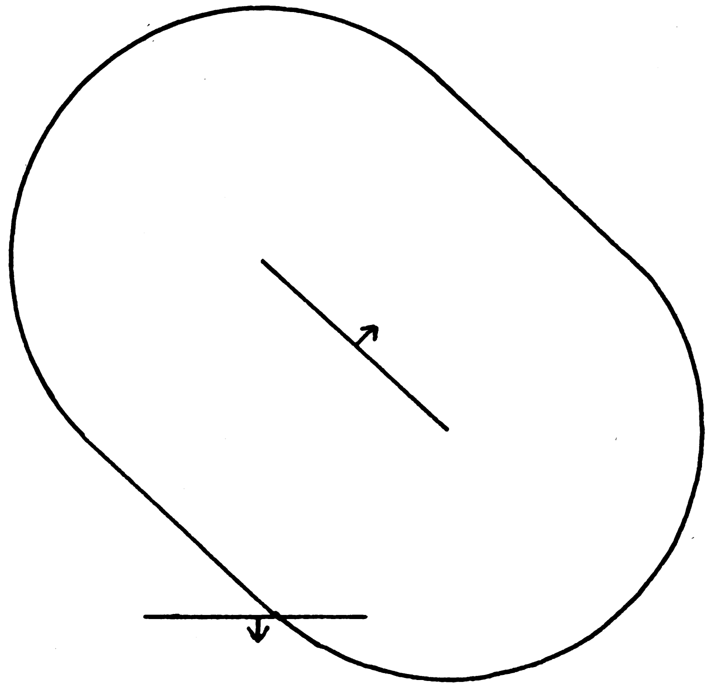

Fig. 8 Clustered surfaces do not necessarily obscure

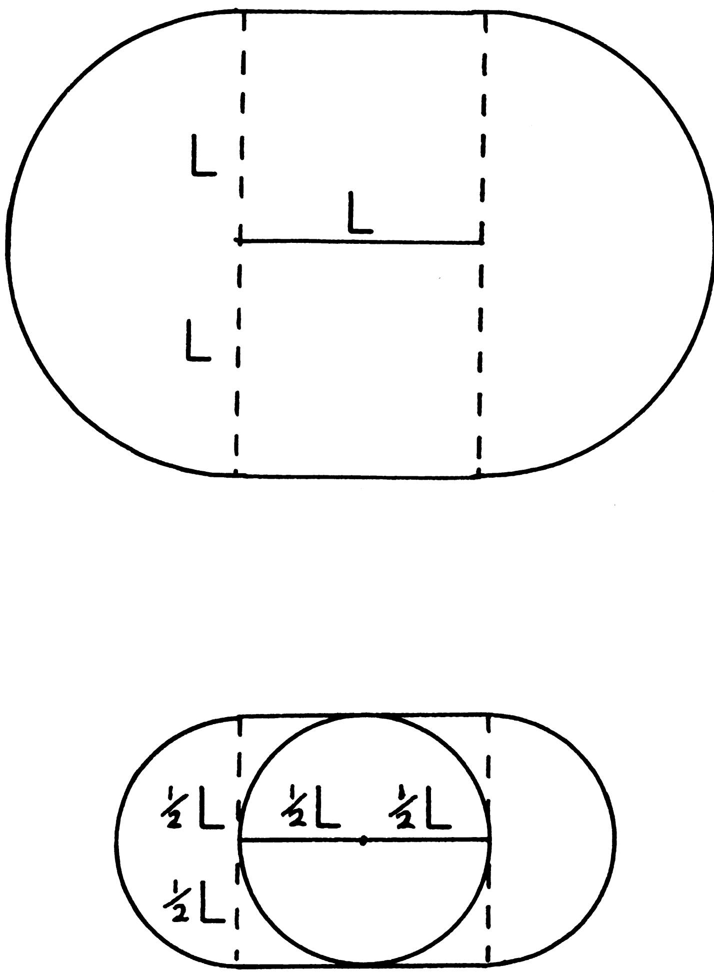

Fig. 9 Area reduction property

$AB \geqslant AD$

$\therefore$ $BC \geqslant AD$

i.e. $AD \leqslant L$ as required.

These derivations apply for any points B and C meeting the original conditions, since those points not explicitly covered lead to BC being longer. Clearly if A lies within L of BC then it certainly lies within a larger radius.

For a given surface it is therefore required that the boundary be at a known radius from that surface, the radius being the length of the surface. The solid line of **Fig. 6** shows such a boundary, and since neither shape nor size depend on the choice of coordinates this is an absolute boundary which can be computed offline. The outline will be referred to as the 'boundary oval' of the surface.

It has not been shown that the boundary oval is the minimal shape with the required properties, and in fact an area reduction of about 2% can be achieved if the minimal shape is required. This is obtained by drawing circles of radius L at each end of the line representing the surface, as indicated by the broken lines in Fig. 6. That the boundary oval is not minimal is a consequence of using simultaneous inequalities in the derivation, so that the expression '$AD \leqslant L$' can sometimes be replaced by the more strict '$AD < L$', allowing the boundary to approach the surface more closely as shown by the broken lines. This double circle shape, referred to by the general term of 'proximity boundary', may be proved to be minimal. The proof is omitted here for two reasons. Firstly, much of what follows does not depend on the precise shape of the boundary, and secondly, depending on the surface representation employed, it may be simpler to test whether a point lies within a fixed distance of the line representing the surface than to test whether it is within a certain distance of either end of that line. For practical purposes it makes little difference that the boundary oval picks up some points which are not in fact ambiguously placed.

Properties of the boundary oval

The following properties are based on plane surfaces, arranged to be visible from one side only, and assumed to be vertical so that, as before, a simple plan view may be utilised.

1. If a model is built from surfaces, such that no surface intersects the boundary oval of any other, then the model may be said to be unclustered and the distance from the viewer to *any* point on the surface may be used as a priority value for that surface.
2. Even though a surface is, in effect, one sided, the complete oval must be used. The front half of the oval tests for surfaces which may obscure the present one, while the rear half tests for surfaces which may be obscured. Note that it is the intersection of surface and oval which is sought, and not the intersection of two ovals.
3. Clustering is, in general, neither associative nor commutative. That it is not commutative usefully limits the size of cluster that need be considered, as shown by **Fig. 7,** where A and C do *not* form a cluster. The associative property means that although A clusters with B, B does not cluster with A, so in testing for a cluster (which A and B together form) it is therefore necessary to test 'both ways'.
4. If a pair of surfaces form a cluster it does not follow that either obscures the other, if the surfaces are considered to be visible from one side only. This is often the case for ease of computing which surfaces are facing away from the viewer. **Fig. 8** shows an example of this.
5. If the surface has a length L (**Fig. 9**) then the oval has an area $(\pi + 2)L^2$. So replacing that surface with two others of length $\frac{1}{2}.L$ leads to a reduction in area by about 65%, thus greatly reducing the 'catchment area' of the oval. Thus smaller surfaces are more likely to yield an unclustered model, at the expense of a greater storage requirement.
6. Under the assumption that surfaces and solids are convex, no two surfaces on the same solid need be tested for clustering, because one surface can never obscure the other. For two surfaces $S1$ and $S2$ not on the same solid, $S1$ can only obscure $S2$ if at least one end of $S2$ is behind the plane containing $S1$; if at least one end of $S1$ is in front of the plane containing $S2$; and if the two vector normals (pointing out from the visible front side of each surface) are within 90° of each other.

The cluster algorithm

It is now possible to specify a new algorithm for use at the model creation stage which will permit the use of distance as an unambiguous measure of priority at run time.

The area reduction property implies that splitting a surface into two or more smaller surfaces will result in a smaller boundary. When a cluster is recognised it is thus necessary to break down surfaces into lesser surfaces until there is no longer a cluster. The precise manner in which this is achieved depends on the representation chosen to hold the model. Certain points are clear however. No two surfaces which touch may ever be declustered into a finite number of surfaces, but in general such surfaces are part of the same object: provided the model is restricted to convex objects it is not necessary to consider such pairs of surfaces, as mentioned in (6) earlier.

Concluding remarks

A property has been presented which is of use to the real time computation of hidden surfaces in a system in which an essentially fixed model is to be viewed from arbitrary positions. The use of a boundary to determine proximity of surfaces is not viewpoint dependent and an important consequence of this is that surfaces which are too close, referred to as being clustered, may be detected at the model creation stage by an implementation dependent compilation process which is responsible for generating smaller surfaces to replace individual clustered surfaces. Once an unclustered model has been created it is possible to use any point on a surface as a measure of the distance, and hence priority, of that surface. Minimising the real time computation to simple distance measures allows real time image generation in a straightforward manner.

References

APPEL, A. (1968). Some techniques for the machine renderings of solids, *AFIPS*, Vol. 32, p. 37.

BOUKNIGHT, W. J. (1970). A procedure for generation of three-dimensional half toned computer graphics representations, *CACM*, Vol. 13, No. 9, p. 527.

GRIMSDALE, R. L., HADJIASLANIS, A. A., and WILLIS, P. J. (1976). Improvement in display apparatus, provisional patent specification 03301/76.

JONES, C. B. (1971). A new approach to the 'hidden-line' problem, *The Computer Journal*, Vol. 14, No. 3, p. 232.

NOLL, A. M. (1971). Scanned display computer graphics, *CACM*, Vol. 14, No. 3, p. 143.

ROMNEY, G. W., WATKINS, G. S., and EVANS, D. C. (1968). Real time display of computer generated half tone perspective pictures, *IFIP*, Vol. 2, p. 973

SUTHERLAND, I. E. (1970). Computer displays, *Scientific American*, Vol. 222, No. 6, p. 57.

SUTHERLAND, I. E., SPROULL, R. F., and SCHUMAKER, R. A. (1973). Sorting and the hidden surface problem, *AFIPS*, Vol. 42, p. 689.

WILLIS, P. J. (1976). The computer generation of images in real time, D.Phil. thesis, University of Sussex.