ances match the actual errors. The results show that it is possible to achieve subpixel accuracy in the range of 0.18 pixel RMS error. Compared to the 0.3–0.4 pixel RMS error achieved in [5] this is a significant improvement.

Note that the same results are valid for cases B and C, also, because the optimal target and noise pixel detection probabilities are the same for the first three cases. In [5], the pixel RMS error depends on detection threshold levels and, hence, target and noise pixel detection probabilities, whereas in the present work, the optimal thresholds obtained using *a priori* information give rise to almost the same pixel detection probabilities irrespective of the initial threshold levels.

For case D the real images were obtained by a FPA based platinum-silicide camera (resolution 256×256) in infrared light. The 256×256 images were reduced to 64×64, with two frames shown in Fig. 2. The images show two cars moving from left to right during daytime. The marked car (on the left) was chosen for tracking. The parameters of the segmentation algorithm are shown in the last row of Tables 1 and 2.

TABLE V.
FILTER ESTIMATED RMS ERRORS.

Case	Position RMS Error	Velocity RMS Error
Present technique	0.2	0.031
Previous technique	0.4	0.090

Using the PDAF, the estimated values of the centroid position and velocity of the target along with the estimated accuracies are given in Fig. 3. The RMS errors yielded by the tracking filter using the optimal layering technique are compared with [5] in Table V. The filter-calculated RMS estimation errors were 0.2 pixel for each coordinate in position and 0.031 pixel/frame in velocity, whereas in [5] the position and velocity RMS errors were 0.4 and 0.09, respectively. Since the ground truth is not known, in order to check the filter's consistency (validity of the filter-calculated estimation accuracy), the 10-calculated and was found to be 2.24, which is within the 95% chi-square limits [0.96, 3.14] based on

$$\frac{\chi^2_{20}}{10}.$$

Thus, the filter-calculated accuracies are reliable.

These results show that, in spite of the non-white and non-Gaussian noise in the image, the method was remarkably successful for this real image.

REFERENCES

[1] Y. Bar-Shalom, *IMDAT – Image Data Association Tracker 3.0*, Interactive Software, 1993.

[2] Y. Bar-Shalom and C.R. Li, *Multitarget-Multisensor Tracking: Principles and Techniques*, YBS Publishing, Storrs, Conn., 1995.

[3] Y. Bar-Shalom, H.M. Shertukde, and K.R. Pattipati, "Use of measurements from an imaging sensor for precision target tracking," *IEEE Trans. on Aerospace and Electronic Systems*, vol. 25, pp. 863-871, November 1989.

[4] A. Kumar, Y. Bar-Shalom, and E. Oron, "Image segmentation based on optimal layering for precision tracking," *DIMACS Workshop Proc.*, 1994.

[5] E. Oron, A. Kumar, and Y. Bar-Shalom, "Precision tracking with segmentation for imaging sensors," *IEEE Trans. on Aerospace and Electronic Systems*, vol. 29, pp. 977-987, July 1993.

[6] D.R. VanRheeden and R.A. Jones, "Noise effects on centroid tracker aim point estimation," *IEEE Trans. on Aerospace and Electronic Systems*, vol. 24, March 1988.

[7] J. Zupan, *Clustering of Large Data Sets*, Academic Press, 1973.

How Far 3D Shapes Can Be Understood from 2D Silhouettes

Aldo Laurentini

***Abstract*— Each 2D silhouette of a 3D unknown object O constrains O inside the volume obtained by back-projecting the silhouette from the corresponding viewpoint. A set of silhouettes specifies a boundary volume R, obtained by intersecting the volumes due to each silhouette. R more or less closely approximates O, depending on the viewpoints and the object itself. This approach to the reconstruction of 3D objects is usually referred to as volume intersection. This correspondence addresses the problem of inferring the shape of the unknown object O from the reconstructed object R. For doing this, we divide the points of the surface of R into *hard points,* which belong to the surface of any possible object originating R, and *soft points,* which may or may not belong to O. We consider two cases: In the first case R is the closest approximation of O which can be obtained from its silhouettes, i.e., its visual hull; in the second case, R is a generic reconstructed object. In both cases we supply necessary and sufficient conditions for a point to be hard and give rules for computing the hard surfaces.**

***Index Terms*— Computer vision, 3D object reconstruction, 2D images, volume intersection, shape from silhouettes, visual hull.**

I. INTRODUCTION

Reconstructing 3D shapes from 2D images is an important area of research in computer vision. The silhouettes of a 3D object are important sources of shape information, which can usually be obtained from intensity images with simple and robust algorithms. The word silhouette indicates the region of a 2D image of an object **O** which contains the projections of the visible points of **O**.

If no *a priori* knowledge about **O** is available, all the information provided by a silhouette S_i is that **O** must lie in the solid region of space $\mathbf{C}_i$ obtained by back-projecting S_i from the corresponding viewpoint $\mathbf{V}_i$. If *n* silhouettes are available, they constrain **O** within the volume $\mathbf{R}_n$:

$$\mathbf{R}_n = \bigcap_{i=1}^{n} \mathbf{C}_i$$

Many object reconstruction algorithms based on this idea have been presented [1]-[13]. They are usually referred to as *volume intersection (VI)* algorithms (Fig. 1) and compute from a set of silhouettes a more or less precise approximation of **R**. Several VI algorithms specify the reconstructed object **R** as an octree ([2], [5], [11]). Other representations have also been used ([1], [8]). The main feature of this popular approach is that it does not compel us to find correspon-

Manuscript received November 30, 1992; revised August 23, 1994. Recommended for acceptance by Dr. T. Henderson.

A. Laurentini is with Dipartimento di Automatica ed Informatica, Politecnico di Torino, C.so Duca degli Abruzzi, 24, 10129 Torino Italy, e-mail laurentini@polito.it.

IEEECS Log Number P95022.

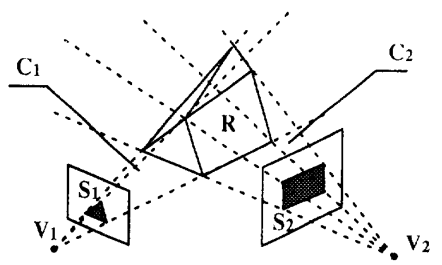

Fig. 1. The volume intersection approach for reconstructing 3D objects.

dences between multiple images. For the purpose of this paper we assume that VI is an ideal operation able to exactly produce **R**.

The regions $\mathbf{C}_i$ are solid cones for perspective projection, solid cylinders for parallel projections. In both cases the regions are bounded by ruled surfaces, referred to as the *circumscribed cones* or *cylinders* of **O**. In the following, for simplicity we will speak of cones, conical surfaces, and viewpoints referring both to perspective and parallel projections.

The bounding volume **R** more or less closely approximates **O**. Obviously, the final goal of VI is the shape of the unknown object. Thus, a fundamental problem raised by this technique is to infer as far as possible the shape of **O** from the reconstructed volume **R**. For dealing with this problem we will use the recently introduced geometric concept of *visual hull* of an object.

This correspondence is organized as follows. In Section II we briefly recall the visual hull idea. In Section III we tackle the problem of inferring the shape of the unknown object in the case of optimal reconstruction, that is starting from the closest approximation of **O** which can be obtained from its silhouettes. In Section IV we find out which shape information is provided by a generic reconstructed volume **R**, or, in other words, from a generic set of silhouettes. In Section V we summarize the results obtained and present concluding remarks.

II. The Visual Hull of a 3D Object

The visual hull is a geometric entity useful for dealing with silhouette-based image understanding [14], [15]. Here we summarize the relevant items of [15], to which the reader is referred for further details.

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

From this definition it follows that:

Proposition 1. **VH (O, V)** is the closest approximation of **O** that can be obtained using VI techniques with viewpoints $V \in \mathbf{V}$.

Thus, an object **O** can be exactly reconstructed by VI using silhouettes observed from a viewing region **V** if and only if it is **O** = **VH (O, V)**.

It can be shown that there is a unique visual hull, the *external* visual hull, or simply the visual hull **VH(O)**, for all viewing regions which completely enclose **O** without entering its convex hull. This is the case of main practical interest, since usually the object to be recognized or reconstructed lies at some distance and can assume any orientation with respect to the viewpoints. The visual hull relative to $\mathbf{V} = \mathbf{E}^3 - \mathbf{O}$ is defined as the *internal* visual hull, and denoted **IVH(O)**. Convex hull, visual hull, internal visual hull, and **O** are related by the following inequalities:

Proposition 2. $\mathbf{O} \le \mathbf{IVH(O)} \le \mathbf{VH(O)} \le \mathbf{CH(O)}$.

Algorithms for computing **VH(O)** and **IVH(O)** have been given for polygonal sets, polyhedra [15], and solids of revolution [14].

III. Inferring the Shape of the Unknown Object in the Optimal Case

The purpose of this section is to investigate which information is supplied about the unknown object **O** by the VI technique in the *optimal case*, that is, when the closest possible approximation of **O** has been reconstructed from its silhouettes. Since the visual hull is the closest approximation of **O** that we can construct from its silhouettes, we address the following problem: *What can be inferred about the shape of an object from its visual hull?* This is the inverse problem of finding the visual hull of an object.

The preliminary discussion of a simple example will provide us with some insight into the matter. Various objects with the same visual hull, a cube, are shown in Fig. 2. The objects are a finite sample of an infinite set of silhouette-equivalent objects, ranging in volume from the cube itself to zero-volume, origami-like objects like that shown in Fig. 2(e). All these objects must not exceed the bounding cube, but what else can be inferred about their shape? An intuitive analysis shows that:

- All these objects share the 12 edges of the cube with the surface of the visual hull.
- The remaining surface of these objects can take any shape inside the cube, as long as no line passes through the cube without intersecting this surface (this does not mean that these surfaces must be connected).

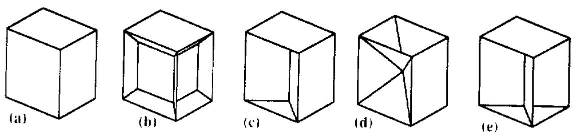

Fig. 2. All objects (a), (b), (c), (d), and (e) have the same visual hull, coincident with object (a). Object (e) is origami-like.

The example shows that, in general, using the information provided by the best reconstructable object, we are unable to completely infer the real shape of the unknown object. However, analyzing the visual hull of Fig. 2, we have seen that some points of its surface (viz., the edges) are also points of the surface of any possible object **O** originating on the visual hull, and other points (viz., the interior points of the faces) may or may not belong to **O**. This leads to the following general definition:

Definition 2. Let P be a point of the surface of a visual hull **VH**. P will be specified as:

- a *hard* point if it also belongs to the surface of any possible object **O** originating **VH**
- a *soft* point otherwise.

In other words, a soft point may or may not belong to the surface of **O** without for that reason affecting the visual hull. Now, the problem is the following: *In general, which points of the surface of a visual hull are hard and which are soft?*

Let us recall that [15]:

Proposition 3. A point P belongs to **VH(O)** if and and only if any line passing through P shares at least a point with **O**.

On the basis of this statement, the following theorem can be demonstrated:

Proposition 4. A necessary and sufficient condition for a point P belonging to the surface of a visual hull **VH(O)** to be hard is that at least one line L passes through P without intersecting **VH(O)** at any other point (Fig. 3).

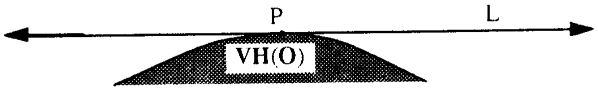

Fig. 3. A necessary and sufficient condition for a point P of the surface of **VH(O)** to belong to **O**. L does not intersect either **VH(O)** or, therefore, **O** at any other point.

Proof. Let us show that the condition is sufficient. Proposition 3 states that every line passing through a point P of the visual hull must share at least a point with **O**. If line L shares only point P with the visual hull, the point which line L must share with **O** is P itself since **O** is enclosed by the visual hull (see Proposition 2).

Let us show that the condition is necessary. This is equivalent to showing that a point P which does not satisfy the condition is soft. A line L through P which does not satisfy the condition shares with **VH(O)** at least one other point Q_L. We can assume, without affecting the visual hull, that for each line L the point Q_L belongs to **O**. With this assumption, P satisfies the condition of Proposition 3, and belongs therefore to **VH(O)**, whether it belongs to **O** or not.

By using Proposition 4 we are able (at least in principle) to divide the surface of the visual hull into a *hard part*, which is coincident with the surface of the original object, and a *soft part*, which is only an outer bound for **O**. In other words, we are able to find the part of the visual hull where unreconstructable concavities may lie. Let us observe that, in spite of its name, volume intersection only provides precise information on the surface of the unknown object.

Another general feature of the object inferred from the visual hull should be emphasized: we call it the *opaqueness property*. It states that, even if the surface of the unknown object **O** can assume many different shapes within the boundary of the soft surface of the visual hull, *no visual ray can cross a soft surface without intersecting **O**.*

We will now show how Proposition 4 can be used for obtaining the hard surfaces for some types of visual hull.

A. Polyhedral Surfaces

Polyhedral visual hulls can be originated by polyhedral objects, although some polyhedra do not yield polyhedral visual hulls (see [15]). In this subsection we discuss how to determine the hard points of a polyhedral surface. It is easy to see that:

Proposition 5. The interior points of a planar face of a visual hull are soft. (Formal proofs of the statements of this section can be found in [17].)

Obviously, Proposition 5 holds for any kind of visual hull, not only polyhedra. From this proposition it follows that *only the edges of a polyhedral surface can consist of hard points.* Let us consider concave edges. It can be shown that:

Proposition 6. Any interior point of a concave edge of a polyhedral surface of a visual hull is soft (see [17])

In conclusion, to find the hard edges we need to consider only the convex edges. This task is made easier by the following proposition:

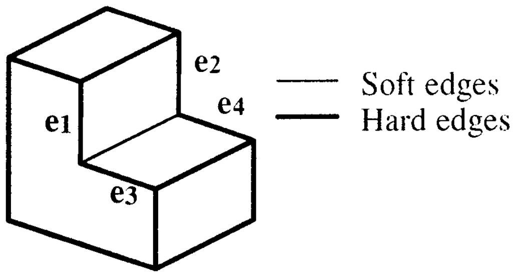

Fig. 4. A polyhedral visual hull and is hard and soft edges.

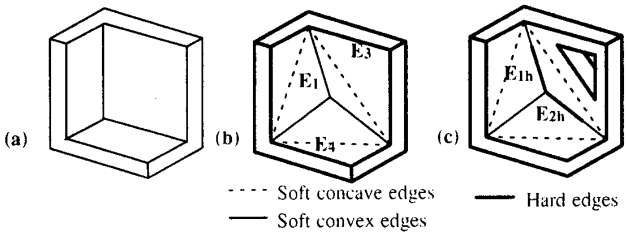

Fig. 5. The visual hull (b) of object (a) has three soft convex edges. A visual hull (c) where two convex edges are partially hard.

Proposition 7. A sufficient condition for a (convex) edge of a polyhedral surface of a visual hull **VH** to be hard is that it also be an edge of the convex hull of **VH** (see [17]).

The convex hull of a polyhedron with n vertices can be computed in time $O(n \log n)$ [16]. Let us consider some simple examples. Proposition 7 allows us to immediately find the 12 hard edges of the cubical visual hull of Fig. 2, and all but four (e_1, e_2, e_3, e_4) of the hard edges of the object in Fig. 4. In this case, it is easy to verify by inspection that all convex edges are hard. However, in general, not every convex edge is hard, as can be verified by inspecting the object in Fig. 5(a). The visual hull of this object (Fig. 5(b)) has three soft convex edges. If we admit exceptional alignment conditions, a convex edge might be only partially hard, as for the visual hull shown in Fig. 5(c). The underlying object is equal to that shown in Fig. 5(a), except for a triangular hole in the right wall. In this case, there are lines passing through E_{1h} and E_{2h} and the triangular hole which satisfy the condition of Proposition 4, and, therefore, E_{1h} and E_{2h} are hard.

An algorithm for computing in $O(n^6)$ time all hard edges or their hard parts is presented in the Appendix. It is opportune to remark that it implies perfect geometry, since it requires checking whether an edge exactly belongs to a planar surface. It is clear that practical algorithms should make allowance for errors relative to the exact position of the edges.

B. Curved Surfaces

Let us consider a non-planar surface of a visual hull. The following proposition holds:

Proposition 8. Let P be a point belonging to a non-planar surface of a visual hull **VH**. A sufficient condition for P to be hard is that P also belongs to the surface of the convex hull of **VH** (see [17]).

This sufficient condition, together with the previous results, makes it possible to easily find the hard part of many visual hulls, for instance the dice-shaped visual hull of Fig. 6,

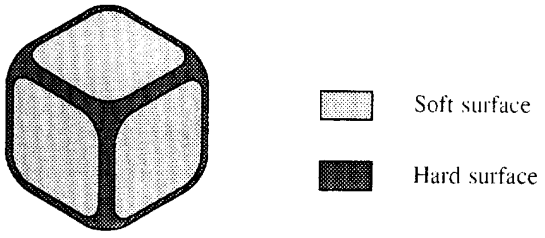

Fig. 6. The hard and soft surfaces of a dice-shaped visual hull.

Let us assume now that the visual hull has a smooth curved surface, such that a plane tangent at each point of the surface can be found. In principle, for determining whether a point P is hard or soft, we can proceed as follows. Let *p* be the plane tangent at P to **VH(O)**. This plane contains all the lines passing through P which are possible candidates for satisfying the condition of Proposition 4. The problem is therefore reduced to the following: Find the intersection **I** of *p* and **VH(O)** (excluding the point P itself) and verify whether there are lines passing through P which do not intersect **I**. This is the same problem as verifying whether P is enclosed by **VH(I)**, the 2D visual hull of **I**, or not.

It is beyond the scope of this paper to discuss any further the computation of the hard part of curved visual hulls. One reason is that it would require a deeper knowledge about the geometry of the visual hull of general curved objects, a subject which is currently under investigation. Another reason is that such algorithms would be mainly of theoretical interest. In fact, it is clear that visual hulls with curved surfaces require infinite intersections to be constructed, except for those special objects whose surface consists of conical and cylindrical patches.

IV. Inferring the Shape from a Generic Set of Silhouettes

The discussion so far presented about the unknown object **O** implies the knowledge of **VH(O)**, and stands as the most favorable case for inferring the shape of **O**. In practice, if, in addition to the silhouettes no other information about **O** is available, we may be unable to obtain the visual hull of an object. An inconvenient, but nevertheless likely, situation is that we do not know whether the object **R** obtained by volume intersection is the visual hull or not. In this section we will find out which information about **O** is provided by a generic object **R** constructed with VI. In other words, we discuss the shape of a 3D object when a generic set of its silhouettes is known.

Let us first consider the simple case of Fig. 7. From the three square silhouettes S_x, S_y, and S_z, obtained with viewing directions parallel to the coordinate axes, we reconstruct a cubic volume **R**, the same object considered in the previous section for an example of optimal reconstruction. It is clear that there are infinite objects enclosed by this volume which are able to produce the same three silhouettes.

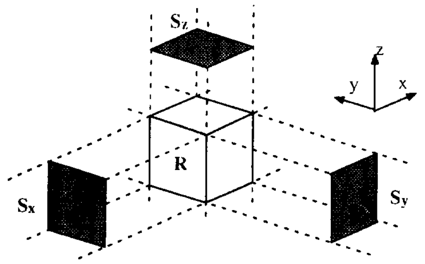

Fig. 7. The volume **R** constructed from the square silhouettes S_x, S_y, and S_z, obtained with viewing directions parallel to the coordinate axes.

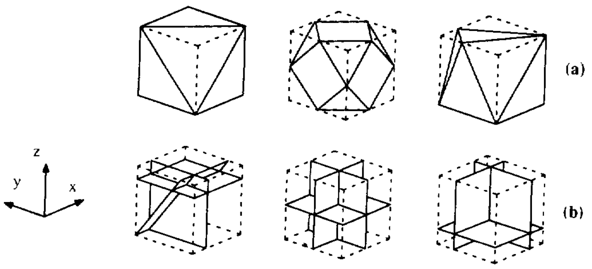

Fig. 8. Three polyhedra (a) and three zero-volume, planar face objects (b) which may yield the silhouettes S_x, S_y, and S_z of Fig. 7.

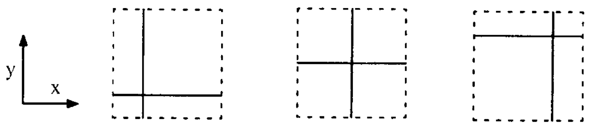

Fig. 9. The intersection of the three origami-like objects of Fig. 8(b) and the top face of the reconstructed cube.

Among them there are many subclasses with infinite members, such as convex polyhedra, and origami-like, planar face objects. Three objects of the former class and three of the latter are shown in Fig. 8(a) and (b) respectively. Also, objects with curved faces can produce the same silhouettes. In terms of visual hull, the object **R** is the visual hull **VH(O, Z)** of the unknown object **O** relative to a viewing region **Z** consisting of the three ideal points of the coordinate axes, and therefore is larger than or equal to **VH(O)**.

Obviously, we expect a generic reconstructed object to supply less information than the same object considered as a visual hull. We have seen in the previous section that all the edges of a cubic visual hull are hard. Let us assume a definition of hard and soft points for a generic reconstructed **R** equal to that given for the visual hull. We will find that no hard point exists on the surface of the reconstructed cube of Fig. 7. In fact, consider the edges that the three objects in Fig. 8(b) share with the top face of the reconstructed cube (Fig. 9). It is easy to verify that no point on this face is shared by the edges of the three objects. Since hard points must be shared by all originating objects, no hard points exist on the top face of the cube. It is easily seen that the same statement holds for the other faces. In conclusion,

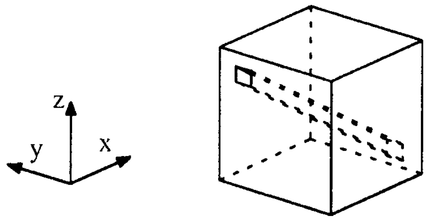

Fig. 10. An object yielding the same square silhouettes S_x, S_y, and S_z of Fig. 9.

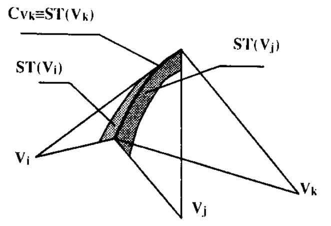

Fig. 12. How a hard line C_{V_k} can be generated.

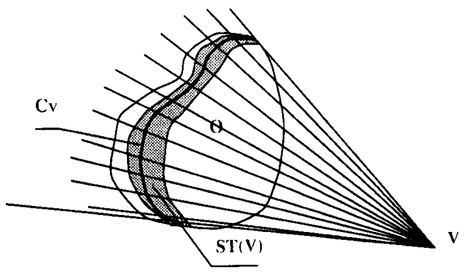

Fig. 11. Algorithm VI reduces the surface of the circumscribed cone relative to V to the surface strip ST(V), which contains the curve C_V belonging to **O**.

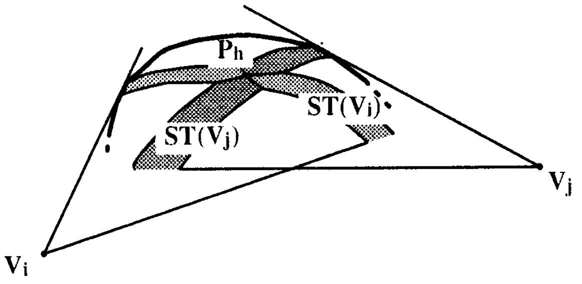

Fig. 13. How to generate a hard point P_h on a smooth surface.

in this case we are unable to find any point of the surface of the unknown object.

From the example, we can derive another important difference from the ideal case discussed in the previous section. It is easy to see that the *opaqueness property* does not hold for objects originating a generic **R**. In fact, let us consider the object shown in Fig. 10. It does not have the opaqueness property, but it produces the square silhouettes S_x, S_y, and S_z as the objects of Fig. 8, since the rectangular hole passing through the object is sufficiently small and skew with respect to the viewing direction parallel to the x axis.

Let us now address the following problems: In general, can there be hard points on the surface of a reconstructed object? And, if this were the case, how could we find them? To answer these questions, let us consider the reconstruction of generic object **O** (Fig. 11). The circumscribed cone starting at the viewpoint V is tangent to **O** along a curve C_V, which therefore is shared by **O** and the surface of the cone. The curve C_V shown in the figure is continuous, but in general it may have discontinuities. In addition, for exceptional alignment conditions, a half-line of the circumscribed cone may share multiple points and even segments with **O**. For instance, this happens when the viewpoint lies in a plane supporting a planar face of **O**. However, for simplicity, we will refer to the points shared by **O** and the circumscribed cone relative to the viewpoint V as the curve C_V.

While the VI algorithm is in progress, parts of the conical volume with vertex V are cut away as well as parts of its surface. However, it is clear that in this process no point of the curve C_V can lie on the discarded surface, since no point belonging to **O** can be removed by VI. In conclusion, the whole curve C_V must persist on the surface of the reconstructed object **R** after any number of intersections. It will lie on a *strip* of variable width which is what is left of the original circumscribed cone relative to V. Let us indicate this strip as ST(V) (Fig. 11). The following main theorem holds:

Proposition 9. Let P be a point of the surface of an object **R** reconstructed with the VI technique. A necessary and sufficient condition for P to be hard is that the boundaries of one[2] strip ST(V) to which it belongs be coincident at P.

In other words, in order to produce hard points the strip must have zero width (see [17] for a formal proof). This theorem gives us a full understanding of the relation between the reconstructed object **R** and the unknown object **O**. To summarize the previous discussion: The surface of any reconstructed object consists of a number of strips $ST(V_i)$, one for each viewpoint V_i. On each strip $ST(V_i)$ there is a curve C_V consisting of points of the unknown object **O**, but, unfortunately, if the strip has a certain width, we are not able to locate C_V on the strip and therefore all the points of the strip are soft. The only case in which we are able to locate hard points of the surface of **R** is when the course of the curve C_V is constrained by a strip of zero width.

An important consequence is that, if we consider objects **R** reconstructed with a finite number of intersections, and thus with a finite number of strips, it is not possible to find hard surfaces of **R**, but only hard points or, at most, hard lines. Let us recall that the visual hull is also able to supply hard surfaces: this is not surprising since these are curved surfaces which require an infinite number of zero-width strips for their full description.

Let us investigate now which kind of surface of **O** can produce hard lines. A hard line results in situations like that shown in Fig. 12, where the intersection C_{V_k} of the circumscribed cones relative to V_i and V_j also lies on the circumscribed cone relative to V_k, and therefore is coincident with the strip $ST(V_k)$. From the figure it is easy to see that hard lines must be *creases* of the surface, that is lines where there is a discontinuity of the surface's normal. It also follows that on a smooth surface it is possible to find only hard points and not hard lines. These hard points can be generated by the intersection of two strips, as shown in Fig. 13.

Unlike Proposition 4 of Section III, Proposition 9 is constructive, and can be directly used for computing hard points and lines. Actually, they can simply be obtained as side-products of the VI algorithm. The rest of this section is devoted to some example of computation of hard points and lines.

[2] As we will see later, more than one strip can share a hard point.

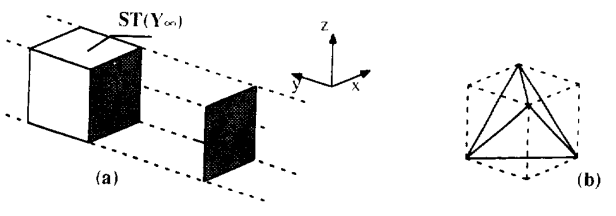

Fig. 14. The strip relative to the point at infinity of the axis y(a), and a tetrahedron where L_X, L_Y, and L_Z overlap (b).

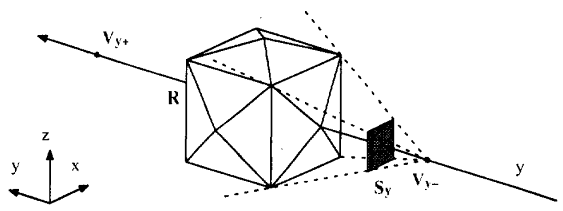

Fig. 15. **R** is generated by six equal square silhouettes obtained from viewpoints lying on the coordinate axes at the same distance from the origin.

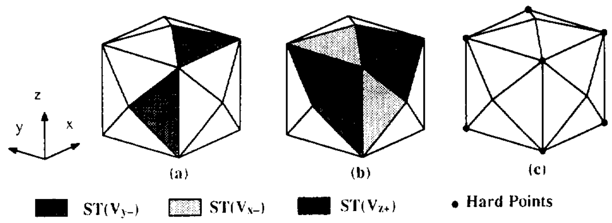

Fig. 16. The strip relative to V_{y-} (a); the strips relative to V_{x-} and V_{z+} (b); the hard points (c).

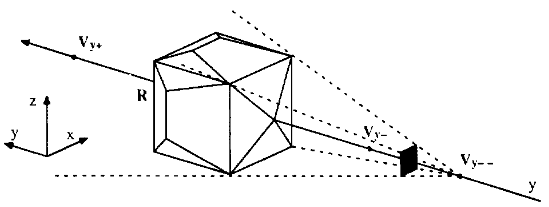

Fig. 17. **R** modified according to the silhouette obtained from V_{y--}.

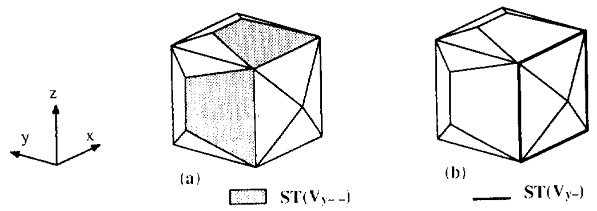

Fig. 18. The strip relative to V_{y--} (a) and that relative to V_{y-} (b), which has zero width and consists therefore of hard points.

Consider again the cube of Fig. 7, obtained from three square silhouettes. Inspecting some possible generating objects has shown that the surface of the cube has no hard points. It is interesting to obtain the same result by using Proposition 9. Let us consider the strip ST(Y∞), originated by the projection along the y axis (see Fig. 14(a)). The strip has the same non-zero width everywhere, and therefore all the points on it are soft. The same conclusion holds for the two strips originated by the viewing directions along x and z. Observe that at each face of the cube *two* strips overlap, and therefore on each face there are *two* curves C_Y. This explains why the objects shown in Fig. 9 share with each square face of the cube at least two arcs joining the two pairs of opposite edges of the face. Actually, only one arc, if it joins two opposite vertices, is sufficient, as in the case of the tetrahedron shown in Fig. 14(b). In this case, the curves C_X, C_Y, and C_Z relative to the three viewing directions overlap at the edges of the tetrahedron.

Let us now consider the volume **R** in Fig. 15. It has been constructed by back-projecting six equal square silhouettes from six viewpoints V_{x-},V_{x+}, V_{y-},V_{y+}, V_{z-}, and V_{z+}, lying on the coordinate axes at the same distance from the origin. For simplicity, in Fig. 15 only the silhouette relative to one of these points, namely V_{y-}, is shown. In Fig. 16(a) the visible part of the strip ST(V_{y-}) is highlighted. In this case the boundaries of the strip overlap at four points, which are therefore hard points. Considering the strips relative to the other viewpoints, it is easy to find that there are eight hard points in all (Fig. 16(c)). Observe that in this case three strips overlap at each hard point. In Fig. 16(b) the three strips relative to the viewpoints V_{y-}, V_{x-}, and V_{z+} are shown.

Let us add another viewpoint V_{y--} on the y axis at a distance from the origin greater than that of V_{y-}. The silhouette relative to this point is subject to some constraints. It must contain the projections of the eight hard points. We also have an outer bound for the silhouette, given by the outline of the object already obtained. Let us assume a silhouette equal to the largest square formed by the projections of the hard points, which satisfies the constraints. Intersecting the new cone and the previous volume we obtain the object shown in Fig. 17.

In Fig. 18(a) we show the new strip ST(V_{y--}), which reduces the bounding volume but does not supply any new information on the hard points

On the contrary, the whole strip ST(V_{y-}) relative to the old viewpoint on the y axis has been reduced to zero width by the new cone, so that we find the four hard segments shown in Fig. 18(b).

V. Summary

We have addressed the problem of inferring the shape of a 3D object **O** from its 2D silhouettes, or, which is equivalent, from the bounding volume **R** reconstructed using the volume intersection technique. We have divided the points lying on the surface of **R** into *hard* points, which also belong to the surface of any possible **O**, and *soft* points, which are only an outer bound for **O**. First, we have considered the limit case of optimal reconstruction, when the closest possible approximation of the object, its visual hull, has been obtained. We have given a necessary and sufficient condition for a point to be hard, and we have shown how to compute the hard points for some categories of objects.

Since in practice we could be unable to obtain the visual hull of an object or to find out if the reconstructed object is the visual hull or not, we have also addressed the problem of finding the hard points for a generic reconstructed object. Also, in this case, we have given a necessary and sufficient condition for a point to be hard. This condition is a constructive one, and is directly suited for the computation of the hard points.

It has also been shown that when, in theory, a reconstructed visual hull can also supply hard surfaces, a generic reconstructed object is able to supply only hard points or hard lines.

Future work will be aimed at using the results presented for constructing an active, object-specific volume intersection algorithm. Some heuristic work along this line has been done by Lavakusha et al.[13] and Shanmukh and Pujari [7]. An active volume intersection algorithm requires: 1) a measure of the current reconstruction accuracy for an unknown object; and 2) efficient rules for choosing the next viewpoint. The theoretical results of this paper suggest to define some kind of *surface* reconstruction accuracy, based on the hard points and lines of the current reconstructed object. It also seems possible to construct practical efficient rules for improving this accuracy, starting from the geometry of the current reconstructed object and its hard points and lines.

APPENDIX

In this Appendix we describe an algorithm for computing the hard convex edges (or parts of edges) of a polyhedral visual hull with *n* edges. Let us consider a point P of a convex edge E_i of a polyhedral visual hull **VH**. If we assume that P is hard, according to Proposition 4 there are lines passing through P and sharing with **VH** no other points. These lines fill a solid cone **C**, whose boundary consists of a number of planar surfaces, each formed by lines which pass through P and an edge E_j of **VH** and which do not share with **VH** any other point. Let us refer to these planar surfaces as PE surfaces. If we assume on the contrary that the point P is soft, the cone **C** is *flat*, i.e., it has no interior points, only boundary surfaces. The boundary of the cone **C** cannot completely disappear, since passing through any point P of the boundary of any visual hull **VH** (even though not polyhedral) there must be at least a line sharing with **VH** only points of its boundary. Otherwise, the point P would be a *concave* point of the boundary of **VH**, which, of course, is not possible.

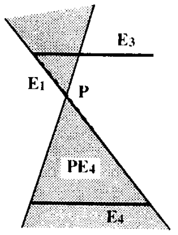

Fig. 19. A PE surface containing isolated edges.

It follows that, to verify whether a point P of a convex edge is hard, we can perform the following steps:

1) Find all the PE_j surfaces relative to P and to any edge E_j $(i \neq j)$
2) For each surface, compute the polygonal intersection SP_j of the surface and **VH**. SP_j could consist also of isolated edges. This happens, for instance, if we consider a point lying on the edge E_1 of the object in Fig. 5(b), and the sur face PE_4 (see Fig. 19).
3) For each surface, verify whether there are lines passing through P which do not intersect SP_j at any other point. It is easy to see that this can be done in $O(n^2)$ time. If such lines exist, P is hard, since we can rotate L, one of these lines, about P at an infinitesimal angle in a plane normal to PE_j (see Fig. 20) so that it becomes a line L' which satisfies the condition of Proposition 4.

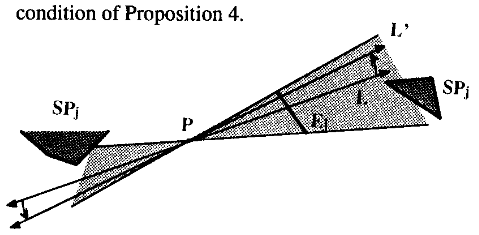

Fig. 20. A hard point P, through which a line L' passes satisfying the condition of Proposition 4.

Adding up, we can verify whether P is hard in $O(n^3)$ time. To apply the above algorithm to a random point of an edge is not sufficient if exceptional alignment conditions are allowed, since in this case an edge can be partially hard. For dealing with these cases, we observe that, if we imagine displacing P along E_i, the PE surfaces of the boundary of **C** can appear or disappear only when P crosses VE or EEE surfaces (see also [15]). The VE surfaces are those produced by the lines passing through a vertex V and an edge E of **VH**; the EEE surfaces are those produced by the lines passing through three edges of **VH**. These surfaces are $O(n_3)$, and divide each edge into $O(n_3)$ parts which must be checked separately. In conclusion, the algorithm in the general case takes $O(n_6)$ time. We shall not mention here some trivial pruning actions which do not affect the time bound of the algorithm.

REFERENCES

[1] Y.C. Kim and J.K. Aggarwal, "Rectangular parallelepiped coding for solid modeling," *Int'l J. Robotics Automation*, vol. 1, pp. 77-85, 1986.
[2] C. Chien and J.K. Aggarwal, "Model reconstruction and shape recognition from occluding contours," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 11, pp. 372-389, 1989.
[3] Lavakusa, A.K. Pujazi, and P.G. Reddy, "Linear octrees by volume intersection," *Computer Vision Graphics Image Process*, vol. 45, pp. 371-379, 1989.
[4] S. Srivastava and N. Ahuia, "An algorithm for generatng octrees from object silhouettes in perspective views," *Proc. IEEE Workshop Computer Vision*, Miami Beach, Fla., pp. 363-365, Nov. 1987.
[5] N. Ahuja and J. Veenstra, "Generating octrees from object silhouettes in orthographic views," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 11, pp. 137-149, 1989.
[6] T.H. Hong and M. Schneier, "Describing a robot's workspace using a sequence of views from a moving camera," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 7, pp. 721-726, 1985.
[7] K. Shanmukh and A.K. Pujari, "Volume intersection with optimal set of directions," *Pattern Recognition Letters*, vol. 12, pp. 165-170, 1991.
[8] W.N. Martin and J.K. Aggarwal, "Volumetric description of objects from multiple views," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 5, pp. 150-158, 1983.
[9] P. Srinivasan, S. Fukusa, and S. Azimoto., "Computational geometric methods in volumetric intersection for 3D reconstruction," *Pattern Recognition*, vol. 23, pp. 843-857, 1990.
[10] M. Potemesil, "Generating octree models of 3D objects from their silhouettes in a sequence of images," *Computer Vision Graphics and Image Process.*, vol. 40, pp. 1-29, 1987.
[11] H. Noborio, P. Liang, and S. Hackwood, "Construction of the octree approximating three-dimensional objects by using multiple views," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 10, pp. 769-782, 1988.
[12] C.H. Chien and J.K. Aggarwal, "Volume/surface octrees for the representation of three-dimensional objects," *Computer Vision, Graphics and Image Processing*, vol. 36, pp. 100-113, 1986.

[13] Lavakusha, A.K. Pujazi, and P.G. Reddy., "Volume intersection algorithms with changing directions of views," *Proc. MIV-89*, Tokyo, pp. 309-314, 1989.
[14] A. Laurentini, "The visual hull of solids of revolution," *Proc. 11th IAPR Int'l Conf. on Pattern Recognition*, The Hague, the Netherlands, pp. 720-724, 1992.
[15] A. Laurentini, "The visual hull concept for silhouette-based image understanding," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 16, pp. 150-162, 1994.
[16] E. Preparata and M. Shamos, *Computational Geometry*, New York: Springer-Verlag, 1985.
[17] A. Laurentini, "Theoretical capabilities and limits of the volume intersection technique," Int. Rept. L292, 1992.

Pose Estimation by Fusing Noisy Data of Different Dimensions

Yacov Hel-Or and Michael Werman

Abstract — **A method for fusing and integrating different 2D and 3D measurements for pose estimation is proposed. The 2D measured data is viewed as 3D data with infinite uncertainty in particular directions. The method is implemented using Kalman filtering. It is robust and easily parallelizable.**

Index Terms—**Sensor fusion, Kalman filter, pose estimation, model based, object recognition.**

I. Introduction

In model-based pose determination the position of a known object is determined from different types of surface measurements (for reviews see [1], [2], [3]). Usually, feature points such as maximum curvature, segment endpoints, and corners are measured. The aim of this correspondence is to determine the correct rigid transformation (translation and orientation) of the model points to the measured points where the measured data is not exact. This problem is known as *absolute orientation* in photogrammetry (for a review see [4]) and is classified into two major categories according to the type of measurements:

1) 3D to 3D correspondence: Both model and measurements supply information about the 3D location of features (measurements from range data, stereo, etc.).
2) 2D to 3D correspondence: The model is 3D while the available measurements supply projected 2D information. The projection can be perspective or orthographic.

Methods to compute the absolute orientation, most of which exploit least-square techniques in either closed (e.g. [5], [6], [7], [8], [9]) or iterative form (e.g. [2], [4], [6], [10], [11]), have been presented. However, each method can be applied to only one of the categories described above.

In this correspondence we suggest a uniform framework to compute the absolute orientation, where the measured data can be a mixture of 2D and 3D information. Unifying the different types of measurements is done by associating an uncertainty matrix with each measured feature. Uncertainty depends both on the measurement noise and the type of measurement. A 2D measurement is a projection (perspective or orthographic) onto a 2D plane, and we regard it as a measurement in 3D with infinite uncertainty in the direction of the projection. Therefore, the dimensionality of the measurements is encoded in the covariance matrix. This representation unifies the two categories of the absolute orientation problem into a single problem that varies only in the uncertainty values associated with the measurements. With this paradigm we obtain a uniform mathematical formulation of the problem and can fuse different kinds of measurements to obtain a better solution. The algorithm we describe has additional advantages of supplying a certainty measure of the estimate, enabling an efficient matching strategy and allowing simple parallelization.

Manuscript received Dec. 3, 1992; revised June 20, 1994.

Y. Hel-Or and M. Werman are with the Institute of Computer Science, Hebrew University of Jerusalem, 91904 Jerusalem, Israel; e-mail werman@cs.huji.ac.il.

IEEECS Log Number P95021.

II. General Overview of the Problem.

A *model M* of a 3D object is represented by a set of points:

$$M = \{u_i\},$$

where $\mathbf{u}_i$ is a three-dimensional object-centered vector associated with the i^{th} point.

A *measurement* of a 3D object is represented by M' which, similar to the model representation, is a collection:

$$M' = \left\{\left(\hat{\mathbf{u}}'_j, \Lambda_j\right)\right\} . \ \hat{\mathbf{u}}'_j$$

is a noise-contaminated measure of the real location-vector u'_i associated with the j^{th} measured point and is represented in a viewer-centered frame of reference.

Λ_j is the covariance matrix depicting the uncertainty in the sensed vector

$$\hat{\mathbf{u}}'_j.$$

We do not constrain the dimensionality of the measured data but allow it to be 3D (stereo, range finder, etc.), or 2D (orthographic or perspective projection).

A *matching* between the model M and the measurement M' is a collection of pairs of the form

$$matching = \left\{\mathbf{u}_k, (\hat{\mathbf{u}}'_k, \Lambda_k)\right\},$$

which represents the correspondence between the model points to the measured points. For simplicity we denote a model point and its matched measurement with the same indices.

A. The Problem.

Given a model M, a measurement M', and a matching as above, estimate a transformation T which optimally maps the points $\boldsymbol{u}_i$ of the model onto the corresponding measured points

$$(\hat{\mathbf{u}}'_i, \Lambda_i).$$

The estimated transformation T describes the position of the measured object M' in the 3D scene.

The method described below fuses the information from all the measured points and estimates the transformation T by incremental refinement using Kalman-filter tools. At each step a matched pair is introduced and an updated solution is produced.