The Visual Hull Concept for Silhouette-Based Image Understanding

Aldo Laurentini

***Abstract*— Many algorithms for both identifying and reconstructing a 3-D object are based on the 2-D silhouettes of the object. In general, identifying a nonconvex object using a silhouette-based approach implies neglecting some features of its surface as identification clues. The same features cannot be reconstructed by volume intersection techniques using multiple silhouettes of the object.**

This paper addresses the problem of finding which parts of a nonconvex object are relevant for silhouette-based image understanding. For this purpose, the geometric concept of *visual hull* of a 3-D object is introduced. The visual hull of a 3-D object S is the closest approximation of S that can be obtained with the volume intersection approach. An equivalent statement, relative to object identification, is that the visual hull of S is the maximal object silhouette-equivalent to S, i.e., which can be substituted for S without affecting any silhouette. Only the parts of the surface of S that also lie on the surface of the visual hull can be reconstructed or identified using silhouette-based algorithms. The visual hull of an object depends not only on the object itself but also on the region allowed to the viewpoint. Two main viewing regions can be considered, resulting in the *external* and *internal* visual hull. In the former case the viewing region is related to the convex hull of S, in the latter it is bounded by S itself. The internal visual hull also admits an interpretation not related to silhouettes: the features of the surface of S that is not coincident with the surface of the internal visual hull cannot be observed from any viewpoint lying outside the convex hull.

After a general discussion of the visual hull and its properties, algorithms for computing the internal and external visual hulls of 2-D objects and 3-D planar face objects are presented and their complexity analyzed. In general, the visual hull of a 3-D planar face object turns out to be bounded by planar and curved patches. A precise statement of the concept of visual hull appears to be novel, as is the problem of its computation.

***Index Terms*— Computer vision, image understanding, shape recognition, silhouettes, shape from contour, volume intersection, convex hull, visual hull.**

I. Introduction

To understand the 3-D content of a scene is a central problem in computer vision, since it can allow computer-driven equipment to perform tasks such as navigation, manipulation, and visual recognition. Many approaches have been proposed for identifying or reconstructing 3-D objects when 2-D images are available; these approaches are usually categorized as shape-from-X, according to the information used.

Manuscript received March 2, 1992; revised October 2, 1992. Recommended for acceptance by Associate Editor T. Henderson.

The author is with the Dipartimento di Automatica ed Informatica, Politecnico di Torino, 10129 Turin, Italy.

IEEE Log Number 9212303.

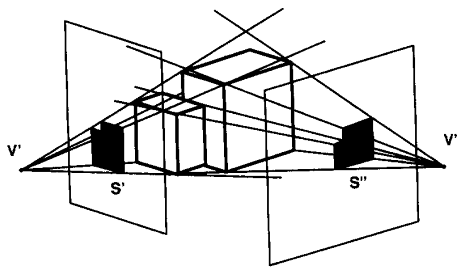


Fig. 1. The volume intersection approach to object reconstruction.

Among other image cues, the silhouettes of a 3-D object S can be selected as sources of shape information. The 2-D silhouette, obtained as a parallel or perspective projection of S onto the image plane, has been indicated as an effective psychophysical clue to shape understanding [1]. Object recognition based on silhouettes has been performed for many years by human observers. Referring to computer-based systems, obtaining the 2-D silhouette is usually a computationally simple task, one that can be most often performed by thresholding an intensity image. The algorithms for obtaining the silhouettes, or their boundaries, usually also perform well with degraded images. For these reasons, the silhouette is a favorite cue for image understanding.

A number of silhouette-based algorithms have been proposed for object identification [2]–[9]; many industrial vision systems also identify objects or their attitude using silhouettes.

Object reconstruction can be performed using the volume intersection approach, which recovers the volumetric description of the object from multiple silhouettes. This approach, proposed in [10] by Martin and Aggarwal, and used by many object reconstruction algorithms [11]–[17], consists in intersecting the solid regions of space in which the object is constrained to lie by each silhouette. These regions are cones obtained by back projecting from a viewpoint V the corresponding silhouette for perspective projections (see Fig. 1), or cylinders obtained by sweeping the silhouette along a line parallel to the viewing direction for orthographic projections. In the following, as the orthographic projection can be considered a limit case of perspective projection, we will always refer to these regions as cones. The boundary of each cone, which consists of all the half-lines starting at V and tangent to S, will be referred to as the circumscribed cone of S relative to V.

As the number of cones increases, the object is reconstructed with higher precision. Some papers also address the problem

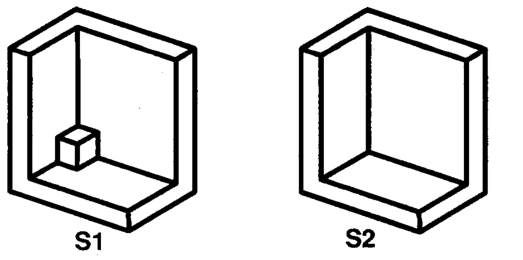


Fig. 2. Two objects that cannot be distinguished using silhouettes

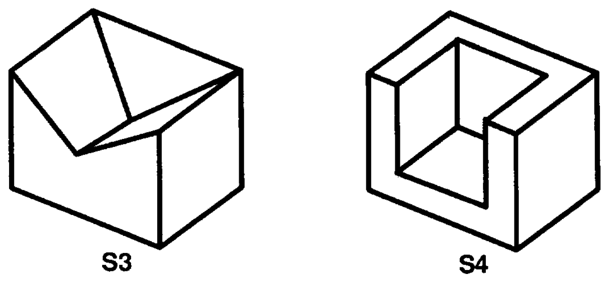


Fig. 3. Two objects that cannot be reconstructed using volume intersection.

of specifying the viewpoints that optimize the construction, without *a priori* information on the object [18].

In spite of their attractive features, silhouettes can be insufficient clues for fully understanding 3-D shapes. It is clear that identification algorithms based on silhouettes are not always able to discriminate different nonconvex objects. For instance, no silhouette-based algorithm appears to be able to tell one of the two objects S_1 and S_2 in Fig. 2 from the other. The difference between S_1 and S_2 consists in a small cube that lies in a concavity. It is easy to understand that this cube does not manifest itself in any possible silhouette of the objects, at least considering viewpoints not too near to the object.

The silhouette-based approach to object identification therefore raises questions like the following.

- Which part of the surface of a 3-D object S can affect the silhouettes of the object?
- To what extent can the complementary part of the surface of S take different shapes without affecting any silhouette?

The algorithms based on the volume intersection approach are not able to exactly reconstruct any possible object, not even performing infinite intersections. Here and in the following, "to exactly reconstruct" means "to reconstruct with arbitrarily high precision," which is the correct statement referring to the reconstruction of general solids, e.g. a sphere. It is not difficult to recognize that none of the simple polyhedral objects S_1 and S_2 in Fig. 2, and S_3 and S_4 in Fig. 3, can be exactly reconstructed using volume intersection, at least considering viewpoints not "too near" the object.

The silhouette-based approach to object reconstruction raises therefore questions such as the following.

- Which part of the surface of a 3-D S object can be exactly reconstructed with volume intersection?
- When S cannot be exactly reconstructed, which is the closest approximation of S that can be obtained using volume intersection techniques?

Precise answers to all the previous questions will be provided in the following sections. Here we will acquire some insight into the problem by inspecting the planar face object S_4 in Fig. 3. This object is sufficiently simple to answer the previous questions by an intuitive analysis.

Let us assume that the possible viewpoints are not too near the object (we will specify in the following the precise meaning of this statement). It is not difficult to realize that the surface highlighted in Fig. 4(a) can be reconstructed by volume intersection. This is also the surface relevant for the silhouettes

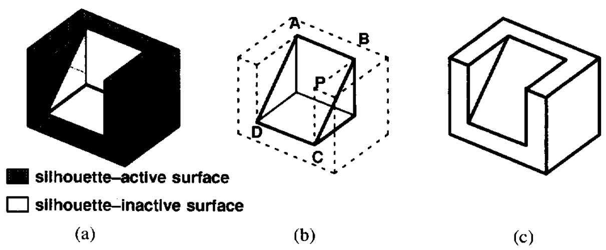


Fig. 4. The behavior of S_4 with respect to silhouette-based identification and reconstruction. (a) The silhouette-active and inactive surfaces of S_4. (b)S_4 can assume any shape inside P without affecting any silhouette. (c) The closest approximation of S_4 that can be obtained with volume intersection.

of S_4. Let this surface be the *silhouette-active* surface of S_4. The complementary *silhouette-inactive* surface of S_4 is also shown in Fig. 4(a). This surface can take any shape, without affecting any silhouette, inside the pentahedron P shown in Fig. 4(b), bounded by the contour-inactive surface and the rectangle ABCD. The closest approximation of S_4, i.e., the smallest object that can be obtained using volume intersection techniques, is the object in Fig. 4(c), which is the union of S_4 and the pentahedron P.

The limits of the contour-based approach for understanding nonconvex shapes have been qualitatively recognized in many papers (see for instance [10] and [11]). Until now, however, to the author's knowledge this topic has not been thoroughly discussed, and no general answers to the previously stated questions are available.

The purpose of this paper is to develop a general geometric tool for dealing in a simple and straightforward way with the questions raised for nonconvex objects by the silhouette-based approach to image understanding. This tool will be called the *visual hull* of an object S. Referring the reader to the following section for more precise definitions, here we introduce the general idea of visual hull. Broadly speaking, the visual hull of an object S is the envelope of all the possible circumscribed cones of S. An equivalent intuition is that the visual hull is the maximal object that gives the same silhouette of S from any possible viewpoint. For instance, the visual hull of the sample object S_4 is shown in Fig. 4(c). Its surface consists of two parts: the contour-active surface of S_4, and the rectangle ABCD. We can consider this second surface as produced by all the visual rays tangent to S at the segments AB and atCD.

The visual hull idea has been first introduced by the author in [19], together with a preliminary algorithm for its computation in 2-D. An algorithm for computing the visual hull of a solid of revolution is described in [20].

The content of this paper is as follows. In Section II, precise definitions of visual hull are stated and its general properties analyzed. With reference to the possible regions that contain the viewpoints, two main cases are considered and the *external* and *internal* visual hulls are defined. In Section III, efficient algorithms for computing both the internal and external visual hulls in 2-D are described. An algorithm for the computation of the silhouette-active surface of a 3-D polyhedral object, based on the 2-D algorithm of Section III, is presented in Section IV. In Section V, the problem of computing the visual hull of a general polyhedron is addressed. The surfaces that are potential boundaries of the external and internal visual hulls of polyhedral objects are determined. It turns out that the visual hulls of polyhedra are bounded by planar and quadric surfaces. Finally, algorithms are presented for finding both visual hulls.

II. The Visual Hull: Definitions and General Properties

This section is organized as follows. In Section II–A a formal definition of the visual hull of S relative to a viewing region is given. It is shown that this definition conforms to the intuitive discussion of Section I, and the questions there presented admit simple answers in terms of visual hull. In Section II–B, considering viewpoints outside the convex hull of the object, we show that there is a unique visual hull, not exceeding the convex hull, for any viewing region completely enclosing S. We refer to this main case as the external visual hull. Section II–C deals with an unrestricted viewing region. The corresponding internal visual hull also supplies information on the part of surface of S that is not observable from points outside its convex hull.

A. The Visual Hull of an Object Relative to a Viewing Region

Let R be the viewing region of 3-D Euclidean space that contains the viewpoints that are allowed for observing the object S. We must take into account the position allowed to the viewpoint V, and we define the visual hull as dependent not only on S but also on R. We give the following simple geometric definition, which characterizes the points belonging to the visual hull.

Definition 1: The visual hull $VH(S, R)$ of an object S relative to a viewing region R is a region of E^3 such that, for each point $P \in VH(S, R)$ and each viewpoint $V \in R$, the half-line starting at V and passing through P contains at least a point of S.

We see obviously that $S \leq VH(S)$, since any point of S satisfies this definition. It is immediately possible to verify that Definition 1 satisfies the intuitive idea of visual hull stated in the introduction. In fact, the two following propositions can be obtained.

Proposition 1: $VH(S, R)$ is the maximal object silhouette-equivalent to S with respect to R (i.e., that gives the same silhouette as S when observed from any $V \in R$).

In fact, the following statements apply.

1) $VH(S, R)$ is silhouette-equivalent to S with respect to R, since 1) according to Definition 1, the projection of any point $P \in VH(S, R)$ from any viewpoint $V \in R$ belongs to the silhouette of S obtained from V, and 2) the projection of any point $Q \in S$ from any viewpoint $V \in R$ belongs to the silhouette of $VH(S, R)$ obtained from V, since $S \leq VH(S, R)$.
2) $VH(S, R)$ is the maximal silhouette-equivalent object, since for any point $P' \notin VH(S, R)$, there is at least a line, starting at a point $V' \in R$ and passing through P', that does not intersect S. Therefore the projection of P' does not belong to the silhouette of S obtained from V'. This implies that P' cannot belong to an object silhouette-equivalent to S. □

Proposition 2: $VH(S, R)$ is the closest approximation of S that can be obtained using volume intersection techniques with viewpoints $V \in R$.

In fact, only points $P \notin VH(S, R)$ can be removed by intersecting 3-D cones, since only such points can lie outside some silhouette and therefore outside some 3-D cone. □

Another property of the visual hull can be readily obtained.

Proposition 3: If $R > R'$, then $VH(S, R) \leq VH(S, R')$.

Of course, if a wider viewing region is available, a greater number of cones can be intersected and more points can be removed. □

In conclusion, the definition given for the visual hull of an object S relative to a viewing region R conforms to the intuitions reported in the introduction. The following answers to the questions there reported can be given in terms of visual hull:

- $sa(S, R)$, the silhouette-active surface of S relative to the region R, is the part of the surface $s(S)$ of S that also belongs to $vh(S, R)$, the boundary of $VH(S, R)$.
- $si(S, R)$, the silhouette-inactive surface of S, can assume any shape inside the volume $VH(S, R) - S$ without affecting any silhouette obtained from R.
- $sa(S, R)$ is the part of $s(S)$ that can be exactly reconstructed by volume intersection using silhouettes obtained from R.
- $VH(S, R)$ is the closest approximation of S that can be obtained by volume intersection using silhouettes obtained from R.

Using the visual hull concept, other synthetic statements referring to silhouette-based image understanding can be formulated. For instance, the following rules apply.

- Two objects S and S' can be discriminated using their silhouettes observed from a viewing region R iff $VH(S, R) \neq VH(S', R)$.
- An object S can be exactly reconstructed by volume intersection techniques using silhouettes observed from a viewing region R iff $S = VH(S, R)$.

Let us make a final remark:

Proposition 4: The silhouette-active surface $sa(S, R)$ of an object uniquely determines its visual hull $VH(S, R)$.

This proposition follows immediately from the fact that all

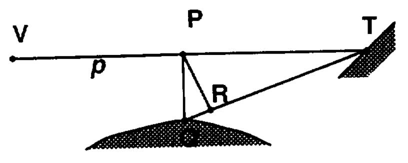


Fig. 5. The visual hull relative to regions completely enclosing S belongs to its convex hull.

the circumscribed cones relative to an object are determined uniquely by its silhouette-active surface. □

B. The Main Case: The External Visual Hull

According to definition 1, there is an infinite set of visual hulls of an object S, one for each possible viewing region R. This seems to reduce in some way the effectiveness of the idea. Actually, we will show that for each object S there is a main case of visual hull, to which the majority of practical cases can be referred. In addition, we will see that this main case is related to $CH(S)$, the convex hull of S.

Observe first that in many practical cases 1) the object to be identified or reconstructed can assume any attitude with respect to the viewpoint V, and 2) V lies at some distance from the unknown object.

Let us consider therefore the visual hull $VH(S, R')$ relative to a region R' that *completely encloses* S. Let us also suppose that

$$R' \in R_c = E^3 - CH(S)$$

i.e., no viewpoint is allowed inside the convex hull of the object. It is easy to show that the following proposition holds.

Proposition 5:

$$VH(S, R') \leq CH(S).$$

To prove this statement, let us consider a point $P \notin CH(S)$. We can easily find viewpoints such that the viewline passing through P does not intersect S. Viewpoints of this kind lie on the plane p passing through P and normal to the segment PQ, where Q is the point of $CH(S)$ closest to P. In fact, by contradiction, let us suppose that the viewline VP, lying on p, intersects $CH(S)$ at point T (see Fig. 5). Point R, which belongs to the convex hull as a linear combination of Q and T, and such that PR is normal to QT, would be closer to P than Q, against the hypothesis. □

Let us now consider another viewing region R'', which like R' encloses S and lies outside the convex hull. We have

Proposition 6:

$$VH(S, R') = VH(S, R'').$$

In fact, for any point $Q \in VH(S, R')$, and for any half-line passing through Q and starting at $V' \in R'$, it is possible to find a point $V'' \in R''$ on the same half-line. Vice-versa, for any point $Q \in VH(S, R'')$ and for any half-line passing through Q starting at $V'' \in R''$, it is possible to find a point $V' \in R'$ on the same half-line (see Fig. 6). Therefore, according to definition 1, any point Q belonging to $VH(S, R')$ also belongs to $VH(S, R'')$ and vice-versa. □

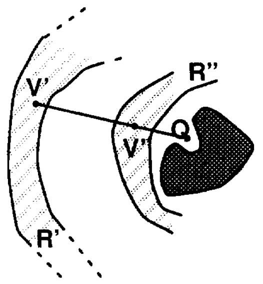


Fig. 6. All the visual hulls of S relative to regions completely enclosing S are equal.

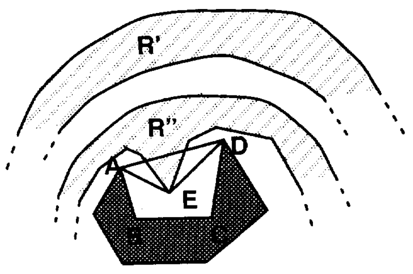


Fig. 7. The visual hulls relative to R'' and R' are different.

Observe that proposition 6 is no more guaranteed to hold if one or both viewing regions have parts inside the convex hull, since one of these regions can have points inside the visual hull of the other. As an example, let us consider the 2-D case of Fig. 7, where the region R'' has a part inside the convex hull of S. It can be easily verified by inspection that $VH(S, R')$ contains the quadrangular region ABCD, and $VH(S, R'')$ contains the pentagonal region ABCDE. This is because the viewpoints of R'' that also belong to $VH(S, R')$ "dig" the visual hull relative to R', deleting the triangular region ADE.

In conclusion, there is a unique visual hull, not exceeding the convex hull of S, relative to all the viewing regions that enclose S and do not enter its convex hull.

Definition 2: $VH(S, R_c)$ is defined as the *external* visual hull, or simply the visual hull of S, $VH(S)$, without any other specification.

The surface of S coincident with the surface $vh(S)$ of $VH(S)$ will be said to be the *external silhouette-active* surface of S or, briefly, the *silhouette-active* surface $sa(S)$. Actually, according to the above discussion, *the viewing region relative to the visual hull can extend also inside the convex hull, as long as it does not enter the visual hull itself.* This suggests another definition of external (or *natural*) visual hull: $VH(S)$ is the largest visual hull of S whose viewing region is bounded by the visual hull itself.

$VH(S)$, which we made implicit reference to in the introduction considering viewpoints "not too near" to the object, has other interesting properties. Let us consider any viewing region R_s that is a subset of R_c and does not include S. The visual hull $VH(S, R_s)$ relative to this region is strictly related to $VH(S)$. In fact, $VH(S, R_s)$ contains $VH(S)$, and its surface $vh(S, R_s)$ is equal to the surface $vh(S)$ of $VH(S)$

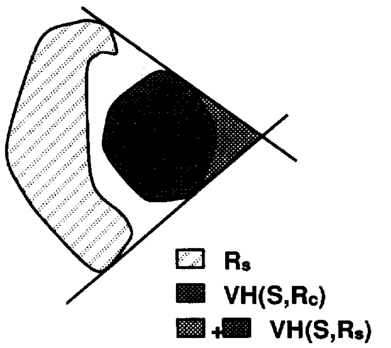


Fig. 8. The visual hull relative to R_s can be obtained from the visual hull relative to R_c.

at the points of tangency to $VH(S)$ of lines passing through R_s. The remaining part of $vh(S, R_s)$ is bounded by the lines tangent to both R_s and $VH(S)$, as it is shown by the example of Fig. 8, where 2-D sections of R_s, $VH(S)$ and $VH(S, R_s)$ are presented.

A point $Q \in VH(S)$ can be characterized very simply in terms of lines passing through Q. In fact, we have the following proposition.

Proposition 7: Q belongs to $VH(S)$ iff any line passing through P contains at least one point of S.

This statement will be used in the following for computing the visual hull.

C. The Internal Visual Hull

If the boundary of the viewing region is S itself, we have another case worth considering. Let $R_i = E^3 - S$.

Definition 3 $VH(S, R_i)$ is defined as the *internal* visual hull of S, $IVH(S)$.

The surface of S coincident with the surface $ivh(S)$ of $IVH(S)$ will be said to be the *internal silhouette-active* surface of S, $isa(S)$. Also, let the complementary surface be the *internal silhouette-inactive* surface, $isi(S)$.

The points belonging to the internal visual hull can be simply characterized in terms of half-lines:

Proposition 8: Q belongs to $IVH(S)$ iff any half-line passing through Q contains at least one point of S.

This statement, like the previous one relative to the visual hull, will be used for constructing the internal visual hull.

An inspection of all the sample objects so far considered shows that their internal visual hulls are equal to the corresponding objects.

It is not likely that an attempt will be made to recognize or reconstruct objects using silhouettes from a viewpoint lying inside their convex hull. However, the internal visual hull admits another interesting visual interpretation not related to silhouettes. In fact, from proposition 8 the following follows immediately.

Proposition 9: $isi(S)$, the internal silhouette-inactive surface, cannot be observed from any viewpoint $V \notin CH(S)$.

Therefore, the geometric concept of internal visual hull can be used for determining which features (edges, textures, etc.) of a 3-D nonconvex object will never appear on any image of the object obtained from points outside the convex hull. Observe that to be V outside $CH(S)$ is a sufficient condition for proposition 9 to hold; $isi(S)$ could be unobservable also from points belonging to $CH(S)$.

Since the viewing region relative to $IVH(S)$ is the largest among the possible viewing regions, from proposition 3 we obtain the following.

Proposition 10:

$$IVH(S) \leq VH(S).$$

III. The Computation of $VH(S)$ and of $IVH(S)$ in 2-D

An effective use of the visual hull idea requires algorithms for its computation. The visual hull is basically a 3-D entity; algorithms for finding visual hulls in 2-D could appear of little use. Actually, the 2-D algorithms are able to deal with 2.5-D sweep solids. In addition, the 2-D algorithms and concepts are useful for solving the 3-D problem for planar face objects, as we will see in the following. Also, the algorithms for computing both visual hulls of solids of revolution presented in [20] stems from the 2-D algorithms.

The content of this section is as follows. In III–A an algorithm for computing the visual hull of a set of polygons SP is presented. The algorithm is based on the simple observation that a point does not belong to the visual hull if there are lines passing through this point that do not intersect SP. It will be shown that these *visual* lines can be arranged into families, and the plane can be divided into polygonal zones such that the visual lines passing through every point of each zone can be arranged into the same number of families. After constructing this partition of the plane, the visual hull can be obtained visiting all the zones and merging the zones having zero families of visual lines.

In III–B we present an algorithm for computing the internal visual hull. This algorithm is derived from the previous one, with suitable modifications.

A. An Efficient Algorithm for Computing the 2-D Visual Hull

Let us consider a set of polygons SP in a plane p. To consider a nonconnected set of polygons is clearly a necessary condition for obtaining $VH(SP) \neq CH(SP)$. Let us define a *visual line* as a straight line belonging to p that does not intersect SP. The visual lines passing through a point Q can be grouped into a number of families, like those shown in Fig. 9.

Definition 4: The *visual number* $VN(Q, SP)$ of a point Q relative to SP is defined as the number of families of visual lines passing through Q.

It is clear that $VN(Q, SP)$ cannot exceed the number of nonconnected components of SP. Referring to proposition 7, we have the following.

Proposition 11: A point Q belongs to $VH(SP)$ if and only if it is $VN(Q, SP) = 0$.

The basic idea of the algorithm for finding VH consists of dividing the plane into regions containing points with the same visual number, and in merging all the regions whose visual number is zero.

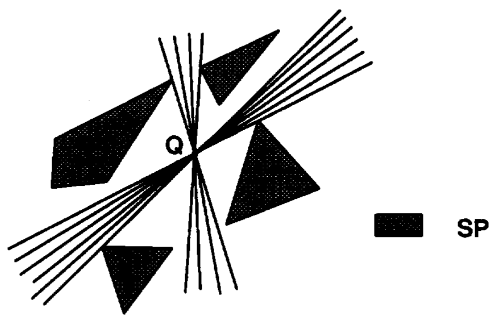


Fig. 9. Through Q pass two families of visual lines.

To obtain this partition of the plane, we will use a number of *active* straight lines. A line is defined to be active if it contains one or more *active segments*; an active segment is a part of a line that is the boundary between points whose visual numbers differ by one. In the following, with the words tangent to a polygon (or polyhedron) S we will specify straight lines that share with the boundary and only the boundaries of S any number of points. It is clear that active lines must be searched among the lines that are tangent to SP at two points and do not intersect SP at any other point. Analyzing these lines, we find the cases of active lines shown in Fig. 10, where these principles apply.

1) The active segments are highlighted with a thicker line.
2) The arrows at both ends of the active lines indicate that these lines do not cross SP at any other point.
3) The arrows crossing the active segments mark the *positive* directions, that is the directions toward the *positive side* of each segment, containing a region whose visual number is higher.
4) In each of the three cases, a point Q on the positive side of an active segment is shown, together with the boundary lines of the family of visual lines that will disappear if the point Q crosses the active segment toward the negative side.

It must be observed that case (c) has been introduced for the sake of the algorithm, which also visits regions inside SP.

Exceptional alignment conditions of some vertexes of S, which lead to cases of multiple tangency, will not be analyzed in detail. In fact, these cases can be easily reduced to combinations of the cases with two tangency points, as can be appreciated from the example shown in Fig. 11. Exceptional vertexes that are common to more than one pair of edges need not be considered for finding active lines.

The outline of the algorithm for computing $VH(SP)$ is as follows.

1) Compute all the active segments.
2) Construct a partition of the plane produced by these segments using a plane-sweep algorithm [21].
3) Starting from a region inside SP, whose visual number is 0, traverse the partition in order of adjacent regions and compute the visual number of each region of the partition, subtracting or adding one to the number of the previous region, according to the crossing direction of each active segment.
4) Merge the regions whose VN is zero.

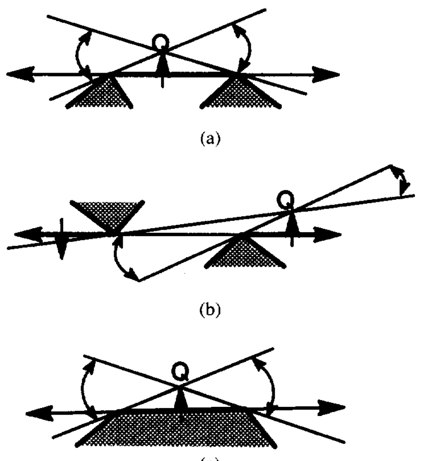


Fig. 10. The three cases (a), (b), and (c) of active lines and their active segments.

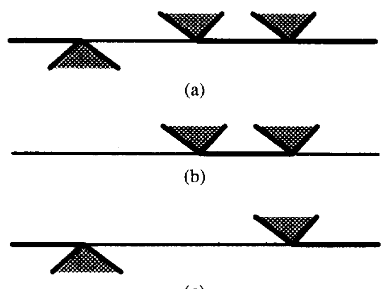


Fig. 11. Example of decomposition of a case of tangency at three points (a) into two cases of tangency at two points (b) and (c).

In the following each step will be described in detail and the complexity of the algorithm analyzed.

Let n be the number of vertexes of SP, which has therefore $O(n)$ edges. The algorithm that we are presenting is, in general, more efficient than the algorithm presented in [19]. In the current case, we compute the active segments and perform their intersections with a plane sweep algorithm. This allows us to make the computation time of the algorithm sensitive to the size of the output. This approach has been recently applied by Gigus, Canny, and Seidel for constructing a partition of the plane with segments of straight lines and conics [22].

Step 1: To determine the active segments requires considering $O(n^2)$ pairs of vertexes. Some pruning of these cases can be performed. Concave vertexes can be discarded. Some of the lines joining two vertexes can be pruned away in constant time, testing whether they cross SP at one of the vertexes. Since the visual hull does not exceed the convex hull (proposition 5), a further pruning can be performed. The potentially active lines of type (b) in Fig. 10 have potentially active segments that extend outside the convex hull. The convex hull can

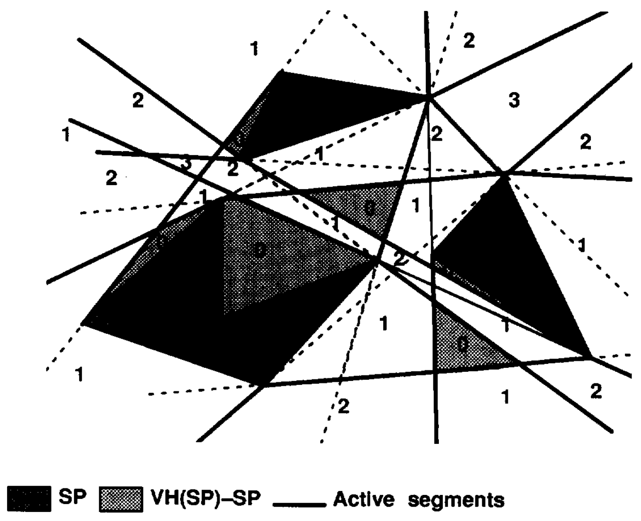


Fig. 12. An example of visual hull in 2-D.

be computed in $O(n \log n)$ time [21], and the parts of the potentially active segments that lie outside the convex hull can be pruned away in $O(n)$ time for each line. The remaining lines—or parts of lines—must be checked for intersection with the $O(n)$ edges of SP. The time bound for finding $O(n^2)$ active segments is $O(n^3)$.

Step 2: Using a plane sweep algorithm we can construct a data structure containing the vertexes and edges of the partition in $O(m \log m)$ time, where m is the number of vertexes of the partition. For each edge we store the positive direction.

Step 3: In this step, we start from a region inside SP whose visual number is zero and traverse the partition computing the visual number for each region. The time required for traversing the partition is $O(m)$, the same required for visiting the dual graph with a depth-first algorithm [23].

Step 4: Merging the adjacent regions that belong to the visual hull, for obtaining a more synthetic representation, requires deleting from the partition all the edges but those separating regions whose VN are zero and one. The time required is linear with the size of the partition, as is the time for outputting the results.

Adding up, the time bound for the algorithm is $O(n^3 + m \log m)$, while the bound for the algorithm given in [20] was $O(n^4)$. An example of visual hull is shown in Fig. 12, where the active segments are highlighted with thicker lines and each region outside SP is labeled with its visual number.

B. An Algorithm for Computing the 2-D Internal Visual Hull

The basic ideas of this algorithm are similar to those of the previous one. Observe that for obtaining $IVH(SP) \neq SP$ it is not necessary to consider an unconnected SP, as for VH. Let a *visual half-line* be a half-line of p that does not intersect SP. The visual half-lines starting at a point Q can be divided into a number of families (see Fig. 13).

Definition 5: The *internal visual number* $IVN(Q, SP)$ is the number of families of visual half-lines of Q with respect to SP.

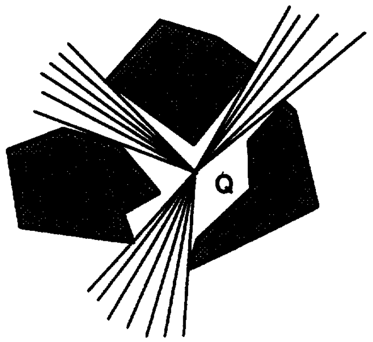
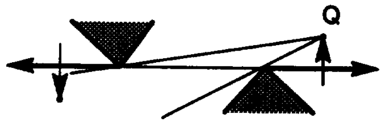
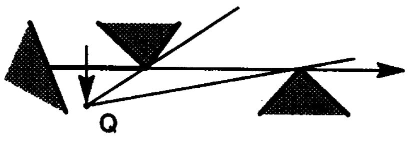


Fig. 13. The internal visual number of Q is three.



(a)



(b)

Fig. 14. Active lines and active segments for the internal visual number.

It is clear that, according to proposition 8, we have the following proposition.

Proposition 12: A point Q belongs to $IVH(SP)$ if and only if it is $IVN(Q, SP) = 0$.

Active segments of *active lines* separate points whose internal visual number differ by one. In Fig. 14 the two cases of active lines are illustrated, with the conventions already used in Fig. 8. Observe that, unlike the previous case, we do not introduce the edges of SP among the active segments. This is because the IVN of a point lying immediately outside SP can assume any integer value. Therefore, crossing the boundary of SP when traversing the partition would require recomputing IVN.

Another difference with the VH case is that only the active segment of case (b) in Fig. 14 can bound IVH. It is easy to understand that the active segments in Fig. 14(a) cannot bound IVH, since when Q crosses the active segment toward the negative side there is always another family of visual half lines in the direction opposite to the family that disappears. This is not the case of the active lines of the VH algorithm, which all can bound VH. Let us observe that segments of type (a), though not possible boundaries of IVH, are necessary for computing IVN with an algorithm that visits the partition in $O(n)$ time.

The outline of the algorithm is similar to the one of the VH algorithm.

1) Compute all the segments active with regard to IVH.
2) Construct a partition of the plane produced by these segments and by the edges of SP using a plane-sweep algorithm.
3) Traverse the partition in order of adjacent regions, computing the internal visual number of each region of the partition and subtracting or adding one to the number of the previous region, according to the crossing direction of each active segment. The boundary of SP should not be crossed. This permits reaching all the regions that are connected together. Any unconnected regions, that is, "holes" in SP, belong to the visual hull.
4) Merge the regions whose IVN is zero.

The steps of the algorithm are similar to those of the VH algorithm and therefore will not be described in detail. One difference is that, after computing the active segments in step 1, we can check whether all the lines belong to case (a) or not. In the former case the algorithm halts, since we have that $IVH(SP) = SP$. This happens for instance to the object in Fig. 12. In addition, it must be observed that, since we do not traverse regions inside SP, we must compute the IVH of the starting region. This requires 1) computing all the lines joining a point P chosen at random in the region with any vertex V_i of SP, and 2) intersecting the lines with the boundary of SP. Any line that intersects SP only at V_i bounds a family of internal visual lines, and therefore the IVN is the half of the number of these lines.

In conclusion, to compute the IVN of the starting region requires $O(n^2)$ time and does not affect the time bound for the whole algorithms, which is $O(n^3 + m \log m)$, as in the VH case.

An example of internal visual hull is shown in Fig. 15,

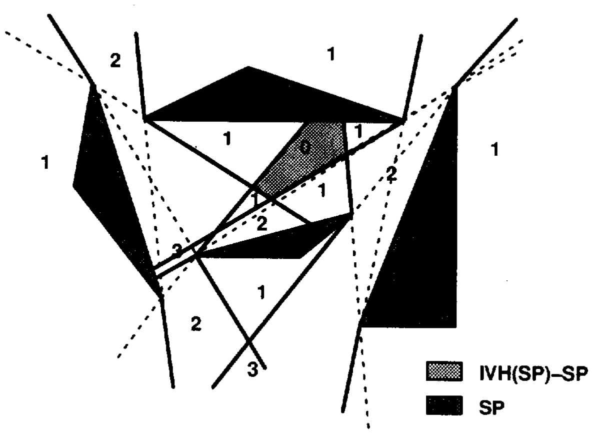


Fig. 15. An example of internal visual hull.

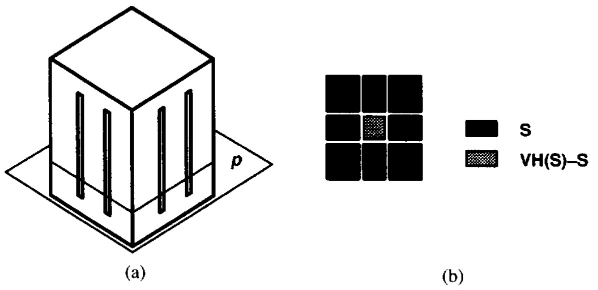


Fig. 16. A solid S (a) and its section (b) with the plane p. The surface of S can be exactly reconstructed, but the object reconstructed is larger than S.

IV. Computing the Silhouette-Active Surfaces of Polyhedral Objects

In this section, after discussing some features of the silhouette-active surface $sa(S)$ of a 3-D object S with respect to image understanding, we present an algorithm for computing $sa(S)$ when S is a polyhedron. For doing this, we will show that it is sufficient to apply the 2-D VH algorithm of the previous section in all the planes supporting a face of S.

The knowledge of the silhouette-active surface of a 3-D object gives partial insight into its behavior with regard to silhouette-based recognition or reconstruction, since $sa(S)$ implies uniquely the visual hull (see proposition 4). For instance, we can state that two objects S' and S'', observed from the viewing region R_c, can be distinguished one from the other using silhouettes iff $sa(S') \neq sa(S'')$, or that an object S can be exactly reconstructed using volume intersection iff $s(S) = sa(S)$.

The latter statement requires some remarks. In general, the visual hull of an object also consists of parts *not connected to S*, which do not belong to S. Therefore, we can have situations where:

1) The surface of S can be exactly reconstructed;
2) The object reconstructed is larger than S, consisting of S plus some unconnected parts.

An example of this situation for a 3-D object is shown in Fig. 16. Of course, the computation of the visual hull of S allows us to discover if the object reconstructed contains such nonconnected parts.

A general approach for computing $sa(S)$ consists of computing $VH(S)$ and finding the common boundary of S and $VH(S)$. Finding $VH(S)$ can be a computationally complex task, as we will see in the following section. We will show in this section that, if S is a polyhedral object, the computation

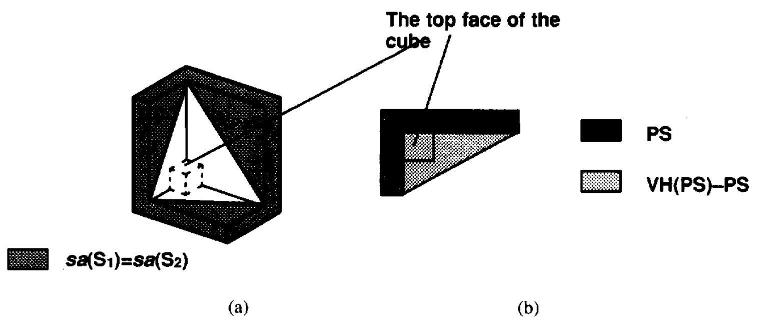


Fig. 17. The silhouette-active surface of both (a) S_1 and S_2 and (b) a section PS of S_1 made by the plane containing the top face of the small cube.

of $se(S)$ can be performed using a simple algorithm based on the 2-D algorithm of the previous section.

As in 2-D, let a *visual line* relative to a polyhedron S be a straight line that does not intersect S; the points lying on a visual line do not belong to the visual hull. Let n be the number of vertexes of S, which has therefore $O(n)$ edges and faces. Finally, let p_i be the plane supporting a face F_i of S and let PS_i indicate the polygonal intersection of p_i and S, excluding F_i itself. For finding which part of F_i, if any, lies on $sa(S)$, let us consider a point Q lying on F_i. Q belongs to $sa(S)$ if there is a visual line passing outside S at an infinitesimal distance from Q; this visual line therefore must be parallel to F_i. We immediately draw the following conclusion.

Proposition 13: A point Q of a face F_i belongs to $sa(S)$ iff it does not belongs to $VH(PS_i)$.

Therefore we can use the following simple algorithm for computing $sa(S)$. For each face,

1) Find PSi, the intersection of p_i and S, minus F_i itself;
2) Compute $VH(PS_i)$, using the 2-D algorithm described in Section III;
3) Determine $F_i - VH(PS_i)$, which is the part of F_i that belongs to $sa(S)$.

In more detail, the computation can be arranged as follows. First, some pruning can be performed by computing in $O(n \log n)$ time [21] the convex hull of S and selecting immediately for membership to $sa(S)$ the faces that belong to the surface of $CH(S)$. For each of the remaining faces, the computation of PS_i can be performed in $O(n)$ time. Some further pruning can be performed by merging steps (2) and (3) above in the following way. When computing the active segments for the 2-D visual hull algorithm in the plane p_i, we consider only the parts of the active segments that are inside F_i. The subsequent steps of the algorithm—that is, computing and visiting the partition—are therefore limited to the region enclosed by F_i, where each region whose visual number is not zero belongs to $sa(S)$.

Visiting each face requires computing a starting visual number. This can be performed in $O(n^2)$ time, computing all the lines joining a point of the starting region with the vertexes, and intersecting each line with the edges of SP. The number of the lines that touch SP only at a vertex, divided by two, gives the starting visual number.

For each face the time required is therefore $O(n^3 + k \log k)$, where k is the size of the partition of the face.

The time bound for the whole algorithm is therefore $O(n^4 + np)$, where p is the average value of $k \log k$. As an example, let us apply this algorithm to the objects S_1 and S_2 of Fig. 2. The resulting surface is the same for both objects and is shown in Fig. 17(a). The faces of the small cube inside the concavity of S_1 do not belong to the active surface, as we can verify for instance from Fig. 17(b), where a section of the object made with the plane supporting the upper face of the cube is shown. Since $sa(S_1) = sa(S_2)$, we have proven formally that we cannot tell S_1 from S_2 using silhouettes obtained from viewpoints outside the convex hull of the objects.

Let us conclude this section observing that, unfortunately, it is not possible to extend the results just obtained for constructing a similar 2-D algorithm for computing $isa(S)$, since the visual half-lines starting at a point whose distance from a face F_i is infinitesimal need not to be parallel to F_i

V. Computing *VH* and *IVH* for Polyhedral Objects

In this section we tackle the problem of computing the visual hulls of unrestricted polyhedra. For doing this, in V–A we extend in 3-D the concept of visual number. In V–B we show that 3-D regions with different visual numbers are separated by planar or quadric surfaces, which can be easily computed from the geometry of the object. Instances of visual hulls of simple polyhedra are presented in V–C. The relationship between visual hull and aspect graph of polyhedra is discussed in V–D. A general algorithm for computing the visual hull of a polyhedron is presented in V–E. As in the 2-D case, the algorithm constructs a partition of the space into zones containing points with the same 3-D visual number, computes the visual numbers, and selects all the zones whose visual number is zero. Finally, in V–F the features of the internal visual hull of a polyhedron are briefly discussed and an algorithm for its computation is outlined.

A. The 3-D Visual Number

For constructing a general algorithm able to compute the visual hull of a polyhedron S, we will extend in 3-D the concept of visual number. $VN3D(Q, S)$, the 3-D visual number of a point Q with respect to S, should be defined in such a way that the following hold true.

1) Its value determines the membership of Q to $VH(S)$.
2) It is possible to compute from the geometry of S a partition of R^3 into regions containing points with the same visual number.

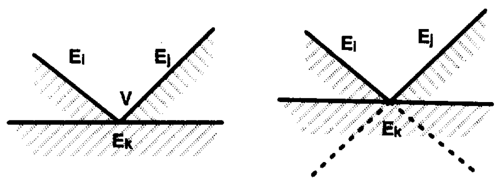


Fig. 18. Examples of two situations in which an edge of the visual cone disappears. In (a) E_i and E_j share the vertex V; in (b) they do not have a common vertex.

Let us consider a point $Q \notin S$ and all the visual lines relative to S passing through P. These lines fill an infinite conical volume that we define as the *visual cone* of Q relative to S.

The boundary of the visual cone consists of a number of infinite planar faces that will be referred to as *visual angles.* Each visual angle is generated by the lines passing through P and tangent to S at an edge E_i, or part of it. We define the 3-D visual number as follows.

Definition 6: $VN3D(Q, S)$ is the number of edges (or faces) of the visual cone of Q relative to S.

It is clear that this definition satisfies the above requirement 1. We have in fact the following.

Proposition 14: $VN3D(Q,S) = 0$ if and only if $Q \in VH(S)$.

In the following we will show that definition 6 also satisfies requirement 2.

B. The Active Surfaces

Let the *active surfaces*, as the active lines in 2-D, be the surfaces that contain *active segments*. The active segments are parts of surfaces that separate points of R^3 whose $VN3D$ are different, and are therefore potential boundaries of $VH(S)$.

Consider an edge E of the visual cone of a point P relative to S, and let E_i and E_j be the edges of S which originate the visual angles that intersect at E. Two cases are possible.

1) E_i and E_j have a common vertex V.
2) E_i and E_j have no common vertex.

For finding which are the active surfaces, we will imagine continuously displacing P until the edge E is removed from the boundary of the visual cone.

In both cases 1 and 2 the edge E disappears when it intersects S at an edge E_k. The difference is that in case 1), E is the line passing through P and V; in case 2), the edge E intersects E_i and E_j at different points for each position of P. Examples of these two situations are shown in Fig. 18(a) and (b), respectively, as seen from a point lying on E on one side of E_i, E_j and E_k.

Let us consider case 1. The active surface for this case is generated by all the lines that pass through the vertex V and the edge E_k (or part of it) and do not share with S any other point, but for the case in which V and E_k belong to the same face of S. If this happens, the lines that form the active surface share some segment with this face. Let the active infinite planar surfaces relative to case 1 be the VE surfaces.

In case 2, the active surface is generated by all the lines tangent to S at three edges E_i, E_j, and E_k. It is a curved surface. In fact, from basic 3-D analytic geometry we have that the lines passing through three lines skew to each other form a hyperboloid of one sheet or a hyperbolic paraboloid. The second case takes place when the three lines are all parallel to a plane. We define these curved active surfaces as EEE surfaces.

In conclusion, we have found two kinds of surfaces that are active with respect to $VN3D$ and are therefore potential boundaries of $VH(S)$. All these surfaces are ruled surfaces, generated by lines tangent to S and sharing with it at least two points. A main result of the previous discussion is as follows.

Proposition 15: The boundary of the visual hull of a planar-face object in general is not planar and consists of planar and quadric patches.

Actually, not all the VE and EEE surfaces that are active with respect to $VN3D$ can also be boundaries of VH. As an example, consider the two cases of Fig. 19(a) and (c). The edges of S that determine the active surfaces are observed from a viewpoint lying on a line L of the active surface and are projected onto an image plane normal to this line. In the first case, relative to a VE surface, let us analyze the situation in the plane q passing through L, whose trace in the image plane is shown in Fig. 19(a). This situation is shown in Fig. 19(b), where the dotted lines indicate visual lines passing on both sides of every part of L. Therefore, L and the whole VE surface cannot belong to the boundary of $VH(S)$. In the second case we analyze three 2-D situations, in three planes a, b and c passing through L. It is clear (see Fig. 19(d)) that in plane a we can construct visual lines on both sides of the external segments of L, in plane b on both sides of the segment E_2E_3, and in plane c on both sides of the segment E_1E_2. Therefore neither L nor the EEE surface that contains L can be boundaries of $VH(S)$. We will not analyze here in detail all the possible cases, since this is not necessary for the brute-force algorithm for computing $VH(S)$ that we will present later in this paper.

C. Examples of Visual Hulls of Simple Polyhedra

This subsection is devoted to clarifying the items presented so far by discussing some simple examples. Let us consider first the polyhedron S_3, one of the objects in Fig. 3. In Fig. 20(a) S_3 is shown together with the part of the active surface due to E_1 and V_1 that is inside the convex hull. When a point crosses this surface in the direction shown in figure, its $VN3D$ changes from 0 to 3. In fact, a visual cone is generated that is bounded by the three visual angles relative to E_1, E_2, and E_3. Therefore, this surface is a boundary of $VH(S_3)$, which is shown in Fig. 20(b). In the same figure another active surface is shown. It is the VE surface due to E_3 and V_3. Crossing this surface in the direction shown in the figure, $VN3D$ is incremented by one, since the visual cone acquires a new visual angle relative to E_4.

Let us now consider the objects S_1 and S_2 of Fig. 2. We have already found their silhouette-active surface (see Fig. 17). It is not difficult to determine the visual hull shown in Fig. 21(a). There are many active surfaces, some of them overlapping because of parallel edges. For instance, the surface pointed at by the arrow in the figure is due to two active VE

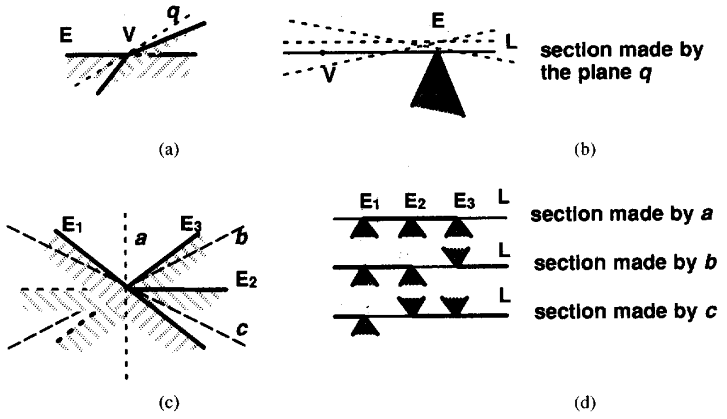


Fig. 19. (a) A VE surface and its section made by (b) the plane q, showing that the surface cannot bound $VH(S)$. Also, the EEE surface in (c) cannot bound $VH(S)$, as is shown in (d).

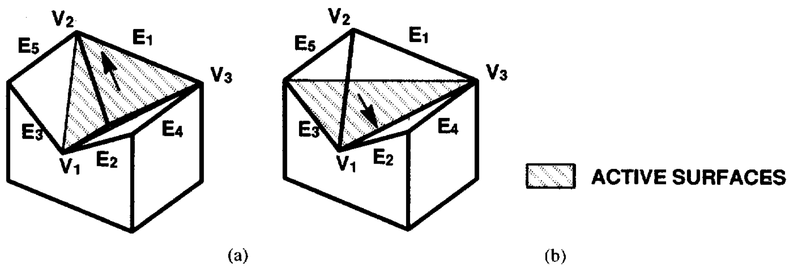


Fig. 20. The polyhedron S_3 and an active surface (a) that adds three to $VN3D$; $VH(S_3)$ and another active surface (b), which adds 1 to $VN3D$.

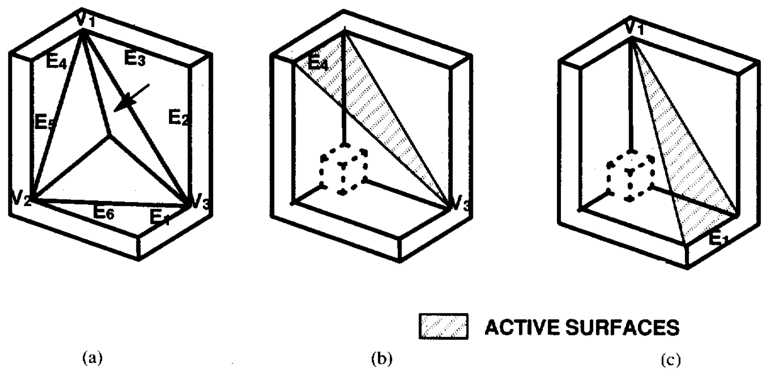


Fig. 21. (a) The visual hull of both S_1 and S_2. The face pointed at by the arrow is generated by two partially overlapping active surfaces, shown in (b) and (c).

surfaces generated by the pairs vertex-edge (V_1, E_1), (V_3, E_4) (see Fig. 21(b) and (c)). Crossing this surface in the direction opposite to the arrow changes $VN3D$ from 0 to 4.

Another object with a slightly more complex concavity is shown in Fig. 22(a). To determine by inspection its visual hull, shown in Fig. 22(b), takes some more reflection.

The objects so far considered have no lines tangent at three points, and therefore no cubic patch belongs to the boundary of their visual hulls. A cubic patch, on the other hand, is present in the boundary of the visual hull (Fig. 23(b)) of the polyhedron in Fig. 23(a). The boundary of the visual hull consists of four patches, P_1, P_2, P_3, and P_4, in addition to the silhouette-active surface. Some of the planar patches are generated by more than one active surface because of parallel edges. P_3 is a segment of the quadric surface generated by the lines tangent at E_1, E_2, and E_3.

D. The Visual Hull and the Aspect Graphs

In this subsection we will discuss briefly the relationship between the visual hull and the aspect graph. The aspect graph and its computation have recently become popular research topics. In short, the aspect graph of a 3-D object stores at

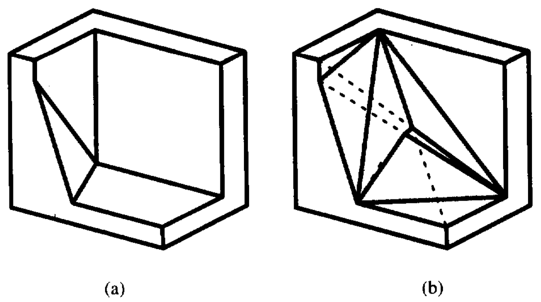


Fig. 22. (a) An object with a more complex concavity and (b) its visual hull. Some dotted lines of construction are also shown.

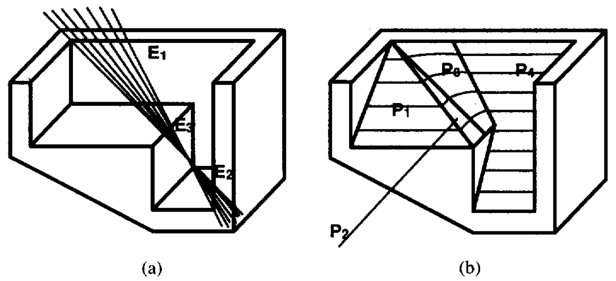


Fig. 23. (a) A polyhedron S and (b) its visual hull. P_3 is a segment of a quadric surface produced by the lines tangent to S at the edges E_1, E_2, and E_3.

the vertexes of a graph all the topologically distinct line drawings of an object. The edges of the graph represent visual events—that is, topological changes in the views. Algorithms for computing the aspect graph have been given for polygons, polyhedra, solids of revolution, and other curved objects. For further details on this matter, the interested reader is referred to [22] and [24], quoted elsewhere in this paper, and to the survey paper [25] of Bowyer and Dyer.

Here we will point out that the active surfaces that we have found in relation to the 3-D visual number are well known in the field of the computation of the aspect graph of polyhedral objects. In fact, VE and EEE surfaces are a subset of the surfaces that divide the viewing space into regions such that each viewpoint of the region gives topologically equivalent views. More specifically, the active surfaces relative to $VN3D$ are the same surfaces relative to the aspect graph that are due to the visual events on the boundary of the line drawing of the polyhedron. We could also say that these surfaces generate the *silhouette* or *contour aspect graph* of an object.

E. A Brute-Force Algorithm for Computing the Visual Hull

The computation of the visual hull of a polyhedron S can be performed as follows.

1) Compute all the *potentially* active surfaces.
2) Construct the partition of R^3 generated by the potentially active surfaces.
3) Compute $VN3D$ for each cell of the partition.
4) Merge together all the cells whose visual number is zero.

In the first step $O(n^3)$ surfaces are computed in $O(n^3)$ time, considering $O(n^2)$ $V-E$ pairs and $O(n^3)$ EEE triplets. m algebraic surfaces divide R^3 in $O(m^3)$ cells. In our case, therefore, the partition has $O(n^9)$ cells, faces, edges, and vertexes. The construction of the partition can be performed in $O(n^9 \log n)$ time using an incremental algorithm, as shown by Plantinga and Dyer in [24].

The $VN3D$ of a point Q chosen at random in a cell can be computed as follows. Compute first in $O(n^2)$ time the polygonal intersections INT_i of $O(n)$ potential visual angle with the $O(n)$ faces of S. For each visual angle, compute the 2-D visual number of Q relative to INT_i. $VN3D$ is obtained by adding up all the 2-D visual numbers. Since the computation of each 2-D visual number takes $O(n^2)$ time, the $VN3D$ of each zone is obtained in $O(n^3)$ time.

The bound for the computation time of step 3 results therefore in $O(n^{12})$, which is also the bound for the whole algorithm, since step 4 can be performed in time linear with the size of the partition. We will not mention some trivial pruning actions that do not affect the time bound of the algorithm.

F. The Computation of the Internal Visual Hull

We can deal with the internal visual hull of a polyhedral object S as we did with the external visual hull, introducing some suitable modifications. Let us first define the *internal* 3-D visual number. Consider a point $Q \notin S$ and all the *visual* half-lines relative to S, starting at P. A visual half-line is a half-line that does not intersect S. These lines fill an infinite half-conical volume that we define to be the *visual half-cone* of Q relative to S.

The boundary of the visual half-cone, which is coincident with the circumscribed cone, consists of a number of infinite planar faces that will be referred to as *visual half-angles*. Each visual half-angle is generated by the half-lines starting at Q and tangent to S at an edge E_i, or part of it. We give the following definition.

Definition 7: $IVN3D(Q, S)$, the internal 3-D visual number of a point Q relative to S, is the number of edges (or faces) of the circumscribed cone.

The following proposition becomes clear.

Proposition 15: $IVN3D(Q, S) = 0$ if and only if $Q \in IVH(S)$.

Since the potentially active surfaces relative to $IVN3D$ are the same surfaces of the $VN3D$ case, we can use the same brute-force algorithm for computing IVH, the only difference being the computation of $IVN3D(Q, S)$ instead of $VN3D(Q, S)$ for a point Q of each cell. It is easy to see that this computation can be performed by adding $O(n)$ 2-D internal visual numbers, each relative to Q and to the intersections of S with each potential visual half-angle. Therefore the computation of $IVN3D$ takes $O(n^3)$ time, so the overall time bound of the algorithm is $O(n^{12})$ again.

VI. Conclusion

We have addressed the problem of finding which parts of a nonconvex object are relevant for silhouette-based image understanding. A new geometric entity, the *visual hull*, has been introduced. We have shown that the knowledge of the visual hull of an object provides a full solution to this problem. The visual hull $VH(S, R)$ of an object S is the maximal object silhouette-equivalent to S with respect to a viewing region R. $VH(S,R)$ also is the closest approximation of S that can be obtained applying the volume intersection technique with viewpoints belonging to R.

Analyzing the possible viewing regions, the existence of a main case relative to the *external*, or *natural*, visual hull $VH(S)$ has been recognized. $VH(S)$ is the visual hull relative to the region R_c, which is complementary to the convex hull of S; it also is the visual hull of any viewing region that is a subset of R_c and completely encloses S. In addition, the visual hull relative to any region that is a subset of R_c can be constructed starting from $VH(S)$.

The case of the largest possible viewing region, bounded by S itself, has been also considered. The corresponding *internal* visual hull $IVH(S)$ also allows us to find which features of the surface of S cannot appear on any image of the object obtained from viewpoints not belonging to the convex hull.

Two efficient algorithms have been described for computing the visual hull and the internal visual hull of 2-D objects. These algorithms divide the plane into regions that either entirely belong to the visual hulls or to its complement. The time bound of both these algorithms is $O(n^3 + m \log m)$, where m is the size of the partition and n is the number of algebraic curves defining the boundary of the object. The 2-D algorithms allow us to find the visual hulls of $2\frac{1}{2} - D$ sweep solids and provide the basis for an algorithm described elsewhere for dealing with solids of revolution.

The silhouette-active surface of a 3-D object gives a partial insight into the behavior of the object under silhouette-based recognition algorithms. For instance, two objects can be discriminated using silhouettes if their silhouette-active surfaces are different. We have shown that for a polyhedron the external silhouette-active surface can be computed applying the 2-D algorithm in each of the planes supporting a face of S. No similar algorithm is possible for the internal silhouette-active surface.

We have also addressed the problem of computing the whole external visual hull and the internal visual hull of a general polyhedron. By extending in 3-D some concepts used in the 2-D algorithm, we have been able to find which surfaces are potential boundaries of both visual hulls. The result is planar and ruled quadric surfaces, so that the boundary of the visual hull of a general polyhedron may be nonplanar. These surfaces also are a subset of the surfaces used by the algorithms for computing the aspect graph of a polyhedron. Brute-force algorithms for computing in $O(n^{12})$ time the visual hulls of general polyhedra have been given.

To be able to compute the visual hull of objects with curved surfaces would enhance the effectiveness of the visual hull idea. This topic is currently under investigation, together with efficient algorithms for the polyhedral case.

References

[1] J. Aloimonos, "Visual shape computation," *IEEE Proc.*, vol. 76, no. 8, pp. 899–916, 1988.
[2] P. J. Besl and R. C. Jain, "Three-dimensional object recognition," *Comput. Surveys*, vol. 17, pp. 75–145, 1985.
[3] C. Chien and J. K. Aggarwal, "Model reconstruction and shape recognition from occluding contours," *IEEE Trans. Patt. Anal. Machine Intell.*, vol. 11, pp. 372–389, 1989.
[4] T. P. Wallace and P. A. Wintz, "An efficient three-dimensional aircraft recognition algorithm using normalized Fourier descriptors," *Comput. Graphics Image Processing*, vol 13, pp. 99–126, 1980.
[5] M. Hebert and T. Kanade, "The 3-D profile method for object recognition," in *Proc. IEEE Computer Society Conf. Computer Vision and Pattern Recognition*, pp. 458–463, June 1985.
[6] A. P. Reeves *et al.*, "Three-dimensional shape analysis using moments and Fourier descriptors," *IEEE Trans. Patt. Anal. Machine Intell.*, vol. 10, pp. 937–943, 1988.
[7] S. A. Dudani *et al.*, "Aircraft identification by moment invariants," *IEEE Trans. Comput.*, vol. C-26, pp. 39–46, 1981.
[8] Y. F. Wang, M. J. Magee, and J. K. Aggarwal, "Matching three-dimensional objects using silhouettes," *IEEE Trans. Patt. Anal. Machine Intell.*, vol. PAMI-6, pp. 513–518, 1984.
[9] M. Brady and A. Yuille, "An extremum principle for shape from contour," *IEEE Trans. Patt. Anal. Machine Intell.*, vol. PAMI-6, pp. 288–301, 1984.
[10] W. N. Martin and J. K. Aggarwal, "Volumetric description of objects from multiple views," *IEEE Trans. Patt. Anal. Machine Intell.*, vol. PAMI-5, pp. 150–158, 1983.
[11] P. Srinivasan *et al.*, "Computational geometric methods in volumetric intersection for 3D reconstruction," *Patt. Recogn.*, vol. 23, pp. 843–857, 1990.
[12] M. Potemesil, "Generating octree models of 3D objects from their silhouettes in a sequence of images," *Comput. Vision Graphics Image Processing*, vol. 40, pp. 1–29, 1987.
[13] H. Noborio *et al.*, "Construction of the octree approximating three-dimensional objects by using multiple views," *IEEE Trans. Patt. Anal. Machine Intell.*, vol. 10, pp. 769–782, 1988.
[14] C. H. Chien and J. K. Aggarwal, "Volume/surface octrees for the representation of three-dimensional objects," *Comput. Vision, Graphics and Image Processing*, vol. 36, pp. 100–113, 1986.
[15] V. Cappellini *et al.*, "From multiple views to object recognition," *IEEE Trans. Circuits Syst.*, vol. CS-34, pp. 1344–1350, 1987.
[16] N. Ahuja and J. Veenstra, "Generating octrees from object silhouettes in orthographic views," *IEEE Trans. Patt. Anal. Machine Intell.*, vol. 11, pp. 137–149, 1989.
[17] T. H. Hong and M. Schneier, "Describing a robot's workspace using a sequence of views from a moving camera," *IEEE Trans. Patt. Anal. Machine Intell.*, vol. PAMI-7, pp. 721–726, 1985.
[18] K. Shanmukh and A. K. Pujari, "Volume intersection with optimal set of directions," *Patt. Recogn. Lett.*, vol. 12, pp. 165–170, 1991.
[19] A. Laurentini, "The visual hull: A new tool for contour-based image understanding," in *Proc. 7th Scandinavian Conf. Image Analysis*, pp. 993–1002, 1991.
[20] ——, "The visual hull of solids of revolution," in *Proc. 11th IAPR*, The Hague, Netherlands, Aug. 30–Sept. 3, 1992, pp. 720–724.
[21] F. Preparata and M. Shamos, *Computational Geometry*. New York: Springer, 1985
[22] Z. Gigus, J. Canny, and R. Seidel, "Efficiently computing and representing aspect graphs of polyhedral objects," *IEEE Trans. Patt. Anal. Machine Intell.*, vol. 13, pp. 542–551, June 1991.
[23] S. Baase, *Computer Algorithms*. New York: Addison-Wesley, 1988.
[24] H. Plantinga and C. R. Dyer, "Visibility, occlusion and the aspect graph," *Int. J. Comput. Vision*, vol. 5, no. 2, pp. 137–160, 1990.
[25] K. W. Bowyer and C. R. Dyer, "Aspect graphs: An introduction and survey of recent results," *Int. J. Imaging Syst. Technol.*, vol. 2, pp. 315–328, 1990.

Aldo Laurentini was born in Genova, Italy, on November 13, 1940. He received the degree in Ingegneria Elettronica from the Politecnico di Milano, Milan, Italy, in 1963.

In 1965 he joined the Politecnico di Torino, Turin, Italy, where he is Professor of Computer Science in the Dipartimento di Automatica ed Informatica. His current research interests are in the areas of computer vision, computer graphics, and document understanding.