provided excellent programming support and many ideas. In addition, D. McKeown, S. Shafer, and D. Smith have provided useful comments and criticism.

REFERENCES

[1] A. Baer, C. Eastman, and M. Henrion, "Geometric modelling: A survey," *Computer-Aided Des.*, vol. 11, pp. 253–272, Sept. 1979.
[2] H. H. Baker and T. O. Binford, "Depth from edge and intensity based stereo," in *Proc. IJCAI '81*, 1981, pp. 631–636.
[3] S. T. Barnard and W. B. Thompson, "Disparity analysis of images," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. PAMI-2, pp. 333–340, July 1980.
[4] H. G. Barrow, R. C. Bolles, T. D. Garvey, J. H. Kremers, J. M. Tenenbaum, and H. C. Wolf, "Experiments in map-guided photo interpretation," in *Proc. IJCAI '77*, Aug. 1977, p. 696.
[5] R. N. Devich and F. M. Weinhaus, "Image perspective transformations," *Proc. SPIE*, vol. 238, pp. 322–332, July 1980.
[6] J. Dolye, "A truth maintenance system," *Artificial Intell.*, vol. 12, pp. 231–272, 1979.
[7] R. O. Duda and P. E. Hart, *Pattern Classification and Scene Analysis.* New York: Wiley, 1973.
[8] C. M. Eastman and K. Preiss, "A unified view of shape representation and geometric modeling," presented at the Israel Conf. on CAD, Jan. 1982.
[9] M. J. Hannah, "Computer matching of areas in stereo images," Stanford Univ., Tech. Rep. AIM-239, July 1974.
[10] R. L. Henderson, W. J. Miller and C. B. Grosch, "Automatic stereo reconstruction of man-made targets," *Proc. SPIE*, vol. 186, pp. 240–248, Aug. 1979.
[11] S. Liebes, "Geometric constraints for interpreting images of common structural elements: Orthogonal trihedral vertices," in *Proc. Image Understanding Workshop*, Apr. 1981, pp. 168–174.
[12] B. D. Lucas and T. Kanade, "An iterative image registration technique with an application to stereo vision," in *Proc. IJCAI '81*, Aug. 1981, pp. 674–679.
[13] D. Marr and T. Poggio, "A computational theory of human stereo vision," *Proc. Royal Soc. London B*, vol. 204, pp. 301–328, 1979.
[14] R. Nevatia and K. R. Babu, "Linear feature extraction and description," *Comput. Graphics Image Processing*, vol. 13, pp. 257–269, July 1980.
[15] S. Rubin, "Natural scene recognition using locus search," *Comput. Graphics Image Processing*, vol. 13, pp. 298–333, 1980.
[16] S. A. Shafer and T. Kanade, "Using shadows in finding surface orientations," Carnegie-Mellon Univ., Tech. Rep. CMU-CS-82-100, Jan. 1982.

Representation and Shape Matching of 3-D Objects

BIR BHANU

Abstract—A three-dimensional scene analysis system for the shape matching of real world 3-D objects is presented. Various issues related to representation and modeling of 3-D objects are addressed. A new method for the approximation of 3-D objects by a set of planar faces is discussed. The major advantage of this method is that it is applicable to a complete object and not restricted to single range view which was the limitation of the previous work in 3-D scene analysis. The method is a sequential region growing algorithm. It is not applied to range images, but rather to a set of 3-D points. The 3-D model of an object is obtained by combining the object points from a sequence of range data images corresponding to various views of the object, applying the necessary transformations and then approximating the surface by polygons. A stochastic labeling technique is used to do the shape matching of 3-D objects. The technique matches the faces of an unknown view against the faces of the model. It explicitly maximizes a criterion function based on the ambiguity and inconsistency of classification. It is hierarchical and uses results obtained at low levels to speed up and improve the accuracy of results at higher levels. The objective here is to match the individual views of the object taken from any vantage point. Details of the algorithm are presented and the results are shown on several unknown views of a complicated automobile casting. The results of partial shape recognition are used to determine the orientation of the object in 3-D space.

Index Terms—Face matching, hierarchical relaxation, optimization, planar approximation, range data analysis, region growing, stochastic labeling, surface representation, 3-D object modeling, 3-D scene analysis, 3-D shape matching.

Manuscript received July 12, 1982; revised April 20, 1983. This work was supported in part by DARPA under Contract F-33615-80-C-1080 and in part by a grant from the Institut National de Recherche en Informatique et en Automatique (INRIA), France.
The author is with the Aeronutronic Division, Ford Aerospace and Communications Corporation, Newport Beach, CA 92660.

I. INTRODUCTION

In the development of robots with vision capability, representation, and shape recognition of 3-D objects are of crucial importance. It is well known that the recognition of even simple objects is not easy, if the object is allowed to rotate and have arbitrary view in 3-D space. Recognition of real objects is required in the process of automatic selection, inspection, manipulation, and assembly of industrial parts, for example, parts going over a conveyor belt, picking the parts from a bin, automation of assembly line operations, etc. Motivated by such practical applications, in this paper we consider the representation, modeling and shape matching aspects of 3-D scene analysis. Our interest is to match individual views of a 3-D object (taken from any arbitrary viewing angle) against the 3-D model. A method based on a laser triangulation to acquire 3-D data will be described. The problems related with 3-D data acquisition and geometric processing will be addressed. A technique for representing a 3-D object by a set of planar convex faces will be presented. These faces are determined by sequentially choosing three very close noncollinear points and investigating the set of points lying in the plane of these points. Two simple tests, one for convexity and the other for narrowness ensure that the set of points is an object face. This set of points is approximated by polygons. The method is used to generate a 3-D model of an object by combining the object points from a sequence of range images. A hierarchical stochastic labeling technique is used for shape matching. The technique explicitly maximizes a criterion function based on the ambiguity and inconsistency of classification. We have used a similar technique to solve the "segment matching" problem in two dimensions [1], [2]. Here we extend this technique to solve the "face matching" problem, which is defined as the recognition of a partial 3-D shape as an approximate match to a part of a larger 3-D shape. The results of matching are used to determine the orientation of the object in three-space. Examples are presented using a complex automobile part.

II. THREE-DIMENSIONAL SCENE ANALYSIS SYSTEM AND DATA ACQUISITION

Fig. 1 shows the schematic diagram of the 3-D scene analysis system implemented in this work. First we acquire 3-D data using a laser ranging system shown in Fig. 2. The acquisition system is based on the principle of active stereoscopy. A laser emits a beam of ruby red light which is reflected by a mirror which rotates and sweeps the beam along the x-axis to produce one scan line. The beam is reflected from the object, and the

0162/8828/84/0500-0340$01.00 © 1984 IEEE

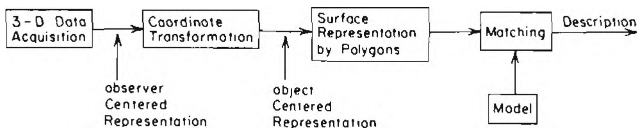

Fig. 1. The schematic diagram of 3-D scene analysis system.

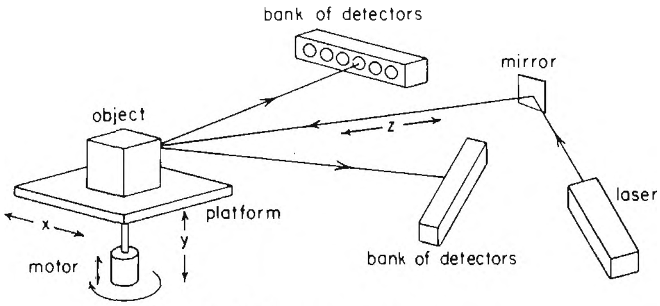

Fig. 2. Laser ranging system.

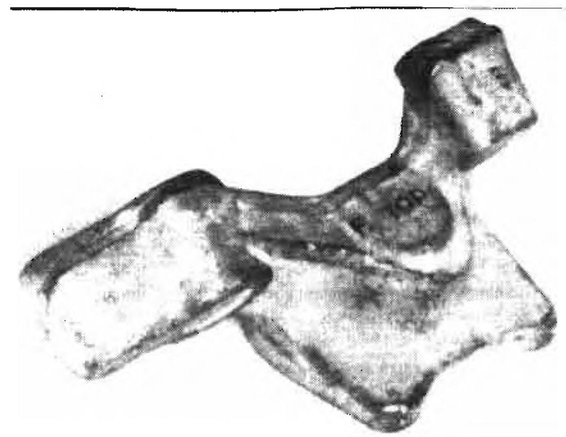

Fig. 3. Automobile piece analyzed.

z-distance is calculated from the location of the maximum response in each bank of detectors. The platform on which the object rests can be raised or lowered (this is the y-axis) and can also be rotated (around the y-axis). The sampling distances used here are 3.0 mm in the x-axis, 2.0 mm in the y-axis, and an accuracy of 0.5 mm in the z-distance is achieved. Objects of sizes up to 750 × 750 × 600 mm can be digitized using this system. Further details about the 3-D sensor can be found in [3].

The data so obtained are in the observer centered coordinate system (the one in which the observer or camera receives the image). While creating a 3-D model of the object, object centered representation (a system centered about the object which allows all points on the surface of the object to be referred with respect to this system) is required. This is computed by marking the zero position for x- and y-axis and obtaining a reference value for z-axis on the platform in Fig. 2. The actual position or orientation of the object on the platform does not matter when acquiring the data related to an unknown view of an object. As an example Fig. 3 shows a complicated casting of an automobile piece. Notice that this object does not contain any major horizontal or vertical surface. In order to create a 3-D model of the object, a range data image was produced for every 30° rotation of the object around the y-axis in the x-z plane. Finally, top and bottom views of the object were taken. These two views were put in correspondence with the other views by having three control points in each of these views which were also visible in the 0° view (requiring six control points in the 0° view) of the object and computing the transformations. The 14 views obtained using the range data acquisition system are shown as gray scale images in Fig. 4. In this figure the lighter points are farther away from the observer and the darker ones are closer. After thresholding the background points, each individual view shown in Fig. 4 had approximately 2000 points except for the 90° and 270° views which had about 900 points. The surface points for the complete object were obtained by following in sequence the views starting from the 30° view and ending with the top view and computing the distance between the transformed point and the points which are already in the list (in the beginning just the 0° view points). If the minimum difference is less than a certain threshold related to the sampling distance, we discard this point; otherwise the point is added to the list. For the sampling distances as mentioned above, using a distance threshold of 3.87 mm, the complete object has 8314 points which are stored as a list. From the set of 3-D points we obtain a higher level representation of surface and finally the unknown scene is matched against the model to obtain the description of the scene.

III. Representation and Modeling of 3-D Objects

Representation: A direct model of a 3-D object as a 3-D array can easily exhaust the memory capacity of a system (for example a 3-D array of size 128 will require 128^3 = 2 097 152 bits of memory). Moreover, this array is sparse. Therefore, we are interested in a suitable representation, not for storage purposes only, but for recognition and description as well. Representation of a 3-D object by means of oct-trees may make space array (triply subscripted binary array) operations more economical in terms of memory space [4].

A simple approach to analyzing 3-D objects is to model them as polyhedra. This requires a description of the object in terms of vertices, edges, and faces. Modeling 3-D objects in this manner results in substantial compression of the data. In order to handle curved and more complex objects, other representations and models have been investigated [5]–[7]. Binford [5] proposed the concept of a generalized cylinder (or cone) to represent curved 3-D objects. These are defined by a 3-D space curve, known as the axis, and cross section of arbitrary shapes and sizes along the axis. There are an infinite number of possible generalized cones representing a single object. More constraints are needed to get a unique description. Although generalized cones or volume representations imply some surface description, they fail to describe the junctions or surface peculiarities [8]. Also one detects surfaces first from partial views, and only after several different views of the object we have enough data to obtain volume properties. Hence the need to find a suitable surface representation. It is possible to represent arbitrary shapes with generalized cones by making them arbitrarily complex, but their computation is difficult. The generalized cone primitives used in [6] are not sufficient to represent the complicated casting, as has been used in this work. Badler and Bajcsy [7] present a good discussion of the relative merits of surface and volume representation.

Methods for segmentation of range data can be classified as "region" or "edge" based just as in the segmentation of intensity images. Many researchers adopted a method which is most suitable for the input device. For example, Duda *et al.* [9] describe a sequential procedure for determining planar surfaces in a scene from registered range and intensity data. The vertical and horizontal surfaces are obtained directly from the range image by a histogram analysis. Slanted surfaces are assumed to have constant intensity and are obtained from the reflectance image. Milgram and Bjorklund [10] find planar surfaces in a range image by fitting a least squares plane in the small neighborhood of each pixel. Underwood and Coates [11] describe a system for inferring 3-D surface description for

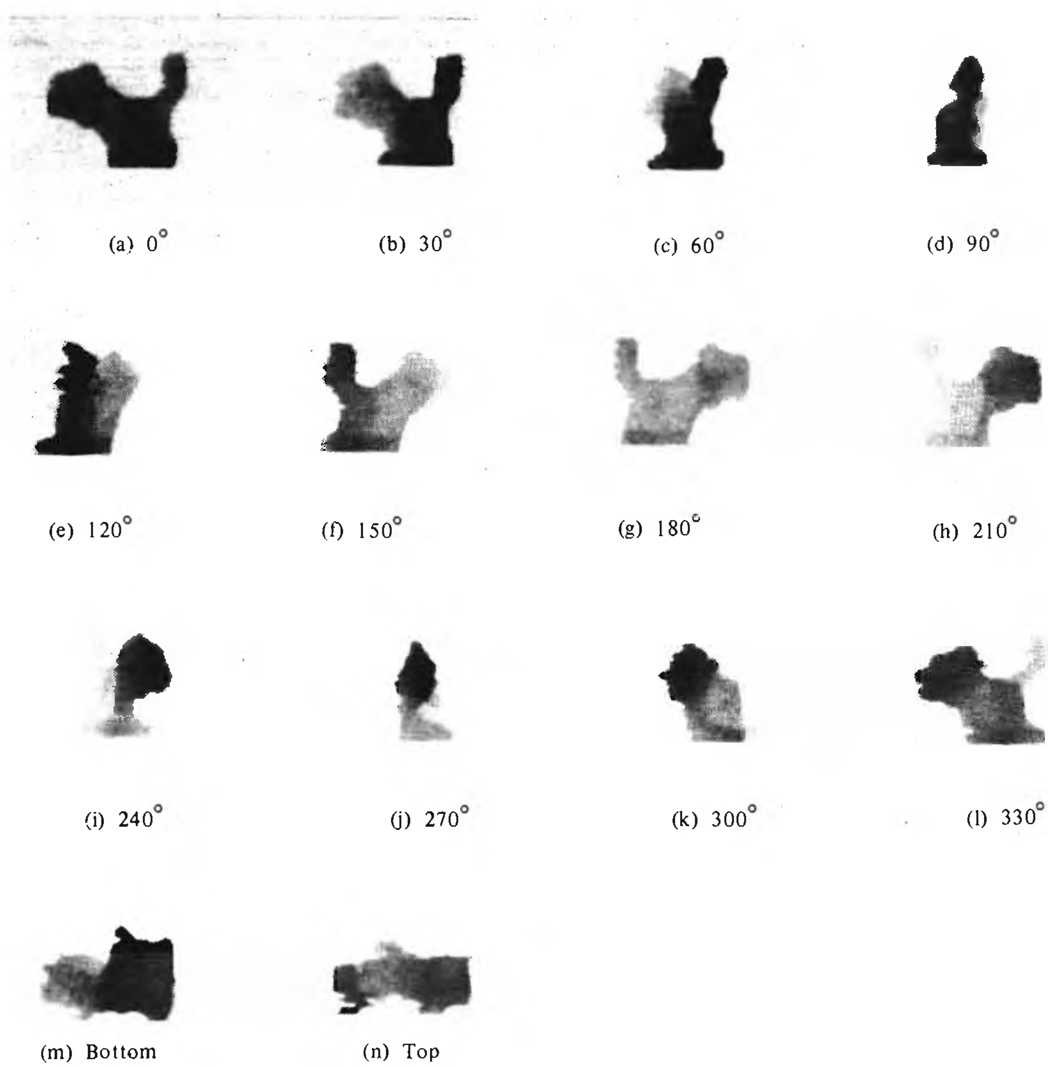

Fig. 4. The 14 range data views of the automobile piece shown as gray scale images. The lighter points are away from the observer and the darker ones are closer.

planar convex objects from a sequence of reflectance images, but the faces are determined from edge information. Ishii and Nagata [12] obtained the contour of an object by controlling a laser spot. Agin [13] fitted quadratic curves to the images of sheets of a laser beam. Shirai [14] and co-workers have used region and edge based methods to represent polyhedrons and simple curved objects. Popplestone *et al.* [15] dealt with polyhedrons and cylinders. Inokuchi and Nevatia [16] and Zucker and Hummel [17] presented techniques for obtaining surface edges. The drawback of these techniques is that the edge responses must be grouped, thinned, and linked in order to produce a reasonable object description in terms of coherent regions. On the other hand, once the line segments are found, the theory of 3-D line semantics can be directly applied. It is possible to extract planar surfaces from single view range data images by extending the iterative endpoint fit method from two dimensions to three [1], [18]. This may work well since the range z can be considered as a function of two spatial coordinates x and y. All the above past techniques except [18] use one range image only. Combining results from several views is a major problem.

Our approach to the analysis of a 3-D range data image is to first extract the relevant 3-D object as sets of 3-D points and then work directly on these sets without regard to the original image. This approach frees one from a particular image when a complete description (3-D model) of the surface of a 3-D object is desired. To obtain a 3-D model of the object, a representation should be complete, that is, it should sample the entire surface of the object, and allow for matching of individual views taken from any arbitrary viewing angle. An object is thus defined by a finite number of selected points in three-space. However, only the geometrical position of each point is known; no topological information is available.

Modeling: Representation and models are intimately connected. Since most of the work in scene analysis has been the interpretation of a 2-D intensity image as a 3-D scene, 2-D models have been commonly used in the analysis with constraints on the configuration of 3-D objects by making use of the *a priori* information about the objects. Such an approach has some inherent problems in that the image of a 3-D object changes with the perspective, it is sensitive to shadows, time of day, weather conditions, and specularity; when several objects occlude each other only parts of some objects are visible in the image and the occluding objects need to be separated from each other. The direct measurement of range simplifies many of these problems considerably.

McKee and Aggarwal [19] recognize partial views of 3-D curved objects like cup or hammer by matching the edge description with the stored model. Their method requires good input of the surface boundaries. Chien and Chang [20] take as input

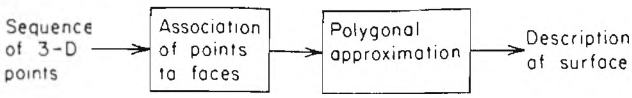

Fig. 5. Two step process to approximate surface by polygons.

a list of vertices in the 2-D line drawing of a 3-D scene of curved objects. The curved edges of a body are represented piecewise linearly and the curved surfaces are represented as a list of vertices with various restrictions. Recognition is accomplished by a model in the form of a tree. Fischler and Elschlager [21] decompose a human face into subparts and construct the model with intensity arrays of subparts, and their configuration. Recognition is performed by matching an input picture to the intensity arrays placed at the best position. Model of subparts can be easily generated, but the matching is sensitive to the scale and shading. Perkins [22] uses 2-D models to recognize nonoverlapping industrial parts on a flat surface. These models can handle some occlusion, but oblique viewing angle problem requires 3-D models.

A tradeoff is involved between representation and modeling. 2-D models make the representation easier at the expense of a complex modeling task. 3-D models are more general, but the representation must take care of mapping it to view-domain. They are very powerful for 3-D shape analysis. Horn [23] uses powerful 3-D surface models of terrains for registration of aerial images. Hierarchical models which involve both 2-D and 3-D models have been used [24]. In this work we generate a 3-D model of the object in terms of planar faces approximated by polygons. The control structure used in shape matching is hierarchical and described in the next section.

Algorithm for Surface Approximation by Polygons: Representing a 3-D object as a set of planar faces approximated by polygons is a two-step process (Fig. 5). In the first step we find the set of points that belong to various faces of the object using a three point seed algorithm [1], [25] and in the second step, approximate the face points obtained in step 1 by polygons.

The three-point seed method for the extraction of planar faces from range data is a model fitting method. It can be viewed as a special case of the Random Sample Consensus (RANSAC) paradigm [26]. It is a sequential region growing algorithm. It is not applied directly to range images, but rather to a set of points. It is not restricted to single view range data image, but applicable to a complete object and does not require the ordering of points. It finds the convex faces of the object, but the information exists to merge convex parts of nonconvex faces. Although the algorithm is applied to a set of 3-D points, it is not directly related to how these points are obtained. The method is ultimately tied to the sampling distance between points on the object.

The 3-Point Seed Method: In a well-sampled 3-D object, any three points lying within the sampling distance of each other (called a 3-point seed) form a plane (called the seed plane) which: a) coincides with that of the object face containing the points, or b) cuts any object face containing any of the three points. A seed plane satisfying a) results in a plane from which a face should be extracted, while a seed plane satisfying b) should be rejected. Two simple conditions that suffice to determine if a plane falls into category b) are: convexity and narrowness. For a given set of points S, the convexity condition requires that for any two points x and y of S, the midpoint of the straight line segment from x to y also lies in S [27]. The characteristic of the set of points obtained after applying the convexity condition is such that when b) occurs, its points all lie essentially on the line passing through two most distant points in the set. Narrowness condition makes a check to determine if it does not happen. The algorithm involves the following steps [1], [25].

1) From the list of surface points select three points which are noncollinear and near relative to sampling distances.
2) Obtain the equation of the plane passing through the three points chosen in step 1.
3) Find the set of points P which are very close to this plane.
4) Apply the convexity condition to the set P to obtain a reduced convex set P'. This separates faces lying in the same plane.
5) Check the set P' obtained in step 4 for narrowness.
6) If the face is obtained correctly (i.e., convexity and narrowness conditions are satisfied), remove the set of points belonging to this face from the list and proceed to step 1 with the reduced number of points in the list.

After the surface points belonging to a face have been obtained, all the points which have been previously associated with various faces are checked for the possible inclusion in the present face. This provides the points which belong to more than one face. This information in turn provides the knowledge about the neighbors of a face and relations among them. The method is applied in stages; the largest faces (in terms of the number of points in the face) are found first, then smaller faces on down to some minimum size. The application of the method in stages is necessary in order to limit the fragmentation of large faces near their extremes. The method requires four thresholds: seed point selection threshold, point to plane threshold, convexity threshold, and narrowness threshold. These thresholds are tied to the sampling distances. The peculiarities of the object to be modeled can be accounted for by the proper choice of these thresholds and the tradeoff involved between the number of faces and the quality of representation can be balanced. After the surface points have been associated with various planar faces some edge points and vertices will be known, however, an independent step is required to obtain polygonal faces. The polygonal approximation of a face is obtained by finding the (x, y, z) coordinates of the boundary points of the face and detecting the points of high curvature [27].

The overall complexity of the 3-point seed algorithm is $O(n^2 \log n)$. Considering isotropic neighborhood of 26 points in 3-D, there are $O(n)$ 3-point seeds. (Note that in a plane each object point can be grouped in 12 ways with its 8 nearest neighbors to produce a seed.) Since in practice the complexity is more dependent on the number of faces than the number of points and the points which have been associated with a face are no longer considered except for finding the points common in different faces, the number of 3-point seeds considered is of the order of number of faces. For each 3-point seed considered, the largest cost is in the convexity test. A straightforward implementation of this test as described in the above is of $O(n^3)$. However, it can be simplified by using a k-d tree [28]. (A k-d tree structure will also aid in the selection of 3-point seed.) A k-d tree is a binary tree of k-dimensional keys (here $k = 3$) which is organized such that at each subdivision step, the data are split at the median along the axis having greatest spread in vector element values along that axis. The data can be organized in a tree structure in $O(n \log n)$ time and it allows the determination of m-nearest neighbors of a given query in $O(\log n)$. Using this tree convexity test can be performed in $O(n^2 \log n)$. This is because for each point in the convex set (in the beginning just the 3-point seed), we have to find midpoint of each of the points in the test set and check if there is a point in the convex set which is near to the midpoint. Since the number of 3-point seeds is proportional to the number of faces, the total complexity of the 3-point seed method is $O(n^2 \log n)$.

An alternate approach to finding planar faces could be a "clustering" type approach [1] which may involve the following steps. 1) Find the reasonable planes. 2) Select the individual faces using connectivity. 3) Consider left over and boundary points etc. This approach has the advantage in that all the faces

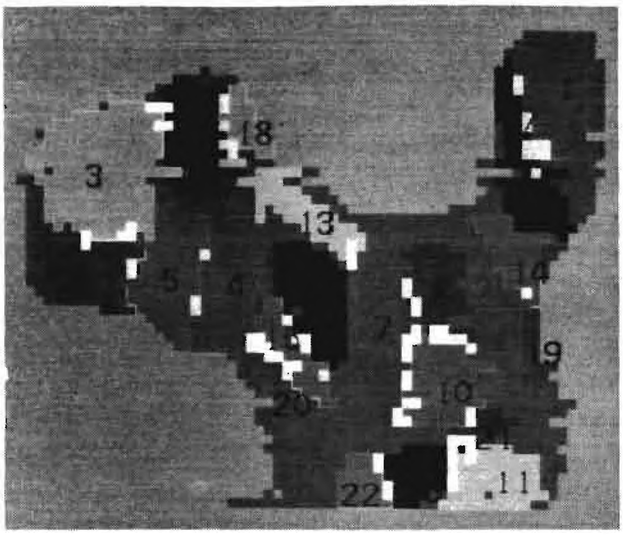

Fig. 6. Faces found in the view shown in Fig. 4(a). There are 22 faces in this view and they are labeled in the order they are found using the algorithm described. The rejected points and the points common to two or more faces are shown in brown and white color, respectively.

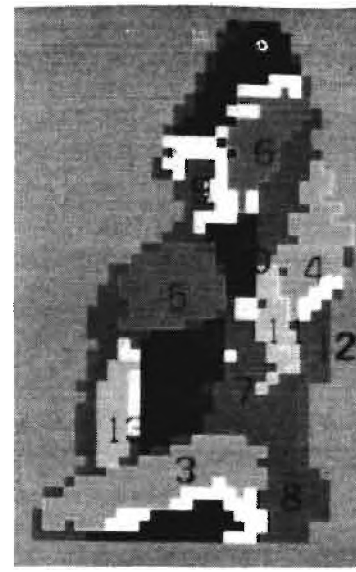

Fig. 7. Faces found in the 90° view [Fig. 4(d)]. There are 14 faces in this view and they are labeled in the order they are found using the algorithm described. The rejected points and the points common to two or more faces are shown in brown and white color, respectively.

TABLE I
LIST OF FACES IN THE 0° VIEW [FIG. 4(a)]. $I1$, $I2$, AND $I3$ ARE THE INDEXES, IN THE LIST OF POINTS FOR THIS VIEW, WHICH MAKE UP A FACE

FACE	I1	I2	I3	No. of Points
1	2	3	11	103
2	69	89	90	84
3	140	141	170	176
4	565	625	626	93
5	573	574	634	94
6	797	798	856	84
7	904	960	961	105
8	782	839	897	64
9	816	817	875	70
10	1310	1350	1391	83
11	1718	1719	1751	67
12	328	329	367	52
13	338	376	377	60
14	597	657	658	43
15	1139	1193	1241	50
16	1725	1758	1759	49
17	38	49	50	35
18	83	84	105	30
19	1165	1166	1213	23
20	1409	1443	1444	36
21	1589	1590	1621	24
22	1766	1767	1799	31

TABLE II
LIST OF FACES IN THE 90° VIEW [FIG. 4(d)]. $I1$, $I2$, AND $I3$ ARE THE INDEXES, IN THE LIST OF POINTS FOR THIS VIEW, WHICH MAKE UP A FACE

FACE	I1	I2	I3	No. of Points
1	1	6	14	82
2	432	455	456	88
3	674	695	718	111
4	143	159	177	60
5	284	303	324	62
6	51	61	75	72
7	385	386	408	43
8	734	757	758	43
9	152	153	170	38
10	227	228	244	42
11	316	317	338	28
12	377	399	400	32
13	531	555	576	37
14	818	819	845	58

(convex or nonconvex) are found at the same time. However, the connectivity used in step 2) will require an ordering of points and if the object does not contain major horizontal or vertical surfaces, step 1) based on Hough transform or obtaining the histogram of z distance or some other local features may be quite expensive.

Surface Approximation Results: The 3-point seed method was applied to the 14 individual views shown in Fig. 4 and to the complete object. Fig. 6 shows the faces found for 0° view. In this figure various faces are shown in different colors. The rejected points and the points common to two or more faces (edge points) are shown in brown and white color, respectively. They are labeled in the order they are found using the 3-point seed algorithm. The points that could not make up a face having at least 20 points were rejected. The area of rejected points fall either on jump points resulting from large z-distance change with correspondingly little x or y change, or they occur in extremely uneven parts of the surface of the object. A rejected point lies inside some of the faces because it has been missed in the process of data acquisition. Also some of the rows have been shifted because of the continuous nature of the data. Table I gives the properties of faces in the 0° view.

Similarly, Fig. 7 shows the faces obtained in the 90° view and Table II lists the properties of these faces. Table III shows the neighbors of a face in the 0° and 90° views. These neighbors are arranged in the descending order of the number of points that they possess. Note that a face may have no neighbors, because a face that could not possess more than a certain minimum number of points was rejected. Different faces have different numbers of neighbors. For example, face 1 in 0° view has face 12 and face 17 as neighbors, and face 11 in 90° view has no neighbors. The method was applied to the complete object to get the 3-D model. In the model 85 faces were found. The number of faces found, and their distribution fits well with the results from the individual views.

IV. SHAPE MATCHING OF 3-D OBJECTS

In 3-D scene analysis we have a model for 3-D objects and a method for matching unknown objects with the model. Milgram and Bjorklund [10] mention preliminary efforts of 3-D matching by using a guided search procedure. The number of flat surfaces in their study is usually small. Fourier descriptors and moments have been used for the recognition of 3-D shapes [29]-[31]. However, moments or Fourier descriptors are global features and cannot solve the important class of problems which require the partial recognition of the shape, because the descriptors of the entire shape do not bear any simple relationship with the descriptors of a part of a shape. Although Wallace

TABLE III
NEIGHBORS OF A FACE IN 0° AND 90° VIEWS. THEY ARE ARRANGED IN THE DESCENDING ORDER BY SIZE. (a) 0° VIEW [FIG. 4(a)]. (b) 90° VIEW [FIG. 4(d)].

FACE NUMBER (a)	NEIGHBORS			FACE NUMBER (b)	NEIGHBORS		
1	12	17	0	1	6	9	0
2	3	13	18	2	7	13	0
3	2	9	0	3	14	8	0
4	5	0	0	4	12	0	0
5	4	9	0	5	0	0	0
6	15	0	0	6	1	10	9
7	10	8	13	7	2	10	0
8	7	10	0	8	3	14	0
9	3	5	0	9	1	6	10
10	7	8	21	10	6	7	9
11	16	21	0	11	0	0	0
12	1	17	0	12	4	0	0
13	7	2	18	13	2	0	0
14	19	0	0	14	3	8	0
15	6	20	0				
16	11	22	21				
17	1	12	0				
18	2	13	0				
19	14	0	0				
20	15	0	0				
21	10	11	16				
22	16	0	0				
(a)				(b)			

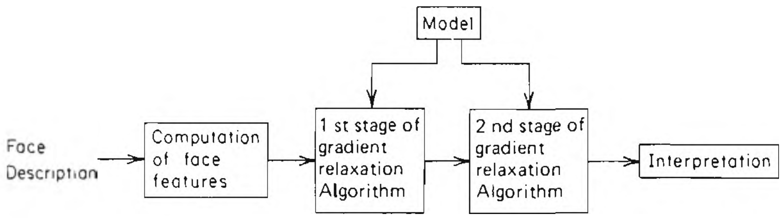

Fig. 8. Block diagram of the 3-D shape matching algorithm.

et al. [29] consider shape analysis of 3-D objects using local shape descriptors, their techniques need major modifications inorder to handle the partial shape recognition problem. Furthermore like Dudani *et al.* [31] these authors are not dealing with the 3-D data, but rather with projections of a 3-D object. Since the image of a 3-D object changes with the viewing angle, they have a large library of three-dimensional projections corresponding to a single object. For example, Dudani *et al.* [31] in the identification of six different aircraft use a training sample set of over 3000 projected images. Oshima and Shirai [32] use range information for the recognition of blocks and simple machine parts by matching the feature and relation based description of the scene with the stored model. The Hough transform technique of Ballard and Sabbah [33] to detect the presence of a 3-D object is based on the fact that all the planar regions be adjacent to each other in the object representation. However, in practice it may not be always feasible. For the complex automobile part used here and the simple parts used by Oshima and Shirai [32], there are faces which are not surrounded by other faces.

Representation and modeling are closely related and the control structures normally depend on the choice of representation. Control structures are defined as the strategy of utilizing the available knowledge to efficiently obtain the goal descriptions. In 3-D scene analysis work bottom up, top down, and a mixture of these two have been used [24]. The hierarchical control structure is a popular choice since it eliminates unnecessary search during the recognition process. Our approach for 3-D shape matching uses planar faces as primitives and matches an unknown view with the structural 3-D model. Since our representation and modeling are based on the prominent actual physical faces of the object, consistency of the segmentation process is assured. This is of importance in shape matching. The control structure of the 3-D shape matching algorithm is hierarchical in the sense that at higher levels of hierarchy more contextual information is used to accomplish the partial shape matching task.

Shape Matching Algorithm: Fig. 8 shows a block diagram of the two stage hierarchical stochastic labeling technique for the shape matching of 3-D objects. Shape matching is performed by matching the face description of an unknown view with the stored model using the available contextual information. The same set of descriptors is used for the description of both the faces of the model and an unknown view.

Let $T = (T_1, T_2, \cdots, T_N)$ and $O = (O_1, O_2, \cdots, O_{L-1})$ be the face representation of an unknown view and the model respectively, where T_i and O_j are planar faces, $i = 1, \cdots, N$ and $j = 1, \cdots, L - 1$. The elements of the unknown view will be referred to as units and elements of the model as classes. We want to identify an unknown view within the model. We are therefore, trying to label each of the faces of an unknown view T_i ($i = 1, \cdots, N$) either as a face O_j ($j = 1, \cdots, L - 1$) or as not belonging to the model O (label O_L = nil). Each face T_i of an unknown view therefore has L possible labels.

To each of the units T_i, we assign a probability $p_i(l)$, $l = 1, \cdots, L$ (using a technique described subsequently) that the unit belongs to class O_k. This is conveniently represented as a probability vector $\vec{p}_i = [p_i(1), \cdots, p_i(L)]^T$. The set of all vectors $\vec{p}_i$ ($i = 1, \cdots, N$) is called a stochastic labeling of the set of units. Units are related to one another through their neighbors. The set of units related to T_i is denoted by V_i. In order to compare the local structure of T and O, the world model is specified by the compatibility functions C_1 and C_2, which are defined over a subset $S_1 \subseteq (N \times L)^2$ and $S_2 \subseteq (N \times L)^3$ for the first and second stage of the hierarchy, respectively. For simplicity, we shall denote compatibility functions $C_1(T_i, O_k, T_j, O_l)$, $T_j \in V_i$ and $C_2(T_i, O_k, T_{i_1}, O_{l_1}, T_{i_2}, O_{l_2})$, $T_{i_1}, T_{i_2} \in V_i$ as $C_1(i, k, j, l)$ and $C_2(i, k, i_1, l_1, i_2, l_2)$, respectively. C_1 and C_2 take values between 0 and 1. $C_1(T_i, O_k, T_j, O_l)$ and $C_2(T_i, O_k, T_{i_1}, O_{l_1}, T_{i_2}, O_{l_2})$ measure the resemblance of the set $\{T_i, T_j\}$ with the set $\{O_k, O_l\}$ and $\{T_i, T_{i_1}, T_{i_2}\}$ with the set $\{O_k, O_{l_1}, O_{l_2}\}$, respectively. We also define a compatibility vector $\vec{q}_i = [q_i(1), \cdots, q_i(L)]^T$ for all the units at each of the stages of hierarchy. Intuitively, this tells us what $\vec{p}_i$ should be given $\vec{p}_j$ at the related units and the compatibility function. Mathematically,

$$q_i^{(j)}(k) = \frac{Q_i^{(j)}(k)}{\sum_{l=1}^{L} Q_i^{(j)}(l)}, \quad j = 1, 2, \quad k = 1, \cdots, L \tag{1}$$

where, at the first stage,

$$Q_i^{(1)}(k) = \sum_{j \in V_i} \sum_{l=1}^{L} C_1(i,k,j,l) p_j(l) \qquad i = 1, \cdots, N, \quad k = 1, \cdots, L \tag{2}$$

and at the second stage,

$$Q_i^{(2)}(k) = \sum_{l_1, l_2 = 1}^{L} C_2(i,k,i_1,l_1,i_2,l_2) p_{i_1}(l_1) p_{i_2}(l_2) \qquad i = 1, \cdots, N, \quad k = 1, \cdots, L \text{ and } i_1, i_2 \in V_i. \tag{3}$$

As discussed in [1], [2], two global criteria that measure the consistency and ambiguity of the labeling over the set of units are given by

$$J^{(j)} = \sum_{i=1}^{N} \vec{p}_i \cdot \vec{q}_i^{(j)}, \quad j = 1, 2. \tag{4}$$

The maximization of (4) results in a reduced inconsistency and ambiguity. Inconsistency is defined as the error between $\vec{p}_i$ and $\vec{q}_i^{(j)}$. Intuitively, this means the discrepancy between what every unit "thinks" about its own labeling ($\vec{p}_i$) and what its neighbors "think" about it ($\vec{q}_i^{(j)}$). Ambiguity is measured by the quadratic entropy and results from the fact that initial labeling $\vec{p}_i^{(0)}$ is ambiguous ($\vec{p}_i^{(0)}$ are not vectors). Note that each term $\vec{p}_i \cdot \vec{q}_i^{(j)}$ is maximum for $\vec{p}_i = \vec{q}_i^{(j)}$ (maximum consistency) and $\vec{p}_i = \vec{q}_i^{(j)}$ = unit vector (maximum unambiguity). The problem of labeling the units T_i is equivalent to an optimization problem: given an initial labeling $\vec{p}_i^{(0)}, i = 1, \cdots, N$, find a local maximum of the criteria $J^{(j)}$ ($j = 1, 2$) closest to the original labeling $\vec{p}_i^{(0)}$ subject to the constraints that $\vec{p}_i$'s are probability vectors. Since C_2 is a better measure than C_1 of the local match between T and O, we are actually interested in finding local maximum of the criterion $J^{(2)}$. On the other hand, maximizing $J^{(1)}$ is easier from the computational standpoint. We therefore use the following hierarchical approach: starting with an initial labeling $\vec{p}_i^{(0)}$, we look for a local maximum $\vec{p}_i^{(1)}$ of the criterion $J^{(1)}$. This labeling is less ambiguous than $\vec{p}_i^{(0)}$ in the sense that many labels have been dropped (their probabilities $p_i(k)$ are equal to zero). We then use the labeling $\vec{p}_i^{(1)}$ as an initial labeling to find a local maximum of the criterion $J^{(2)}$. The computational saving comes from the fact that the values C_2 corresponding to probabilities $p_{i_1}(l_1)$ or $p_{i_2}(l_2)$ equal to zero are not computed. The problem of maximizing (4) is efficiently solved using the gradient projection method [1].

Initial Assignment of Probabilities: The initial probabilities are computed using the features of a face. The features set consists of area, perimeter, length of the maximum, minimum and average radius vectors from the centroid of a face, number of vertices in the polygonal approximation of the boundary of a face, angle between the maximum and minimum radius vectors, and ratio of area/perimeter^2 of a face. Let P be the number of features used. We measure the quality of correspondence between the faces T_i and O_k as

$$M(T_i, O_k) = \sum_{p=1}^{P} |f_{tp} - f_{op}| W_p \tag{5}$$

where

f_{tp} = pth feature value for the face of an unknown view
f_{op} = pth feature value for the face of the model
W_p = weight factor for the pth feature.

Weights of the features are used to account for their importance and range of values. The initial probabilities are chosen proportional to $1/(1 + M(T_i, O_k))$ and normalized so that they sum to 1.

Computation of Compatibilities: The compatibility function determines the degree by which two or three neighboring units are compatible with each other. At the first stage the computation of $C_1(i,k,j,l)$ involves binary relations and at the second stage $C_2(i,k,i_1,l_1,i_2,l_2)$ involves a subset of ternary relations. The compatibility of a face of an unknown view with a face in the model is obtained by finding transformations, applying them and computing the error in feature values. At the first stage, we find two transformations $TR1$ and $TR2$ such that

$$TR1: T_i \to O_k \quad \text{and} \quad TR2: T_j \to O_l.$$

Now $TR1$ is applied to T_j giving matching error $M(TR1(T_j), O_l)$ and $TR2$ is applied to T_i giving matching error $M(TR2(T_i), O_k)$, where matching error is given by

$$M(TR(T_m), O_n) = \sum_{p=1}^{P} |f_{t'p} - f_{op}| W_p \tag{6}$$

where $f_{t'p}$ = pth feature value for the transformed unit, and other quantities are similar to those defined in (5). Features used in computing (6) are (x, y, z) centroid, area, orientation, and rotation.

The average of these two errors is obtained and

$$C_1(i,k,j,l) = \frac{1}{1 + \text{average error}}.$$

At the second stage instead of finding two transformations, we find three transformations and the average error will be the average of six error terms and the compatibility

$$C_2(i,k,i_1,l_1,i_2,l_2) = \frac{1}{1 + \text{average error}}.$$

The transformations used in computing C_1 and C_2 are based on:

1) scale, the ratio of area of two faces;
2) translation, difference in the centroidal coordinates of the two faces;
3) orientation, difference in the orientation of two faces so that they are in the same plane;
4) rotation, to obtain maximum area of intercept, once the two faces are in the same plane; it is found with an accuracy of 45°.

The problem of defining p_i (nil), C_1 and C_2 when some of the faces in the unknown view are matched to the nil class is solved as follows [1]. p_i (nil) is assigned a small constant value,

TABLE IV
LABELS AT DIFFERENT ITERATIONS FOR THE FACES SHOWN IN FIG. 6. EXAMPLE 1.

FACE NUMBER	FIRST STAGE ITERATION NUMBER			SECOND STAGE ITERATION NUMBER		
	0	1	3	1	3	6
1	86(.10)	86(.22)	1(.37)	1(.41)	1(1.0)	1(1.0)
2	86(.10)	86(.13)	2(.19)	2(.28)	2(.48)	2(1.0)
3	3(.35)	3(.80)	3(1.0)	3(1.0)	3(1.0)	3(1.0)
4	86(.10)	86(.22)	86(.29)	86(.33)	86(.36)	4(1.0)
5	86(.10)	86(.21)	5(.33)	5(.42)	5(.61)	5(1.0)
6	86(.10)	86(.24)	6(.36)	6(.46)	6(1.0)	6(1.0)
7	7(.11)	7(.62)	7(1.0)	7(1.0)	7(1.0)	7(1.0)
8	21(.19)	21(.60)	21(1.0)	21(1.0)	21(1.0)	21(1.0)
9	86(.10)	22(.38)	22(1.0)	22(1.0)	22(1.0)	22(1.0)
10	24(.18)	24(.60)	24(1.0)	24(1.0)	24(1.0)	24(1.0)
11	25(.14)	25(.37)	25(.53)	25(.69)	25(1.0)	25(1.0)
12	86(.10)	86(.26)	86(.33)	86(.41)	86(.78)	86(1.0)
13	86(.10)	33(.61)	33(1.0)	33(1.0)	33(1.0)	33(1.0)
14	86(.10)	86(.35)	86(.35)	86(.34)	34(.58)	86(1.0)
15	86(.10)	86(.33)	86(.40)	86(.52)	86(1.0)	86(1.0)
16	86(.10)	53(.18)	53(.36)	53(.52)	53(1.0)	53(1.0)
17	86(.10)	86(.34)	86(.44)	45(.53)	45(1.0)	45(1.0)
18	46(.10)	46(.22)	46(.53)	46(.67)	46(1.0)	46(1.0)
19	86(.10)	86(.27)	86(.30)	86(.27)	67(.39)	50(1.0)
20	86(.10)	86(.32)	86(.35)	86(.34)	34(.59)	34(1.0)
21	86(.10)	86(.16)	56(.34)	50(.53)	50(1.0)	50(1.0)
22	86(.10)	86(.26)	86(.34)	86(.34)	79(.39)	79(1.0)
Value of Criterion	--	1.091	8.094	9.419	13.463	20.857

$J^{(1)}$ $J^{(2)}$

depending upon the *a priori* information, between 0.05 to 0.30. Its actual value is not critical, however, it affects the convergence of probabilities, hence the number of iterations. Compatibilities involving nil class are assigned as follows:

$$C_1\ (i, k, j, \text{nil}) = C_2\ (i, k, i_1, \text{nil}, i_2, \text{nil}) = p_i\ (k)$$

$$C_1\ (i, \text{nil}, j, l) = C_2\ (i, \text{nil}, i_1, l_1, i_2, l_2) = p_i\ (\text{nil})$$

$$C_2\ (i, k, i_1, \text{nil}, i_2, l_2) = C_1\ (i, k, i_2, l_2)$$

$$C_2\ (i, k, i_1, l_1, i_2, \text{nil}) = C_1\ (i, k, i_1, l_1).$$

Examples and Comments: In testing the shape matching algorithm we consider three unknown views shown in Fig. 4 (a), (b), and (l) corresponding to 0°, 30°, and 330°, respectively. Although the model is obtained with these views included, the model, as previously explained, does not contain all the faces corresponding to each unknown view. This is due to the procedure by which the surface points corresponding to the complete object were obtained. Therefore, the use of these unknown views is justified for the evaluation of the shape matching technique. It is noted that the shape matching algorithm does not assume that the unknown view was among the set of the model building views. It can be any arbitrary view. The number of faces in an unknown view of the automobile piece varied from 10 to 25 and the number of faces in the model is 85. In matching, only the best 29 faces of the model are considered, in order to reduce the complexity of the matching task. The evaluation of the compatibility vector $q_i^{(j)}$ $(j = 1, 2)$ requires the knowledge about the neighbors of a face [see (1)]. The larger neighbors are given preference over the smaller neighbors, when a unit has several neighbors and only a subset of them are considered in the computation of compatibilities. Normally we have considered the number of neighbors to be 1 in the computation of $q_i^{(1)}$ and 2 neighbors in the computation of $q_i^{(2)}$ for all the units. If a unit has only one neighbor while computing $q_i^{(2)}$, then compatibility C_1 is used instead of C_2. If a unit has no neighbors, then this unit is firmly assigned to the best matched class at the time of computation of initial probabilities.

Example 1: Fig. 6 shows the faces found in the 0° view shown in Fig. 4(a). Table III(a) shows the neighbors of the faces. The neighbors are arranged by size in descending order. Table IV shows the results of labeling at different iterations. Only the label with the highest probability of assignment is shown. In the bracket we have indicated this probability. Label 86 is the nil class. One way of checking the results of labeling is to compute the relative orientation of the object using the final assignment of units. To compute the orientation, we need to compute the transformation matrix T in

$$T\boldsymbol{x} = \boldsymbol{b} \tag{7}$$

where

$$T = \begin{bmatrix} l_1 & l_2 & l_3 \\ m_1 & m_2 & m_3 \\ n_1 & n_2 & n_3 \end{bmatrix}, \quad \boldsymbol{x} = \begin{bmatrix} x' \\ y' \\ z' \end{bmatrix}, \quad \boldsymbol{b} = \begin{bmatrix} x \\ y \\ z \end{bmatrix}$$

and (l_1, m_1, n_1), (l_2, m_2, n_2), and (l_3, m_3, n_3) are the direction cosines of the x', y', z', axis (unknown view) relative to x, y, z coordinates (model), respectively.

From the results of matching the transformation matrix T is obtained by selecting three units, (called a triple of units) which are not assigned to the nil class, and solving a set of nine linear equations to evaluate the nine coefficients of the matrix T. The (x, y, z) and (x', y', z') of any triple are taken as the centroids of the matched model face and an unknown view face, respectively. From the results shown in Table IV several triple of units such as (1, 2, 3), (1, 3, 4), (1, 13, 16), (16, 17, 18), (5, 6, 7), (2, 3, 4), (2, 17, 18) produce the coefficients of the matrix T very accurately. For example, the triple (5, 6, 7) solves the matrix T as

$$\begin{bmatrix} 1.00000 & -0.00305 & -0.01102 \\ -0.00534 & 1.00000 & 0.00586 \\ -0.00916 & 0.00000 & 1.00000 \end{bmatrix}.$$

The arc cos of coefficients (1, 1) or (3, 3) (rotation in the x-z plane around y axis) give 0° as the relative orientation of the unknown view with respect to the model. This is in agreement

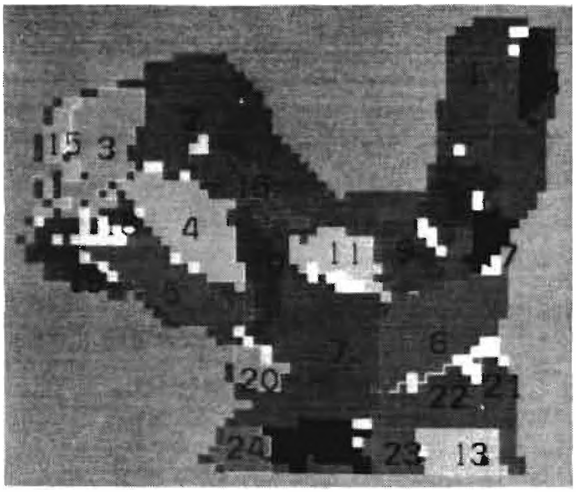

Fig. 9. Faces found in the view shown in Fig. 4(b). There are 24 faces in this view and they are labeled in the order they are found using the algorithm described. The rejected points and the points common to two or more faces are shown in brown and white color, respectively.

with the true orientation for this view. Note that the coefficients (1, 3) and (3, 1) of the transformation matrix should ideally be equal to zero. Translation can be obtained by finding the difference in the centroids of the matched faces. The total computation time for surface approximation, matching and the determination of orientation for this view is 566.4 s.

Whenever using the local matching results to obtain a global information such as the determination of orientation, a fundamental problem arises in that how we can use the matching results to come up with a unique answer. In general, it is possible that the matching results of any three units may not give the correct direction cosines as indicated by the values of the coefficients of the matrix T. All the coefficients of the matrix T should be within ± 1. So if the labeling of any of the three selected units happens to be wrong, the direction cosines will be errorneous. Moreover, since we are interested only in the approximate matches not the exact matches as they may not exist and measurement errors are possible, it is quite likely that some triples do not lead to the valid direction cosines. Also different triples may lead to slightly different solutions. There are several approaches to obtain the solution for this problem. For example an average of several valid solutions (coefficients of T within ± 1) can be taken or more precisely the problem could be formulated as a least square problem subject to the constraints that T is a rotation matrix. For the results presented in this paper the three units needed for the computation of the transformation matrix T have been arbitrarily chosen provided none of them is assigned to the nil class and their values are within the interval $[-1, 1]$.

Example 2: Fig. 9 shows the faces found in the 30° view shown in Fig. 4(b). There are 24 faces in this view and they are labeled in the order they are found. Table V shows the neighbors of the faces. Comparing Figs. 6 and 9, it can be seen how some of the faces of Fig. 9 should be labeled. For example, faces 11, 7, and 21 in Fig. 9 correspond to faces 8, 10, and 21 in Fig. 6, respectively. Similarly the correspondence for some other faces can be obtained and the matching results can be verified by using Tables IV and VI. As in the Example 1, various triple of units allow us to compute the transformation matrix T. For example, using the triple (4, 7, 8), matrix T is obtained as

$$\begin{bmatrix} 0.88383 & 0.09058 & 0.46854 \\ -0.20947 & 1.00000 & -0.19653 \\ -0.45183 & 0.01863 & 0.86441 \end{bmatrix}.$$

TABLE V

NEIGHBORS OF THE FACES SHOWN IN FIG. 9. THEY ARE ARRANGED IN THE DESCENDING ORDER BY SIZE.

FACE NUMBER	NEIGHBORS			
1	8	14	0	0
2	4	16	0	0
3	18	15	0	0
4	5	2	18	0
5	4	18	19	20
6	22	17	21	0
7	11	0	0	0
8	1	9	17	0
9	8	0	0	0
10	20	0	0	0
11	7	0	0	0
12	22	23	0	0
13	23	0	0	0
14	1	0	0	0
15	3	18	0	0
16	2	0	0	0
17	6	8	0	0
18	5	4	3	15
19	5	0	0	0
20	5	10	0	0
21	6	22	0	0
22	6	12	21	0
23	12	13	0	0
24	0	0	0	0

Note that the matrix T is not strictly a rotation matrix. For example, the coefficients (1, 1) and (3, 3) are not equal and the magnitude of the coefficients (1, 3) and (3, 1) is not identical. This is because of the inherent measurement errors and the exact matches may not exist and we have not explicitly constrained T to be a rotation matrix. However, the average of the coefficients (1, 1) and (3, 3) can be taken and we can use its arc cos to obtain a reasonable estimate of the rotation information. Following this the relative rotation in the x-z plane of about 30° is obtained for the view shown in Fig. 9. The total computation time for this view is 425.6 s.

Example 3: Fig. 10 shows the faces obtained in the 330° view shown in Fig. 4(l). There are 24 faces in this view and as before they are labeled in the order they are found. Neighbors of the faces are shown in Table VII. Comparing Figs. 6, 9, and 10 one can observe how the face description has changed. Also it can be inferred how the faces of Fig. 10 should be labeled with respect to the labeling of the faces in Figs. 6 and 9. For example, face 13 of Fig. 6 and face 11 of Fig. 10 match with the model face 33. Face 22 of Figs. 9 and 10 match with the model face 77. A few labels such as for faces 13 and 21 appear only in this view. Such labels have been verified independently with the model. Table VIII shows the results of the stochastic labeling. Most of the labels are correct, but a few of them are wrong because of the higher degree of similarity of the local structure of the incorrect match with the model. An example of such an incorrect label is for the face 18 which matches with the model face 79. Actually, face 22 of Fig. 6 matches with the model face 79. As in the previous examples, triples of units can be used to compute the transformation matrix T. For example, using the triple (2, 5, 6), matrix T is obtained as

$$\begin{bmatrix} 0.89699 & -0.02619 & -0.58716 \\ -0.14116 & 1.00000 & -0.01034 \\ 0.49117 & -0.02597 & 0.79737 \end{bmatrix}.$$

Using the discussion presented above in the Examples 1 and 2, a relative orientation of about 330° in the x-z plane is obtained. The total comutation time for this view is 1022 s.

V. CONCLUSIONS

In this paper we presented representation, modeling and matching techniques incorporated in a 3-D scene analysis system. A geometric technique is used to approximate surfaces

TABLE VI
LABELS AT DIFFERENT ITERATIONS FOR THE FACES SHOWN IN FIG. 9. EXAMPLE 2.

FACE NUMBER	FIRST STAGE ITERATION NUMBER			SECOND STAGE ITERATION NUMBER		
	0	1	3	1	4	7
1	86(.08)	86(.08)	86(.19)	86(.21)	1(.56)	1(1.0)
2	86(.08)	86(.17)	86(.21)	2(.25)	2(.74)	2(1.0)
3	86(.08)	86(.18)	2(.27)	86(.26)	2(1.0)	2(1.0)
4	86(.08)	86(.16)	5(.21)	5(.32)	5(1.0)	5(1.0)
5	86(.08)	86(.17)	86(.20)	86(.31)	5(.49)	5(1.0)
6	86(.08)	86(.18)	86(.21)	86(.30)	86(1.0)	86(1.0)
7	86(.08)	86(.17)	86(.20)	86(.25)	24(.57)	24(1.0)
8	86(.08)	86(.16)	20(.20)	20(.27)	20(.80)	20(1.0)
9	86(.08)	86(.24)	34(.34)	34(.39)	34(1.0)	34(1.0)
10	86(.08)	86(.23)	35(.30)	35(.35)	45(1.0)	45(1.0)
11	86(.08)	86(.22)	34(.30)	34(.40)	34(.61)	53(1.0)
12	86(.08)	86(.19)	26(.23)	26(.31)	26(.49)	53(1.0)
13	86(.08)	86(.30)	86(.33)	40(.33)	53(1.0)	53(1.0)
14	86(.08)	86(.24)	86(.31)	64(.41)	86(1.0)	86(1.0)
15	86(.08)	86(.20)	86(.25)	86(.31)	86(.54)	86(1.0)
16	86(.08)	86(.23)	86(.27)	41(.29)	86(1.0)	86(1.0)
17	86(.08)	86(.20)	86(.25)	86(.28)	62(.39)	62(1.0)
18	86(.08)	86(.28)	77(.38)	86(.54)	86(1.0)	86(1.0)
19	86(.08)	86(.16)	86(.20)	65(.25)	65(1.0)	65(1.0)
20	86(.08)	86(.16)	86(.22)	86(.22)	50(.54)	52(1.0)
21	86(.08)	86(.17)	70(.22)	86(.26)	86(.36)	50(1.0)
22	86(.08)	86(.23)	77(.39)	77(.50)	77(1.0)	77(1.0)
23	86(.08)	86(.19)	86(.23)	52(.27)	52(.39)	52(1.0)
24	86(1.0)	86(1.0)	86(1.0)	86(1.0)	86(1.0)	86(1.0)
Value of Criterion	--	1.874	3.647	4.327	12.606	23.017
		$J^{(1)}$			$J^{(2)}$	

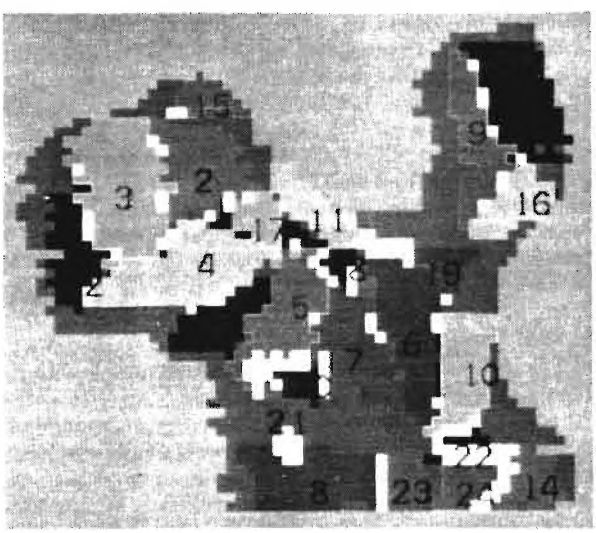

Fig. 10. Faces found in the view shown in Fig. 4(l). There are 24 faces in this view and they are labeled in the order they are found using the algorithm described. The rejected points and the points common to two or more faces are shown in brown and white color, respectively.

TABLE VII
NEIGHBORS OF THE FACES SHOWN IN FIG. 10. THEY ARE ARRANGED IN THE DESCENDING ORDER BY SIZE.

FACE NUMBER	NEIGHBORS			
1	9	16	0	0
2	3	4	17	15
3	4	2	12	0
4	3	2	12	13
5	7	20	18	0
6	7	10	11	19
7	6	5	20	18
8	21	23	0	0
9	1	16	0	0
10	6	22	0	0
11	6	0	0	0
12	3	4	0	0
13	4	0	0	0
14	24	22	0	0
15	2	0	0	0
16	1	9	0	0
17	2	0	0	0
18	7	5	0	0
19	6	0	0	0
20	7	5	21	0
21	8	20	0	0
22	10	24	14	0
23	8	0	0	0
24	14	22	0	0

by planar faces. The results of shape matching are good. A few incorrect assignments result because the structure and description of a unit with its neighbors matches better with incorrect match than the correct match. Also if the object has some symmetry, it is likely that there will be multiple matches. The results of shape matching depend upon the planar surface approximation, its consistency and neighborhood information. An approximation of the surface of an object which includes planar and curved faces which are contiguous (there are no rejected points) and which provides complete neighborhood information will be desirable since then contextual information will be more effective. The number of views to obtain a model depends upon the complexity of the object. The computation time for surface approximation of an unknown view, matching and the determination of orientation varied from about 7–20 min. on a PDP-10 (KL-10 processor). Over 95 percent of this time is spent in the computation of rotation needed in the compatibility computation. This is because we store only the boundary of the image of a face. Also we do not store the compatibility values, and recompute them when the gradient is required [1]. By storing the images of the faces and the compatibility values, computation time will be much smaller. It can be further cut in certain situations by assuming that the object is rigid and it can have only a finite number of stable positions. The results of labeling allow us to obtain the orienta-

TABLE VIII
LABELS AT DIFFERENT ITERATIONS FOR THE FACES SHOWN IN FIG 10. EXAMPLE 3.

FACE NUMBER	FIRST STAGE ITERATION NUMBER			SECOND STAGE ITERATION NUMBER		
	0	1	3	1	4	8
1	86(.08)	86(.15)	5(.21)	86(.30)	86(.62)	86(1.0)
2	2(.08)	2(.18)	2(.34)	2(.46)	2(1.0)	2(1.0)
3	86(.08)	86(.15)	86(.22)	86(.24)	86(.28)	3(1.0)
4	86(.08)	86(.13)	5(.20)	5(.25)	5(.50)	5(1.0)
5	86(.08)	86(.17)	6(.24)	6(.35)	6(1.0)	6(1.0)
6	86(.08)	86(.17)	7(.32)	7(.41)	7(1.0)	7(1.0)
7	86(.08)	86(.16)	86(.22)	7(.32)	7(1.0)	7(1.0)
8	86(.08)	86(.21)	86(.25)	86(.38)	86(.71)	86(1.0)
9	86(.08)	86(.18)	86(.22)	86(.50)	86(1.0)	86(1.0)
10	86(.08)	86(.15)	86(.20)	24(.24)	24(.41)	24(1.0)
11	86(.08)	86(.21)	86(.25)	86(.26)	33(1.0)	33(1.0)
12	86(.08)	86(.19)	86(.25)	86(.36)	86(.61)	86(1.0)
13	86(.08)	86(.21)	86(.22)	86(.29)	27(.53)	27(1.0)
14	86(.08)	86(.24)	86(.27)	86(.39)	86(.59)	34(1.0)
15	86(.08)	86(.13)	86(.20)	86(.23)	46(.58)	46(1.0)
16	86(.08)	86(.23)	34(.30)	86(.36)	34(1.0)	34(1.0)
17	86(.08)	46(.13)	46(.27)	46(.33)	46(.73)	46(1.0)
18	86(.08)	86(.18)	86(.24)	79(.35)	79(.60)	79(1.0)
19	86(.08)	86(.21)	86(.27)	86(.31)	70(.53)	70(1.0)
20	86(.08)	86(.24)	86(.28)	86(.38)	86(1.0)	86(1.0)
21	86(.08)	86(.23)	86(.30)	86(.29)	47(.36)	72(1.0)
22	86(.08)	86(.27)	77(.68)	77(1.0)	77(1.0)	77(1.0)
23	86(.08)	86(.16)	86(.19)	67(.21)	86(.71)	86(1.0)
24	86(.08)	86(.31)	86(.36)	86(.41)	36(1.0)	36(1.0)
Value of Criterion	--	.9126	2.759	3.867	9.358	23.50
		$J^{(1)}$			$J^{(2)}$	

tion of the object in three-space. Translation information can also be obtained. Normally, we used 3 iterations at the first stage and 4 to 8 iterations at the second stage. We found that these two stages of hierarchy are sufficient for matching purposes, although the method generalizes to include higher levels at the expense of increased computation. The first stage does not resolve all the ambiguous labelings. The second stage helps in correcting these labelings. These matching results could be useful in controlling a robot manipulator on an assembly line or inspection stages of the production. The shape matching technique presented here can be extended to handle occlusion of two or more objects by following the algorithm discussed in [34].

REFERENCES

[1] B. Bhanu, "Shape matching and image segmentation using stochastic labeling," Image Processing Inst., Univ. Southern California, Los Angeles, USCIPI Rep. 1030, Aug. 1981.

[2] B. Bhanu and O. D. Faugeras, "Shape recognition of 2-D objects," in *Proc. 2nd Scandinavian Conf. Image Anal.*, Helsinki, Finland, June 15-17, 1981, pp. 72-77.

[3] J. D. Boissonnat and F. Germain, "A new approach to the problem of acquiring randomly oriented workpieces out of a bin," in *Proc. 7th Int. Joint Conf. Artificial Intell.*, Vancouver, Canada, Aug. 24-28, 1981.

[4] C. L. Jackins and S. L. Tanimoto, "Oct-trees and their use in representing three-dimensional objects," *Comput. Graphics Image Processing*, vol. 14, pp. 249-270, 1980.

[5] T. O. Binford, "Visual perception by computer," in *Proc. IEEE Conf. Syst. Contr.*, Miami, FL, Dec. 1971.

[6] R. Nevatia and T. O. Binford, "Description and recognition of curved objects," *Artificial Intell.*, vol. 8, pp. 77-98, 1977.

[7] N. Badler and R. Bajcsy, "Three-dimensional representations for computer graphics and computer vision," *ACM Comput. Graphics*, vol. 12, pp. 153-160, Aug. 1978.

[8] R. Bajcsy, "Three dimensional scene analysis," in *Proc. 5th Int. Conf. Pattern Recognition*, Miami Beach, FL, Dec. 1980, pp. 1064-1074.

[9] R. O. Duda, D. Nitzan, and P. Barrett, "Use of range and reflectance data to find planar surface regions," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. PAMI-1, pp. 259-271, July 1979.

[10] D. L. Milgram and C. M. Bjorklund, "Range image processing: Planar surface extraction," in *Proc. 5th Int. Conf. Pattern Recognition*, Miami Beach, FL, Dec. 1980, pp. 912-919.

[11] S. A. Underwood and C. L. Coates, Jr., "Visual learning from multiple views," *IEEE Trans. Comput.*, vol. C-24, pp. 651-661, June 1975.

[12] M. Ishii and T. Nagata, "Feature extraction of three-dimensional objects and visual processing in a hand-eye system using a laser tracker," *Pattern Recognition*, vol. 8, pp. 229-237, 1976.

[13] G. J. Agin and T. O. Binford, "Computer description of curved objects," in *Proc. 3rd Int. Joint Conf. Artificial Intell.*, 1973, pp. 629-640.

[14] Y. Shirai, "Recent advances in 3-D scene analysis," in *Proc. 4th Int. Joint Conf Pattern Recognition*, Kyoto, Japan, Nov. 1978, pp. 86-94.

[15] R. J. Popplestone *et al.*, "Forming models of plane-and-cylinder faced bodies," in *Proc. 4th Int. Joint Conf Artificial Intell.*, Tbilisi, U.S.S.R., 1975, pp. 664-668.

[16] S. Inokuchi and R. Nevatia, "Boundary detection in range pictures," *Proc. 5th Int. Conf Pattern Recognition*, Miami Beach, FL, Dec. 1980, pp. 1301-1303.

[17] S. W. Zucker and R. A. Hummel, "An optimal three-dimensional edge operator," in *Proc. IEEE Conf. Pattern Recognition and Image Processing*, 1980, pp. 162-168.

[18] J. D. Boissonnat and O. D. Faugeras, "Triangulation of 3-D objects," in *Proc. 7th Int. Joint Conf. on Artificial Intell.*, Vancouver, Canada, Aug. 24-28, 1981, pp. 658-660.

[19] J. W. McKee and J. K. Aggarwal, "Finding the edges of the surfaces of three-dimensional curved objects by computer," *Pattern Recognition*, vol. 7, pp. 25-52, 1975.

[20] R. T. Chien and Y. H. Chang, "Recognition of curved objects and object assemblies," in *Proc. 2nd Int. Joint Conf. Pattern Recognition*, Denmark, Aug. 13-15, 1974, pp. 496-510.

[21] M. A. Fischler and R. A. Elschlager, "The representation and matching of pictorial structures," *IEEE Trans. Comput.*, vol. C-22, pp. 67-92, Jan 1973.

[22] W. A. Perkins, "A model based vision system for industrial parts," *IEEE Trans. Comput.*, vol. C-27, pp. 126-143, Feb. 1978.

[23] B.K.P. Horn and B. L. Bachman, "Using synthetic images to register real images with surface models," in *Proc. Image Understanding Workshop*, Oct. 1977.

[24] T. Kanade, "Model representations and control structures in image understanding," in *Proc. 5th Int. Joint Conf. Artificial Intell.*, 1977, pp. 1074-1082.

[25] T. C. Henderson and B. Bhanu, "Three point seed method for the extraction of planar faces from range data," in *Proc. IEEE Int. Conf Industrial Applications of Machine Vision*, Raleigh, NC, May 3-5, 1982, pp. 181-186.
[26] M. A. Fischler and R. C. Bolles, "Random consensus: A paradigm for model fitting with applications in image analysis and automated cartography," *Commun. Ass. Comput. Mach.*, vol. 24, pp. 381-395, June 1981.
[27] A. Rosenfeld and A. C. Kak, *Digital Picture Processing*. New York: Academic, 1976.
[28] J. H. Friedman *et al.*, "An algorithm for finding best matches in logarithmic expected time," *ACM Trans. Math. Software*, vol. 3, pp. 209-226, Sept. 1977.
[29] T. P. Wallace, O. R. Mitchell, and K. Fukunaga, "Three-dimensional shape analysis using local shape descriptors," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. PAMI-3, pp. 310-323, May 1981.
[30] F. A. Sadjadi and E. L. Hall, "Three-dimensional moment invariants," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. PAMI-2, pp. 127-136, Mar. 1980.
[31] S. A. Dudani *et al.*, "Aircraft identification by moment invariants," *IEEE Trans. Comput.*, vol. C-26, pp. 39-45, Jan. 1977.
[32] M. Oshima and Y. Shirai, "Object recognition using three-dimensional information," in *Proc. 7th Int. Joint Conf. Artifical Intell.*, Vancouver, Canada, Aug. 24-28, 1981, pp. 601-606.
[33] D. H. Ballard and D. Sabbah, "On shapes," in *Proc. 7th Int. Joint Conf Artificial Intell.*, Vancouver, Canada, Aug. 24-28, 1981, pp. 607-612.
[34] B. Bhanu, "Recognition of occluded objects," in *Proc. 8th Int. Joint Artificial Intell.*, Karlsruhe, West Germany, Aug. 8-12, 1983.

A Syntactic Approach to 3-D Object Representation

W. C LIN AND K. S. FU

Abstract—**A 3-D object representation scheme which uses surfaces as primitives and grammatical production rules as structural relationship descriptors is proposed. Possible selections of surface primitives are discussed. Examples are given to illustrate the object description method.**

Index Terms—**Computer vision, origami world, primitive surface, syntactic approach, 3-D object representation, 3-D-plex grammar.**

I. Introduction

Computer representation of three-dimensional (3-D) objects has attracted the attention of researchers of scene analysis and computer graphics in the past several years [1]-[3], [25]. In model-based approach of image recognition, a 3-D object model is constructed in order to match its 2-D perspective transformation to a specific object in a 2-D picture. The 3-D object can also be displayed by projection methods for computer graphics applications.

In this paper, we propose a 3-D object description scheme using surfaces as primitives and grammatical production rules as structural relationship descriptors. It is well known that the syntactic approach to pattern recognition provides a capability for describing a large set of complex patterns by using small set of simple pattern primitives and grammatical rules [4]. As will be seen in Section III, one of the most attractive aspects of this capability is the use of a recursive nature of a grammar. A grammar (rewriting) rule can be applied any number of times, so it is possible to express in a very compact way some basic structural characteristics of an infinite set of sentences. Another important feature of this modeling scheme is that it unambiguously specifies how the surface patches are assembled which facilitates surfaces identification in computer vision applications.

In the next section, we briefly review several different schemes proposed in the literature of machine vision and computer-aided design. Then, in Section III, possible selections of surface primitives are discussed. The modeling grammar—3-D-plex grammar—is described. An algorithm to derive a sentence from a left parse and a 3-D-plex grammar is presented. Several examples are given to illustrate the modeling procedures. Finally, in Section IV, the representation scheme is evaluated based on general criteria for judging the effectiveness of a method of structural object representation.

Manuscript received May 4, 1983; revised July 29, 1983. This work was supported by the National Science Foundation under Grant ECS-81-19886.

The authors are with the School of Electrical Engineering, Purdue University, West Lafayette, IN 47907.

II. Descriptions of 3-D Objects

Depending on the types of "building blocks" used in the model construction process, there are three general classes of representation for 3-D rigid solid: 1) surface or boundary, 2) sweep, and 3) volumetric.

A. Boundary Representations

With these methods, a 3-D solid object is represented by segmenting its boundary (or enclosing surfaces) into a finite number of bounded subsets usually called "faces" or "patches" and describing the structural relationships between the segmented faces [5].

Designers involved in ship, automobile, and airplane building are using computer graphics display to help visualize prototype shape and changes to existing designs [6]. A number of approaches, Coons patches, bicubic surface patches, Bezier methods, Hermite methods, and *B*-splines, for example, have been devised [7]-[9].

Another approach to surface representation is to express the surfaces as functions on the "Gaussian sphere" (the distance from the origin to a point on the surface is a function of the direction of the point, or of its longitude and latitude if it were radially projected on a sphere with the center at the origin). This class of surfaces, although restricted, is useful in some application areas, such as modeling of human heart [10], [11].

An influential system for using face-based representations for planar polyhedral objects, is the "winged edge" representation [12]. Such a representation can be made efficient for accessing all faces, edges, or vertices; for accessing vertex or edge parameters; for polyhedron building; and for splitting edges and faces.

In [26], a set of manipulative operations for boundary models of solid objects has been presented to construct a solid modeling system. They are designed for CAD/CAM environments rather than computer vision applications. The building block in the system are a set of "atomic" functions called the Euler operators which work on the topology of a boundary model, that is, on the relative arrangement of its faces, edges, and vertices. The destructive and the creative operations allow the system to perform arbitrary modifications necessary for boundary representation models whose faces are planar polygons.

Since surfaces are what is seen, the boundary representations are important for computer vision. For certain objects, primarily those constructed from thin sheet-like material, surface descriptions are natural for representation purposes. However, for conventional boundary representation schemes, correct

0162-8828/84/0500-0351$01.00 © 1984 IEEE