Inferring CSG-Based Object Representation From Range Image†

Wei-Chung Lin and Tsu-Wang Chen
Dept. of Electrical Engineering and Computer Science
Northwestern University
Evanston, IL 60208

Abstract

3-D object representation is a challenging problem in the areas of computer vision, computer graphics, and CAD/CAM. In this paper, we propose a system to infer CSG-based representation scheme - precedence graphs - from range images. The system starts with segmenting a range image into primitive surface patches with simplest geometric characteristics. Then, the primitive patches are classified as surfaces of primitive volumes such as ellipsoids, blocks, cylinders, and cones. Finally, the order and types of set operations on these primitives are determined. All these are then used to construct a precedence graph. The advantages and disadvantages of the proposed approach are also discussed.

1. Introduction

The problem of representing 3-D objects in computers has been attracting the attention of researchers in the areas of computer vision, computer graphics, and computer-aided-design/computer-aided-manufacturing (CAD/CAM) [1-5]. Three general classes of 3-D solid object representation schemes which are often used include surface or boundary [6,7], volumetric [9,10], and sweep [4,8,9] methods. One of the volumetric methods - Constructive Solid Geometry (CSG) - is most effective in representing unsculptured, man-made objects such as industrial parts [11]. CSG represents an object as a binary tree where each leaf represents an instance of a primitive and each internal node represents an operation on its descendent(s). Primitives such as blocks, spheres, cylinders, cones, and tori are first transformed (translation, rotation, and scaling) and then combined by using the boolean set operators (union, intersection, and difference) from the bottom to the top of the tree. The internal nodes specify the types of boolean operations on its immediate children. The final shape is constructed at the root node of the tree.

Although CSG has been extensively used in CAD/CAM areas, to our knowledge there is no vision system built based on this representation scheme. The criticisms on using CSG representation scheme in robot vision applications mainly concern its accessibility [2,12]. More specifically, the criticisms center around the issues that it is computationally expensive to evaluate the surfaces and it is difficult to reconstruct the CSG tree from a given intensity image. However, most of the problems (except self-occlusion) can be solved if range images rather than intensity images are used. Since CSG method is potentially useful in integrating CAD/CAM and computer vision systems, we propose the technique to infer CSG-based representations - precedence graph - from range images in this paper. The advantage of inferring this representation scheme rather than a CSG tree is that it is unique in representing a 3-D object. Moreover, a precedence graph can be easily transformed into a CSG tree.

The proposed approach to inferring precedence graph consists of three major steps. The block diagram of the system is shown in Fig. 1. The system starts with segmenting a range image into primitive surface patches with the simplest geometric characteristics. Then, the primitive patches are classified as surfaces of primitive volumes such as spheres, cylinders, blocks, or cones. In addition, the locations and orientations of these primitives are estimated. The various techniques to recognize the primitive surface patches are discussed in Section 3. Finally, the order and types of set operations on these primitives are determined. This is achieved by constructing a precedence graph describing the order of operations. Section 5 shows some experimental results. The advantages and disadvantages of the proposed approach are discussed in the final section.

2. Range Image Segmentation

The range images used in this paper are obtained from a structured-light-based range sensor [13]. The goal of segmentation in our research is to partition a range image into regions corresponding to planar or quadratic surfaces. In this project, we adopt the strategy proposed in [14-16] for this purpose.

3. Primitive Recognition

The objective of this step is to recognize the primitives in the scene which are the building blocks of the object of interest. First, the surface equation for each region is computed. From the surface equations, we then derive the types and sizes of the primitives. However, the altitudes of

†This project is supported by the National Science Foundation under Grant ECS-8809147.

cylinders and cones are not derivable from the surface equations. They must be estimated from the image directly. Finally, the sizes of blocks are measured.

Before we start describing the process of primitive recognition, we'd like to point out the assumptions made on the scope of objects and operators in this research. Then, the major steps in this process will be elaborated.

3.1. Assumptions

3.1.1. Domain of Objects

In this paper, we assume that only primitive objects with quadratic surfaces (e.g. ellipsoids, cylinders and cones) and blocks are allowed to compose an object. One of the difficulties in primitive recognition is caused by the failure in segmenting two different surface patches whose boundaries consist of second or higher order discontinuities. To our knowledge, there is no reliable approach to detecting such edges to date. Another class of objects - polyhedra - which can be constructed by boolean operations on the block primitives will not be considered either since their descriptions using CSG scheme are awkward and difficult to derive from the scene. They can be better represented by piecing together the bounding planar surface patches rather than composing blocks [17].

3.1.2. Operators for Composing Objects

In CSG modeling scheme, any intersection operation can be replaced by two consecutive difference operations, i.e. $A \cap B = A-(A-B)$. Thus, we can use only union and difference operators to build a CSG tree of an object without loss of generality. In most cases, when a primitive object is subtracted from another one, a concave surface or a plane is created on the subtracted one. In this paper, the subtrahend object is called a *positive* object, and the subtracter, *negative* object. For example, in Fig. 2, object i is *positive* while object j is *negative*.

Another assumption made is that in modeling an object a primitive can be a subtracter at most once. All the primitives (positive or negative) used in our research have convex or planar surfaces. If a primitive plays the role of subtracter twice, it's resultant surface patch on the object becomes convex. Thus, it is difficult to decide if the primitive of a convex surface patch has never been a subtracter or has been a subtracter an even number of times. Therefore, under this restriction, a convex primitive surface patch can never be a subtracter while a concave one is the result of being a subtracter once. Thus, we can infer whether a primitive is positive or negative simply from the convexity or concavity of the visible surface in the image.

3.2. Determine Surface Equations

During the segmentation step, whether a surface patch is planar or curved is determined. Therefore, the surface equations can be derived as well. Since the equations are derived based on certain reference coordinate system, we will describe this system before we explain how the equations are obtained. The coordinate system for the range image is defined as follows [13]:

1) x-axis: the direction parallel to the projected laser stripes. In the image, it is the direction along the rows.
2) y-axis: the direction along which the reference plane moves. In the image, it is the direction along the columns.
3) z-axis: the direction perpendicular to both x- and y-axes. The directions of x-, y-, and z- axes form a right-handed coordinate system.
4) origin: the lower left-handed corner of the image.

The units of the image coordinate system are in inches. Hereafter, all the computations involving the coordinates of image points are based on this coordinate system.

The planar surface equations are represented in the form of $z = ax + by + c$. In general, a quadratic surface can be represented by the equation

$$ax^2+by^2+cz^2+dxy+eyz+fzx+gx+hy+iz = 1$$

where a, b, c, d, e, f, g, h, and i are coefficients to be determined. In this step, all the points on the surface patch except those seriously corrupted by noise are used. These noisy points are detected during surface fitting in the segmentation process. From the surface equation, the location, orientation, and type of the primitive object can be determined.

3.3. Determine Convexity or Concavity of Curved Surface Patches

From the definitions given in Sec. 3.1.2, we know that the primitive with convex surface is positive while the primitive with concave surface is negative. For the quadratic surfaces used in our research, if the surface equation is formulated such that z is a function of x and y, there will be two possible surface equations: one is concave and the other is convex. This is because there are two solutions for z for some specific x and y values if the quadratic equation in z is solvable. In this step, the convexity and concavity of curved surface patches are determined. It is shown in Fig. 3.

3.4. Determine the Altitudes of Cylinders and Cones

In this step, the altitudes of cylinders and cones are estimated by projecting certain vectors on the principal axes. For every boundary point of the cylinder, make a vector whose tail is the reference point and compute the inner product of this vector and the unit vector of the principal axis. The difference between the maximum and the minimum of projections is used as the estimate of the altitude. The plane equations of bases may be calculated with the direction vector of the principal axis and the points where the maximum and minimum projections occur. In a similar mannar, the altitudes and the base equations of cones can be found except the reference points are the vertices and the maximum magnitudes of

the projections are the estimated altitudes. The ideas are illustrated in Figs. 4, 5.

3.5. Determine Blocks

Up to now, all the parameters of primitive objects except blocks are determined and there are only planar surface patches left in the image. The next step is to check, for each planar surface patch, if it is a lower base or upper base of a neighboring cylinder or a base of a neighboring cone. This can be achieved by comparing its plane equation with those of the upper and lower bases of a cylinder or the base of a cone. If they are identical, the plane may be the upper base or the lower base of the cylinder or the base of the cone. Next, we choose a point on the patch. If the point is inside the curved surface of that cylinder or cone, the plane is a base. Otherwise it is a face of a block.

The planar surface patches which are parts of cylinders or cones are ignored. This is due to the fact that they only provide redundant information. Now, the only surface patches left are faces of blocks which are not associated with bases of cylinders or cones. In this paper, we assume that more than two sides are present in the solid model of a block. For example, the object of Fig. 6 is constructed by subtracting a block from a cone. It is impossible to determine the parameters of the block by using just one planar surface patch.

Under the general position assumption, at least two sides of each block are visible in the scene. For a planar surface patch, its neighboring planar surface patches belonging to the same block is searched by examining whether their normals are perpendicular or not.

Next, we want to determine whether the block is positive or negative. In Fig. 7(a), patches A and B are two faces of a block. Compute the dot product of the normal of plane A, $\hat{n}_A$, and a vector whose head lies on plane B and tail on plane A. If the result is positive, it means that the block is a negative object as illustrated in Fig. 7(b). Otherwise, it is a positive object as illustrated in Fig. 7(c).

The orientation of the block can be determined by $\hat{n}_A$, $\hat{n}_B$ and $\hat{n}_A \times \hat{n}_B$. The length of the block along $\hat{n}_A$ direction is determined by the method similar to that of determining the altitude of a cylinder described in Section 3.4 as is illustrated in Fig. 7(d). The derived block is shown in Fig. 8.

4. Precedence Graph Derivation

Up to this stage, all the parameters of primitive objects are determined. In other words, all the information of the leaf nodes in CSG tree is known. Positive primitives are the results of addition while negative primitives are the results of subtraction. What left to be determined is the order of introducing these primitives in the construction process. In this section, an inference procedure to construct a CSG-based representation scheme is described. In this research, we propose a CSG-based representation scheme - precedence graph - which can be converted to a CSG tree if needed. It is more specific than a CSG tree and can also be used to regenerate the shape of the inferred object for visualization.

4.1. Definition of Precedence Graph

A precedence graph, denoted by $PG = (A, R)$, consists of a finite set of nodes A and a set of binary relation R. It is an acyclic directed graph whose nodes correspond to individual primitives and whose structure represents the order of participation of the primitives in forming the graph. The nodes are equivalent to the leaf nodes of CSG trees and are classified as positive or negative. They are labeled from 1 to n if there are n nodes. An edge from node j to node i means that primitive j can be introduced to the construction process only after primitive i has been introduced. A precedence number is assigned to each node to indicate its order in the construction process.

For two adjacent positive primitives, there is no precedence relation between them. If there is no other restriction, they can be introduced in any order. The same reasoning applies to two adjacent negative primitives. However, there are two kinds of precedence relations between a positive primitive and its adjacent negative primitive. The first one occurs when a negative primitive is subtracted from a positive one as is shown in Fig. 2. The negative primitive must be introduced later than the positive one. Otherwise, without the existence of the positive primitive, there is nothing to subtract from. In this case, an arc from the negative node j to the positive one i is established and (i, j) is added to R to indicate the precedence relation. It will be referred to as *a subtracting arc* in the subsequent discussions. The second relation happens when a positive primitive j resides on the object which the negative primitive i subtracts as is illustrated in Fig. 9. The positive primitive must be introduced later than the negative one. Otherwise, the positive primitive j will be subtracted by the negative one. In this case, an arc is established from the positive node j to the negative node i and (i, j) is added to R. It will be referred to as *an appending arc* in the subsequent discussions.

For every primitive, its adjacent primitives can be found from the segmented image. The relation between a positive primitive and a negative one is determined by checking the points on the negative one to see whether they are inside the positive one or not. If they are inside, the relation is the first kind. Otherwise, it is the second kind. Since there is only one possible precedence relationship between two adjacent positive and negative primitives, there is only one way to establish the precedence graph. Therefore, the precedence graph is unique.

4.2. Construction of Precedence Graph

The precedence graph may be composed of several connected digraphs (or directed graphs). Because there is no arc between the individual connected digraph, there is no precedence relation between them. Thus, for each connected digraph, the corresponding object can be constructed independently. These objects are referred as *subobjects* . The whole object is just the union of these subobjects.

According the method described in the last subsection, the appending and subtracting arcs are established between nodes. The resulting graph has several connected digraphs. For acyclic digraphs, linear order can be found using topological sorting [20]. After the ordering, the nodes with odd order are positive, while those with even order are negative. The order indicates the precedence relationships in the construction process.

5. Experimental Result

The example object shown in Fig. 10 consists of two subobjects. Subobject one is obtained by subtracting a cylinder from a block and then unifying the result with two cylinders. Subobject two is a cylinder placed on the top of subobject one. For suboject one, the block is the order one primitive since it is introduced first. The large cylinder is the order two primitive since it subtracts the block. The order three primitives are the two small cylinders. From the scene, the only way to construct the object is to unify these two cylinders after the subtraction of the large cylinder. Subobject two is composed of only a cylinder. It has no precedence relationship with the primitives in subobject one. The whole object is the union of these two subobjects. The precedence graph indicates this information precisely. The inferred geometric information is satisfactory except the size of the negative cylinder. The radius of the cross section of the negative cylinder is 2.5". Therefore, the perimemter of the cross section is about 15.7". For a cylinder with this size, we may see half of the perimeter along the cross section, i.e. 7.85". In this object, only about 1.2" along the cross section of this cylinder is visible. The inferred equation shows that the cross section is an ellipse with major and minor axis lengths 0.52" and 0.07", respectively. The curvature of this cylindrical surface is relatively small compared with other primitives and only a small portion of it is visible to be used to compute the surface equation. This is a reasonable result because there is no sufficient evidence to infer the surface equation correctly. This problem must be taken into consideration in the stage of object recognition.

6. Concluding Remarks

We have proposed an automated system to infer object representations from range images consisting of objects constructed with the CSG method. The major advantage of using CSG scheme to represent 3-D objects in a computer vision system is that only a small number of primitive types are present and thus each primitive surface patch can only be categorized into one of a few classes. Another advantage is that a 3-D object rather than a $2\frac{1}{2}$-D object is reconstructed from the image. It is due to the fact that not only physically existing but also imaginary primitive volumes are inferred by the system. This makes the subsequent matching task much easier if the models prestored in the computer are also 3-D representations (which is true in most systems). The transformation to make the inferred objects and the models compatible (or in the same representation scheme) becomes unnecessary. Furthermore, if it is used in an automated manufacturing environment, the vision system and the CAD/CAM can share the same database without the need to convert from one representation scheme to another.

However, the proposed representation scheme has several limitations. The method is not applicable when the descriptions of the objects of interest include subtraction of negative objects or tori. This limits the scope of objects which can be dealt with the proposed method. Fortunately, only a small percentage of industrial parts fall into this category. Since the method is surface-oriented (though the resultant representation scheme is volumetric), every primitive volume must have its surfaces visible to a certain degree so that the surface parameters can be estimated correctly. However, this is an inherent limitation in any system using single-view approach. A single view is sometimes not adequate for determining the parameters of the primitives since some of them may be hidden or partially occluded. These should be handled by a proper inexact matching algorithm during recognition. Object recognition based on precedence graph matching is currently under investigation.

Reference

[1] P. J. Besl and R. C. Jain, "Three-Dimensional Object Recognition," *ACM Computing Surveys,* Vol. 17, No. 1, pp. 75-145, Mar. 1985.

[2] D. Marr and H. K. Nishihara, "Representation and Recognition of the Spatial Organization," *Proc. R. Soc. Lond. B. 200,* pp. 269-294, 1978.

[3] G. J. Agin and T. O. Binford, "Computer Description of Curved Objects," *IEEE trans. on Computers,* Vol. C-25, No. 4, April 1976.

[4] A. A. G. Requicha, "Representations for Rigid Solids : Theory, Method, and Systems," *ACM Computing Surveys,* Vol. 12, No. 4, Dec. 1980.

[5] S. N. Srihari, "Representation of Three-Dimension Digital Images," *ACM Computing Surveys,* Vol. 13, No. 4, pp. 399-424, Dec. 1981.

[6] R. E. Barnhill and R. F. Rosenfeld, "Surface Representation for Computer Aided Design," in *Data Structures, Computer Graphics and Pattern Recognition,* A. Klinger, K. S. Fu, and T. L. Kunii, Eds. New York: Academic, 1977.

[7] W. C. Lin and K. S. Fu, "A Syntactic Approach to 3-D Object Representation," *IEEE Trans. Pattern Anal. Machine Intell.,* Vol. PAMI-6, No. 3, pp. 351-364, May 1984.

[8] R. Nevatia, *Machine Perception,* Prentice-Hall, Inc., 1982, pp. 75-79.

[9] D. Hearn and M. P. Baker, *Computer Graphics,* Prentice-Hall, Inc., 1986, pp. 213-215.

[10] D. H. Ballard and C.M. Brown, *Computer Vision,* Prentice-Hall, Inc., 1982, Chap.9.

[11] S. D. Roth, "Ray Casting for Modeling Solids," *Computer Graphics and Image Processing,* Vol.18, pp. 109-144, 1982.

[12] B. Bhanu and C. C. Ho, "CAD-Based 3D Object Representation for Robot Vision," *IEEE Computer Magazine,* pp. 19-35, Aug. 1987.

[13] F. Lo, "An Effective Calibration Procedure For Structured-Light-Based Range Sensors", Master Thesis, Dept. Elec. Eng. Comput. Sci., Northwestern Univ., Evanston, Sept. 1988.

[14] P. J. Besl, "Surfaces in Early Range Image Understanding," Ph.D. dissertation, Dept. Elec. Eng. Comput. Sci., Univ. Michigan, Ann Arbor, Rep. RSD-TR-10-86, Mar. 1986.

[15] P. J. Besl and R. C. Jain, "Segmentation Through Variable-Order Surface Fitting," *IEEE Trans. Pattern Anal. Machine Intell.,* Vol. PAMI-10, No. 2, pp. 167-192, Mar. 1988.

[16] P. J. Besl, *Surfaces in Range Image Understanding,* Springer-Verlag, 1988.

[17] B. A. Boyter, "Three-Dimensional Matching Using Range Data," *The First Conference on Artificial Intelligence Applications,* Sheraton, Denver Tech center, Dec. 1984, pp. 211-216.

[18] W. H. Press, B. P. Flannery, S. A. Teukolsky, and W. T. Vetterling, *Numerical Recipes in C,* Cambridge University Press, 1988, pp. 60-72.

[19] S. H. Friedberg, A. J. Insel and L. E. Spence, *Linear Algebra,* Prentice-Hall, Inc., 1979, pp. 393-397, pp. 412-415.

[20] B. Kolman, R. C. Busby, *Discrete Mathematical Structures for Computer Science,* Prentice-Hall, Inc., 1984, pp. 133-134, pp. 172.

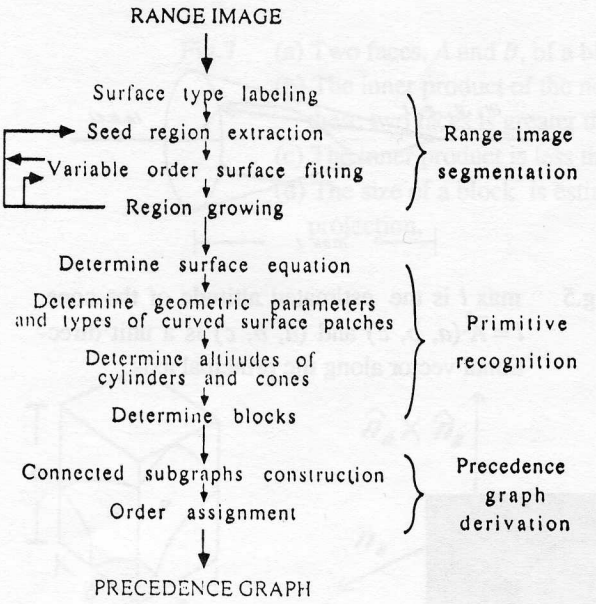


Fig.1 Block diagram of the proposed system.

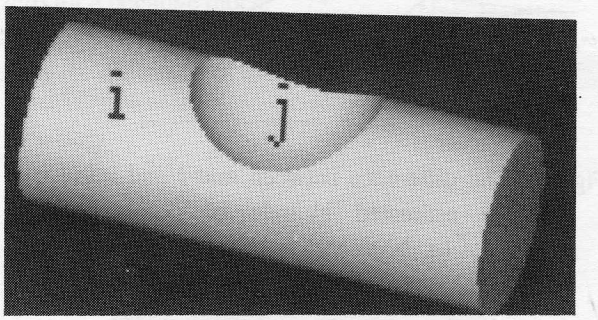
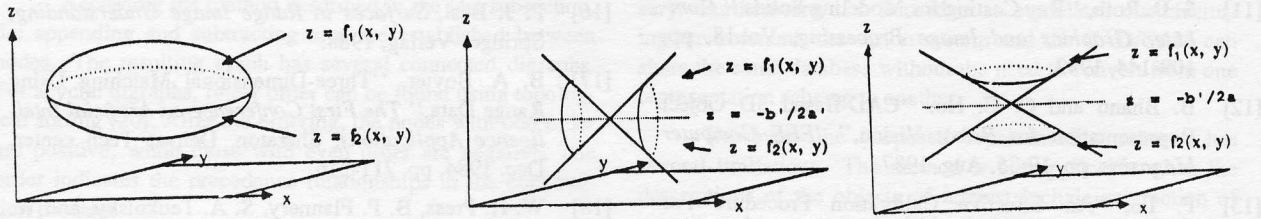


Fig.2 i (the cylindrical primitive) is a positive object while j (the derived or imaginary spherical primitive) is a negative object.



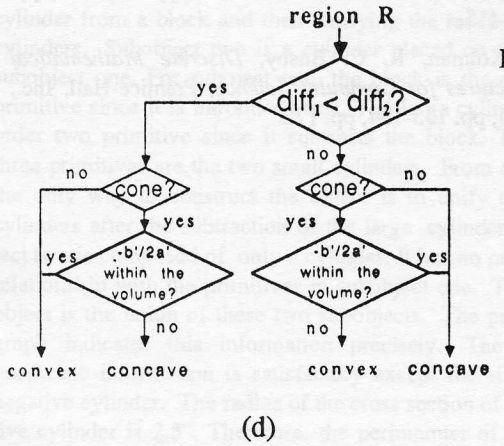


Fig.3 (a) For an ellipsoid or cylinder the surface with smaller z values is concave while the surface with larger z values is convex.
(b) The situation for cones in which the same method used in determining the convexity or concavity of ellipsoids and cylinders can be used.
(c) The situation for cones in which the decision concluded by using the method for ellipsoids and cylinders must be reversed.
(d) Flow chart of the decision process.

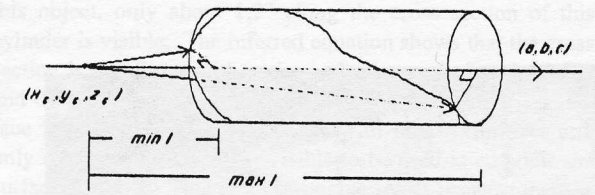


Fig.4 The difference between max l and min l is the estimated altitude of the cylinder. $l = \vec{A} \cdot (a, b, c)$ and (a, b, c) is a unit directional vector along the principal axis.

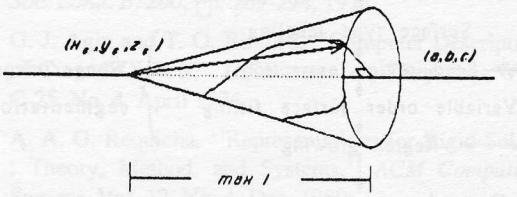


Fig.5 max l is the estimated altitude of the cone. $l = \vec{A} \cdot (a, b, c)$ and (a, b, c) is a unit directional vector along the principal axis.

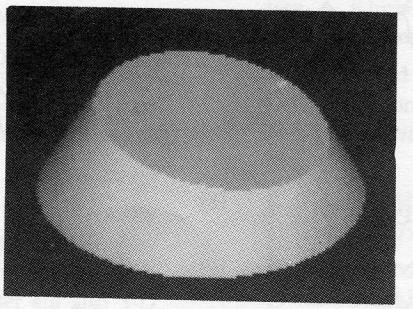
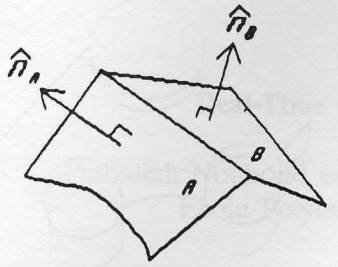
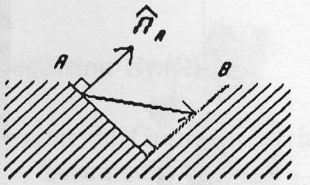


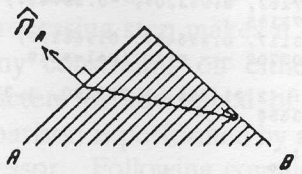
Fig.6 It is difficult to determine the parameters of a block with just one face present.



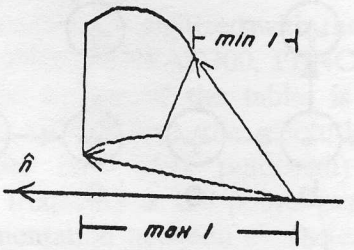
(a)



(b)

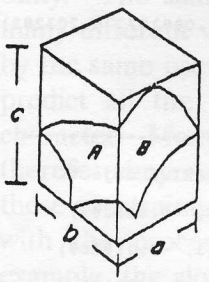


(c)

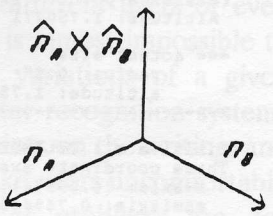


(d)

Fig.7 (a) Two faces, A and B, of a block with normal $\hat{n}_A$ and $\hat{n}_B$, respectively.
(b) The inner product of the normal of one face and a vector across these two faces is greater than zero for a negative block.
(c) The inner product is less than zero for a positive block.
(d) The size of a block is estimated by the difference of the projection.



(a)



(b)

Fig.8 (a) The derived block primitive.
(b) The orientation of the block.

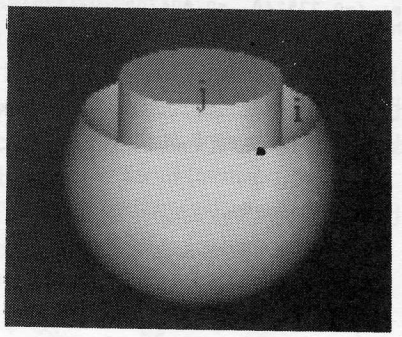
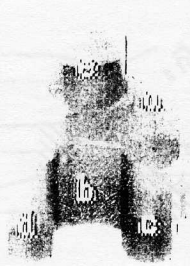
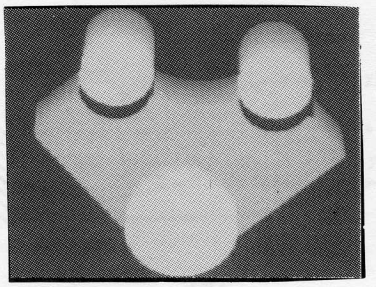


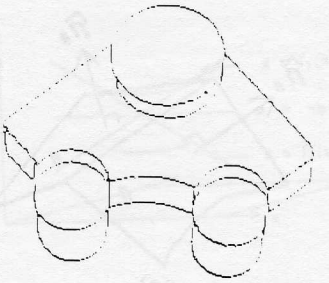
Fig.9 The positive primitive j resides on the object which the negative primitive i subtracts.



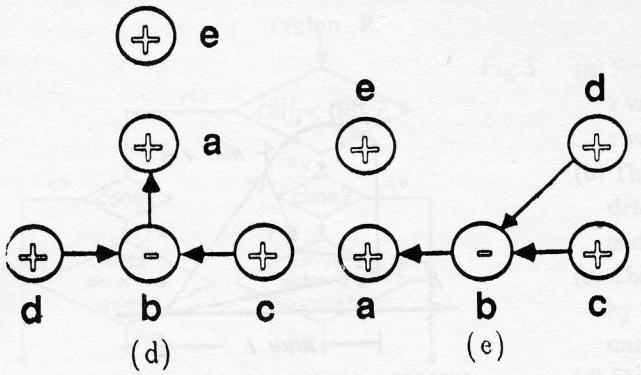
(a)



(b)



(c)




```
---------- Precedence Graph Construction ----------

There are 2 subobjects:

------------- Subobject 1 --------------
  Order 1 Primitive:
   Primitive a
  Order 2 Primitive:
   Primitive b
  Order 3 Primitives:
   Primitive c
   Primitive d

------------- Subobject 2 --------------
  Order 1 Primitive:
   Primitive e

------------ PRIMITIVES INFORMATION ------------

Primitive a : BLOCK --- POSITIVE object
Three coordinate axes:
  X-axis: (-0.630542, 0.755632, -0.177301)
  length: 3.195559
  Y-axis: (-0.008507, -0.274689, -0.961495)
  length: 1.468365
  Z-axis: (-0.775240, -0.604755, 0.179632)
  length: 3.165530

vertex: (4.402141, 2.361938, 1.996428)

--- actual size: 3.0" * 1.5" * 3.0"
-----------------------------------------
```

(f)

```
Primitive b : CYLINDER --- NEGATIVE object
Three coordinate axes:
  X-axis: (0.996750, -0.080513, -0.002481)
  semiaxis: 0.522945
  Y-axis: (0.077162, 0.963207, -0.257447)
  semiaxis: 0.072185
  Z-axis: (0.023117, 0.256419, 0.966289)
  semiaxis: 3.507709 --- principal-axis

reference point: (2.214396, 1.799580, 1.373004)
 Altitude: 1.403658

--- actual size:
      radius: 2.5"
      altitude: 1.5"
-----------------------------------------

Primitive c : CYLINDER --- POSITIVE object
Three coordinate axes:
  X-axis: (0.792896, 0.580253, -0.186072)
  semiaxis: 0.503597
  Y-axis: (-0.608730, 0.768104, -0.198659)
  semiaxis: 0.484890
  Z-axis: (0.027650, 0.270783, 0.962243)
  semiaxis: 7.036047 --- principal-axis

reference point: (3.192575, 0.984286, 0.435881)
 Altitude: 1.773550

--- actual size:
      radius: 0.5"
      altitude: 1.75"
-----------------------------------------

Primitive d : CYLINDER --- POSITIVE object
Three coordinate axes:
  X-axis: (0.991740, -0.120795, 0.043140)
  semiaxis: 0.473861
  Y-axis: (0.127980, 0.954387, -0.269752)
  semiaxis: 0.407281
  Z-axis: (-0.008588, 0.273045, 0.961963)
  semiaxis: 6.033414 --- principal-axis

reference point: (1.133507, 1.099187, 0.203283)
 Altitude: 1.750271

--- actual size:
      radius: 0.5"
      altitude: 1.75"
-----------------------------------------

Primitive e : CYLINDER --- POSITIVE object
Three coordinate axes:
  X-axis: (0.999741, 0.019136, -0.012287)
  semiaxis: 0.765608
  Y-axis: (-0.021730, 0.963143, -0.268109)
  semiaxis: 0.746136
  Z-axis: (0.006704, 0.268307, 0.963310)
  semiaxis: 4.938795 --- principal-axis
reference point: (2.508324, 8.351080, 18.979370)
 Altitude: 0.881827

--- actual size:
      radius: 0.75"
      altitude: 0.75"
-----------------------------------------
```

Fig.10 (a) Photograph of object B.
a: (positive) block
b: (negative) cylinder
c: (positive) cylinder
d: (positive) cylinder
e: (positive) cylinder
(b) Range image of object B.
(c) Image after segmentation
(d) Precedence graph.
(e) Sorted graph.
(f) Geometric information.