Efficient 3-D Object Representations for Industrial Vision Systems

THOMAS C. HENDERSON

Abstract—The representation of 3-D objects is an important step in solving many problems in scene analysis. One of the most successful techniques is that based on the surfaces of objects. We describe several methods for obtaining such surface representations from various types of intrinsic images. In particular, previous work is reviewed and an algorithm based on region growing is investigated in terms of its efficiency in segmenting a set of points in 3-D space into planar faces. Information on the neighborhood structure of the points in the form of a spatial proximity graph is used to direct the segmentation. Applications to industrial objects are demonstrated.

Index Terms—Spatial proximity graph, surface segmentation, 3-D representations.

I. Introduction

CURRENT machine vision systems impose severe constraints on the work environment and are generally restricted to rigid objects observed from standard viewpoints in high contrast lighting with no shadows or occlusion allowed. Moreover, recognition and analysis are based on 2-D representations of the objects. Although many tasks can be engineered to meet these constraints, many important problems cannot. For example, recognition of nonrigid objects or of workpieces on overhead conveyors, bin picking, and automatic vehicle guidance are beyond the ability of current vision systems. Rosen [44] describes these problems in detail. We believe that one major stumbling block to the development of more powerful computer vision techniques is the traditional 2-D array image representation. A survey of three-dimensional representations is presented, and a particularly efficient structure, the underlying 3-D neighborhood graph is proposed as the basic low-level representation of image information.

To overcome the problems outlined above, the long-range goal of computer vision research is to provide a description of the objects in a scene and the relations between these objects. Most early work on the subject dealt strictly with intensity images or with filtered versions of intensity images, e.g., edge images or line drawing images, and the major problems posed were segmentation and object identification. These problems were found to be more difficult than they were originally thought to be, and workers turned their attention from high-level problems such as object identification to a closer investigation of the kind and quality of information available in intensity images. The most successful current work is the fruit of that labor and has provided a much better understanding of low-level vision processes and the information available from intensity images. However, vision systems have a long way to go before they will be general purpose (see Binford [10] for a review).

Marr [35] shows how distance and surface orientation can be derived from intensity images, and calls the set of images describing this information the 2.5-D sketch. Another approach is that of Tennenbaum *et al.* [50]; they describe a set of intrinsic characteristics recoverable from intensity images and represent these characteristics as a set of arrays registered with the original image. These arrays are called intrinsic images and include images of surface orientation, surface reflectance, and incident illumination. Other characteristics are possible: range, color, etc. Several other approaches exist to determine intrinsic images, and in particular depth information: structured light [41], shape from shading [30], and even direct range finders [21].

Taking account of these developments, we see that a general purpose vision system should provide several levels of representation. At the lowest level is the sensor information (usually an intensity image), and at the highest level is a symbolic description of objects in the scene and their relations. In between these extremes, there may be several intermediate levels; for example, among the levels proposed by Tennenbaum are:

Level 0—Original image
Level 1—Intrinsic surface characteristics
Level 2—3-D Surface descriptions
Level 3—3-D Object descriptions
Level 4—Symbolic description of scene.

The computational problems involved in deriving Level 1 from Level 0 are fairly well understood now. The next step from Level 1 to Level 2 is just starting to receive attention.

Clearly, the choice of object representation at Level 3 will directly influence the possible methods for surface extraction. There are currently three major representation schemes for 3-D objects: volume, surface, and skeleton representations (see Bajcsy [4] for an overview of these methods). However, for recognition and matching all three methods depend to some extent on the surfaces of the objects.

Several kinds of volume representations exist for describing three-dimensional shape. Srihari [49] gives an introduction to the subject but the emphasis is on voxel-based images. (Voxels stands for volume element.) The representations discussed are the following.

Topological:

- *3-D Euler Characteristic*—The shape is encoded in terms of the number of components, tunnels, and enclosed cavities.
- *Adjacency Tree*—A structure which describes the containment relation of the background, components, and holes.

Manuscript received August 23, 1982; revised June 15, 1983.
The author is with the Department of Computer Science, University of Utah, Salt Lake City, UT 84112.

0162-8828/83/1100-0609$01.00

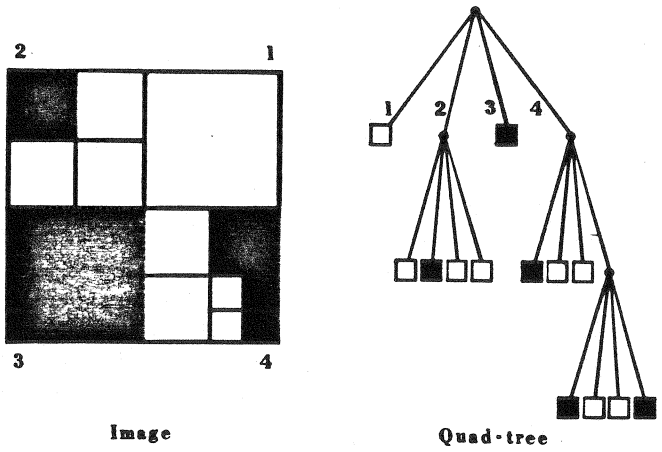


Fig. 1. A sample image and its corresponding quad-trees.

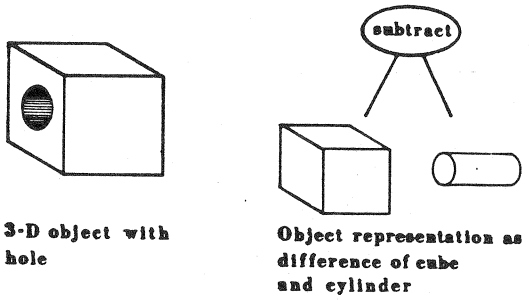


Fig. 2. A cube with a cylindrical hole through it and its representation as the difference of two primitive solids.

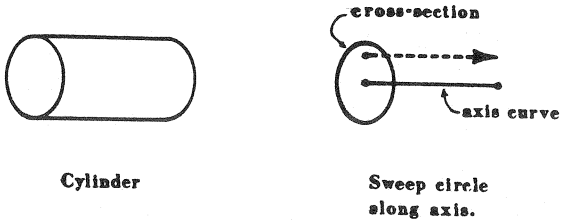


Fig. 3. Generalized cylinders.

Geometrical:

- *Borders*—The voxels on the boundary of the shape.
- *Medial Axis Transform*—The 3-D extension of Blum's 2-D transform [11].
- *Features*—Extensions of standard 2-D feature models to 3-D are given, including analytic functions, Fourier descriptors, moments, and convexity measures.

Spatial Organization:

- *Generalized Cylinders*—Shape is described in terms of a 3-D space curve and an associated cross section.
- *Generalized Blobs*—A set of primitive objects are given and relations can be defined on them.
- *Skeleton*—Stick figures capture the structure of the shape.

Another representation not mentioned above is the oct-tree (see Jackins and Tanimoto [33]). See Fig. 1 for a sample image and corresponding quad-tree. The oct-tree is the three-dimensional analog of a quad-tree. The oct-tree of a volume is based on successive subdivisions of the volume into octants. A uniformly valued octant is represented by a leaf in the tree, while a nonuniformly valued octant is further subdivided. Such a representation can be used for geometric modeling and space planning. Jackins and Tanimoto give efficient algorithms for space array operations such as rotations and translations of oct-tree representations.

Requicha [43] has described a volume representation based on constructive solid geometry. A set of primitive solids is given and defines the lowest possible level. The usual set operations are defined on these primitives and any 3-D object to be modeled is represented as a combination (under the defined operations) in terms of other solids, all ultimately defined in terms of the primitive solids. Fig. 2 demonstrates this approach. Unfortunately, until techniques are developed for obtaining and processing 3-D volume imagery, this type of model does not lend itself well to computer graphics techniques.

The use of 3-D skeleton models (or generalized cones) has been explored by a number of people (see Agin [1] or Nevatia and Binford [37]). A generalized cone is defined by a 2-D shape and a curve (or axis) in space along which the shape is translated. The 2-D shape may vary along the axis as defined by a sweeping function which describes the variation in shape. Fig. 3 illustrates this approach. Hierarchical representations are possible, too, in which objects are described as combinations of a few primitive generalized cones. Two approaches exist to extract generalized cones from range data, namely, an analysis of ribbons and a search for ellipses. Brooks *et al.* [15] describe the use of ribbons which are 2-D generalized cones, while Bolles and Fischler [13] describe fitting ellipses to structured light data. One major disadvantage of this method is that it is not easy to describe objects which have no axis of symmetry, i.e., the sweeping function is unduly complicated. Even so, this method offers several advantages, including hierarchical structure, compact representation, and local descriptions.

Surface (or boundary) representations describe solids in terms of their boundaries. This is analogous to 2-D shape representations based on the silhouette or periphery of a planar shape. 3-D boundaries, however, are represented in terms of points, curves and surfaces which permit "faces" to be defined. In order to make the problem of surface representation tractable, it is necessary to define the computational problem to be solved. This allows relevant mathematical results to be used. Since the surface is to be defined in terms of "faces," this implies that the notion of "face" must be examined carefully. As Brown [16] points out, even when the type of face allowed is specified, and thus, the class of possible surfaces implied, it may still be difficult to determine if a given object constitutes a valid boundary. For example, Hall *et al.* [25] define a "simple surface" as a function describing the surface, and, in particular, they fit a quadric surface equation to the data points. As mentioned previously, such representations specify efficiently the geometrical information necessary for graphics and for machine operations that are numerically controlled. Although many types of surface description can be formulated, we shall be concerned with polyhedral and cylindrical approximation to surfaces. One of the major advantages of a polyhedral approximation is that it allows the use of the powerful techniques of computational geometry to find the convex parts of objects and to test the intersections of objects. This is essential, for example, to determine a collision-free

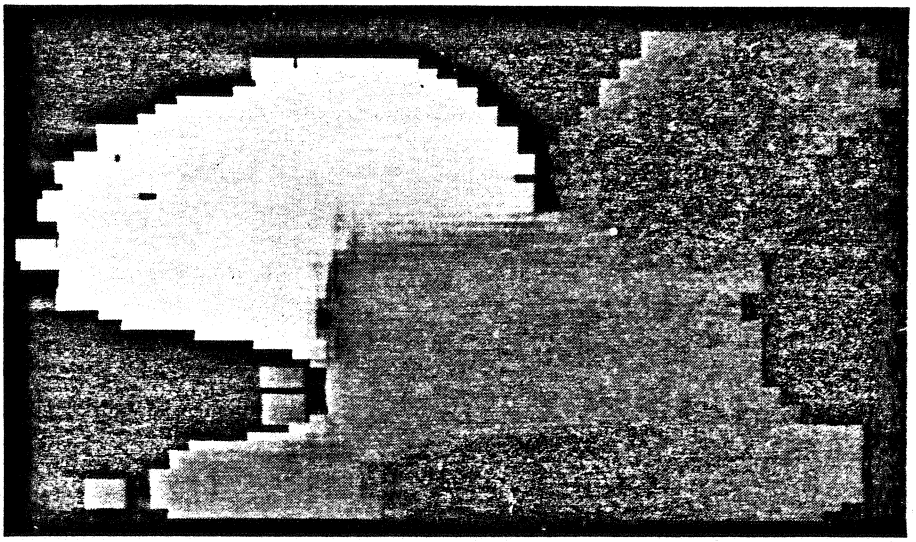


Fig. 4. Range data image.

course through a group of obstacles [3], [14], [17], [34], [52]. Moreover, the use of computer graphics techniques to display the results is simple to implement with this type of representation and allows for the development of interactive methods for model construction.

We develop and compare methods for automatic surface approximation of a set of points in 3-D space. Polyhedral approximation is a two-step process consisting of a local analysis of points followed by a grouping operation to produce faces. This process is studied in the context of an underlying graph which represents the neighborhood structure of the points. This structure can be exploited by all these methods to achieve a more rapid segmentation. This graph-based segmentation process can be viewed as a generalization from the implicit image array topology to that defined by a surface graph. Moreover, all 3-D representation schemes can, in principle, take advantage of this low-level organizational structure. Other approaches to the recovery of structure in point sets include the use of the minimal spanning tree [55], the relative neighborhood graph [51], and the Voronoi triangulation [2]. However, if what is desired is a description of all points within some fixed distance for every point in the set, then none of the structures listed above gives that information explicitly, nor can it be derived efficiently from them.

II. Related Work

The methodology proposed here assumes that 3-D information is available in the form of points in 3-space, i.e., range data. Many methods are currently available which provide a scene analysis system either directly or indirectly with range data (see [21], [24], [25]). Figs. 4 and 5 show the types of data available, namely, range data images and sets of points in 3-space, respectively. One major goal of this paper is to describe efficient algorithms to approximate surfaces in such data. However, the extraction of 3-D surfaces from range data is a difficult problem since range data varies smoothly at surface boundaries and contains "jump" edges at occluding edges. Even so, some progress has been made. Most methods treat the surface extraction problem as one of direct image processing; that is, given a range data image, segment the image into connected planar regions. The problem can be posed, however, in a more abstract manner. Namely, given a set of 3-D points, partition the set into subsets of points, where each subset constitutes a planar connected region. Alternatively, one might try to characterize the polyhedra which have the set of points as surface points. The latter approach might work well if the set of points is known to constitute a single object, but will perform poorly in a cluttered environment.

As an example of a problem treated by this approach, consider the problem of finding the convex hull of a set of points. A set is convex if and only if for every pair of points in the set, the line segment joining the pair of points is completely contained in the set. The convex hull of a set is the smallest convex set containing the set. Shamos [45] shows that for sets of planar points, an algorithm exists that runs in Order(N) expected time and has Order($N\log N$) worst case behavior. Preparata and Hong [42] describe an Order($N\log N$) algorithm for sets of 3-D points. Although in some circumstances the convex hull of a set of points suffices as a description, it is clear that nonconvex polyhedra require a complete polyhedral representation. It would be convenient if computational geometry provided optimal algorithms for 3-D problems, but as Bentley and Shamos [8] point out, computational geometry has not successfully addressed problems in more than two dimensions.

Although various types of surfaces can be extracted from range data (e.g., see Hall *et al.* [25] or Agin [1]), there are several criteria which make polyhedral approximations useful as object representations:

- they are general enough to represent wide classes of objects;
- they are compact;
- they permit reasoning in terms of abstract notions, e.g., faces, convex parts, holes, etc.;
- they allow hierarchical representations, i.e., the description of more complicated objects as collections of simpler objects;
- they permit an easy interface to graphics display; and

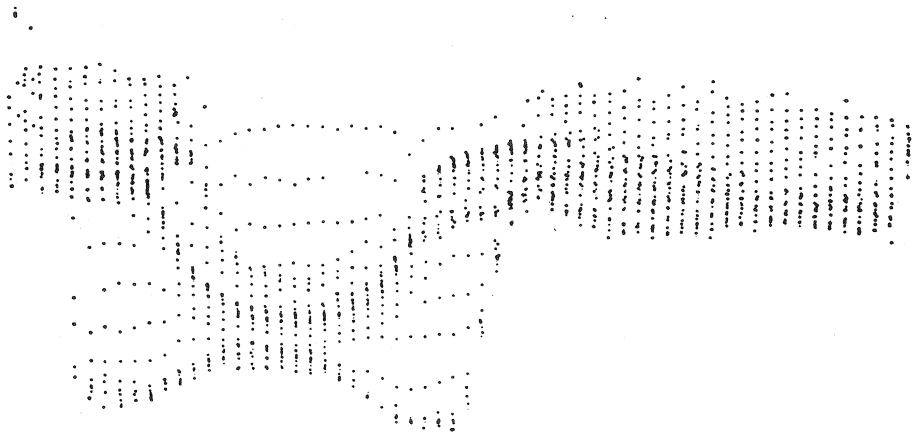


Fig. 5. Set of points in 3-space.

- they provide direct information concerning surfaces which is most often the information desired.

We now briefly review surface extraction methods.

Existing methods for segmentation of range data into faces can be divided into "region" and "edge" based, just as in the segmentation of intensity images. Many researchers adopt a method which is most suitable for their input device. For example, Duda *et al.* [21] 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 data by histogram analysis. Slanted surfaces are assumed to have constant intensity and are obtained from the intensity image. Finally, an iterative procedure refines the initial set of surfaces found. Milgram and Bjorklund [36] find planar surfaces in a range image (consisting mostly of planar regions) to determine the actual sensor position with respect to a given reference model. Planar surfaces are found by fitting a least squares plane in the small neighborhood of each pixel. The orientation of the plane, its altitude from the sensor, and goodness of fit are recorded as local features of each pixel. Finally, pixels are examined for their association with their neighbors to form connected components. This step uses essentially a standard region coloring algorithm. When the range image consists of complex objects having nonplanar surfaces, too, this approach is not suitable.

Underwood and Coates [53] describe a system for inferring 3-D surface descriptions for planar convex objects from a sequence of views (reflectance images), and the faces are determined from edge information (also, see Baker [5]). Several workers have proposed the identification of space curves in terms of projections (see [46], [47], [54]). Ishii and Nagata [32] obtained the contour of an object by controlling a laser spot. Shirai [48] and co-workers have used region and edge based methods to represent polyhedral and simple curved objects. Popplestone *et al.* [41] dealt with polyhedra and cylinders using light planes. Inokuchi and Nevatia [31] present a technique for obtaining surface edges, effectively a 3-D edge operator, while Zucker and Hummel [56] describe an optimal 3-D surface edge operator which produces a smooth surface separating adjacent volumes. 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 faces from a single view range data image by extending the iterative endpoint fit method (see Duda and Hart [20]) from 2 to 3 dimensions. This may work well since the range z can be considered as a function of two spatial coordinates, x and y. All the above techniques apply to one range data image at a time. Combining the results from several views is a major problem.

Many shape analysis methods are based on a planar face representation. Even those systems which allow curved surfaces, usually determine the curved faces by grouping elemental planar faces. Bhanu [9] uses polyhedral approximations to 3-D objects in his relaxation-based matching scheme. Object recognition is performed by Oshima and Shirai [39] in terms of properties of planar surfaces and their relations. Ballard's generalized Hough transform approach to 3-D modeling assumes planar faces as part of the input [6]; also see the planar face based 3-D Hough transform proposed by Henderson and Keskes [28] and Henderson and Mitiche [29]. Finally, the planar face representation has been used in the analysis of time-varying imagery by Dreschler [19] with good results. These are just a few examples of the many 3-D shape representation schemes based on polyhedral approximation.

A polyhedron of minimal surface area has been suggested by O'Rourke [38] as the most natural polyhedral model for a set of given 3-D points. Let S be the set of points, then P is a minimal surface area polyhedron for the set of 3-D points S if and only if the set of vertexes of P is identical to S, and no other simple polyhedron whose vertex set is S has a smaller surface area. The idea is to start from the convex hull of S and modify a given polyhedron to include new points. The method has several serious drawbacks. The problem of finding the minimal polyhedron is thought to be NP-hard. Moreover, the heuristic algorithm given is Order(N^3) and can be seriously misled by certain inputs. Worst of all, the algorithm is not guaranteed to reduce the residual error to get closer to the minimal polyhedron.

Several alternative approaches to polyhedral approximation are possible, including region growing, local feature clustering,

and divide-and-conquer techniques. Region growing methods attempt to locate small seed areas on the surface that are interior to planar faces with a high degree of certainty, and then to grow outward by testing whether or not neighboring areas satisfy the conditions necessary to join the expanding region. Local feature clustering involves measuring some feature at each point in the date (e.g., the surface normal at that point), then applying standard clustering techniques to the resulting vectors. Obviously, such methods are sensitive to the calculation of the features, and features should be chosen that a small change in the local arrangement of the sample points leads to small changes in the vector produced. Finally, the ubiquitous divide-and-conquer strategy applies a recursive algorithm to two halves of the data and produces a fast method. Unfortunately, most divide-and-conquer methods work on 3-D volumes, not surfaces. However, certain results from computational geometry do carry over to 3-D; for example, the minimum spanning tree can be found in Order(Nlog$N$) time.

One method of tremendous potential is the application of clustering techniques to local features measured on the range data. For a review of the various methods and applications of clustering see Diday *et al.* [18]. Clustering methods are required to be fast, to take neighbor constraints into account, and to allow the use of supplementary information.

The problem of surface extraction can be posed as a clustering problem by associating a feature vector with each of the detected points and then using a clustering method to group these vectors. The basic motivation of this approach then is to exploit the fact that points lying on the same face yield similar feature vectors based on the characterization of the plane containing the face and the neighborhood relation. It remains to be seen whether current clustering methods are robust enough to permit the extraction of faces from range data.

Divide-and-conquer methods have been successfully used to solve problems in multidimensional space (see Preparata and Hong [42], Bentley and Shamos [8], and Bentley [7]), and and divide-and-conquer approach that seems to lend itself immediately to the polyhedral approximation problem is that of triangulation. Namely, given a set S of N points, join these points by straight line segments such that any two line segments intersect only at points in the set S, and interior to the convex hull, every region is a triangle. An Order(Nlog$N$) algorithm exists based on Voronoi diagrams to solve this problem for sets of points in the plane (Shamos [45]), but the 3-D generalization partitions space into tetrahedra. One graph-guided divide-and-conquer approach to polyhedral approximation of a set of points on a surface in 3-D space has been proposed by Boissonnat and Faugeras [12]. A graph structure is imposed on the set of points sampled on the surface, and the algorithm proceeds by approximating the surface with triangles. The essential innovation here is to use an underlying graph to structure the data for association with approximating triangles. Whenever a triangle does not approximate very well the data associated with it, then graph methods are used to divide up the data and associate the parts uniquely with a better approximating triangle. The time complexity of the algorithm is Order (Nlog$N$), and the results obtained on a variety of objects (sphere, bar-bell, and helix) are very encouraging; that is, the triangulation is excellent with very few defective areas [22].

III. Graph-Guided Surface Segmentation

In this section we describe an efficient low-level representation for the organization and analysis of sets of 3-space points.

A. Basic Organization of the Data

Given a set of points in 3-D space, we assume that there exists a graph describing these points and their neighbor relations; this graph will be called the spatial proximity graph and will be designated as $G(V, E)$ or as G, where V is the vertex set of G, and E is the edge set. If this information is not directly available from the scanning system, then G can be constructed as follows.

- Let $P = \{X_i, Y_i, Z_i)\}, i = 1, N$ be the set of points. Form a k-d tree [23] using the three coordinate values X_i, Y_i, and Z_i as a 3-D key. A k-d tree is a binary tree for sorting k-dimensional keys, and its structure is optimized with respect to the number of records accessed for nearest neighbor queries. Nonterminal nodes represent some subset of the data and describe how that subset is split at the next level. A subset is split at the median value along the axis with the greatest spread in the data on that axis. (This step can be performed in Order(Nlog$N$) time.)
- Find the m nearest neighbors of each point, where m is the likely number of neighbors within one raster distance. (This step can be performed in Order(log$N$) time for each point.)
- Threshold according to the known sampling information and reject any of the m nearest neighbors which are too far away to be actual neighbors. (Alternatively, incorporate this step into the neighbor query.)

This yields a graph G describing the neighborhood of each point; this graph is a generalization of the (planar) region adjacency graph of Pavlidis [40]. However, the implicit assumption is that the points really lie on some surface, and that the surface is essentially 2-D. (This may not be a good model, for example, of a grassy surface.) The k-d tree organization of the data allows rapid access to the m-nearest neighbors of any point and thus provides the essential information for further analysis. Fig. 6 shows the spatial proximity graph generated for the data in Fig. 5.

B. Region Growing: Three-Point Seed Method

The 3-point seed method is a sequential region growing algorithm and works on sets of 3-D points (see Bhanu [9], Henderson [26], and Henderson and Bhanu [27]). Planar convex faces are determined by sequentially choosing three very close noncollinear points and investigating the set of points lying in the plane of these three points. By taking advantage of the nearest neighbor information in the k-d tree of the data, the complexity of finding planar faces is reduced to Order(Nlog$N$). Suppose that every 3-point seed grows into an acceptable face, then the number of 3-point seeds will be on

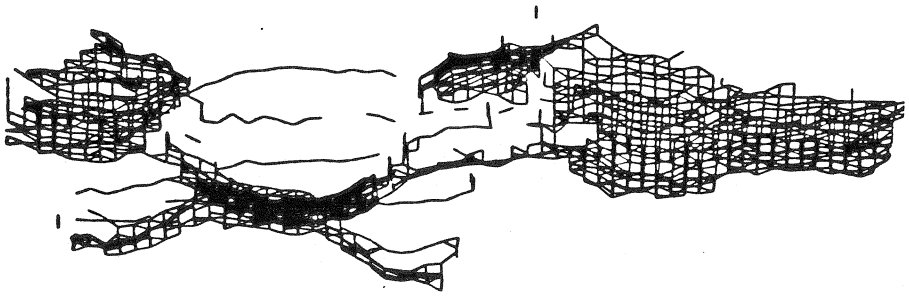


Fig. 6. Spatial proximity graph of points in Fig. 5.

```
(* Check every point not yet part of a plane as the possible seed of a
plane *)

for i := 1 to number_of_points do
  begin
  if Unmarked(i) then
    begin
    Find_plane(i,a,b,c,d,found_plane);
    if found_plane then
      begin
      Mark(i);
      Mark_neighbors(i);
      Put_neighbors_on_queue(i);
      number_in_face := Count_queue + 1;

(* while the queue is not empty, check if its elements lie in the plane *)

    while Not_empty do
      begin
      index := Queue_head;
      for j := 1 to number_of_neighbors do
        begin
        next := graph[index, j];
         if Unmarked(next) then
          if In_plane(next,a,b,c,d) then
            begin
            number_in_face := number_in_face + 1;
            Mark(next);
            Enqueu(next);
            end;
        end;
      end;

(* Save face if there are enough points contributing to it *)

    if number_in_face > minimum_allowed_in_face then
      begin
      Save_face;
      face_number := face_number + 1;
      end;
    end;
  end;
end;
```

Fig. 7. Spiral plane fitting algorithm.

the order of the number of faces of the object. This number is independent of the number of sample points and can be considered constant for any given scene. Now, for every 3-point seed considered, initialize the test face set of points as the 3 points of the seed and enter the 3 points on a queue; next, investigate the neighbors of the points on the queue. If the queue is not empty, then take a point off the queue. Consider each neighbor in turn; if the neighbor lies on the plane defined by the seed points, then add it to the face set and put on the queue any of its neighbors not already on the queue. Since all N points should pass through the queue one time, and since the planarity test takes constant time, the worst case time complexity is thus Order(N) (see Fig. 7). However, the construction of the neighborhood graph is the most expensive step as it takes Order(Nlog$N$) operations.

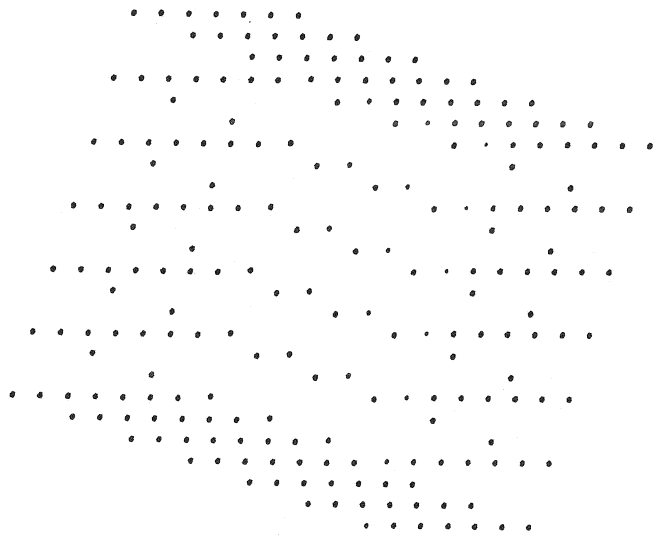


Fig. 8. A set of surface points sampled from a cube.

Both synthesized objects and an industrial piece used in an automobile suspension system have been analyzed with the proposed method for surface approximation by polygons. Results obtained on synthetic cubes were exactly the faces of the cubes, and faces found on synthetic spheres were reasonable (see Figs. 8-11). A more difficult case is that of the complicated casting shown in Fig. 12. This object does not contain any major horizontal or vertical surfaces. Fourteen views were obtained with a range data acquisition system, and three of these are shown in Figs. 13-15. Distance is encoded as a function of gray scale with darker areas nearer and lighter ones more distant. In this figure the lighter a point is the farther away it is. After thresholding the background points, each view shown has about 2000 points. Surface points for the composite object were obtained by rotating the points in 12 of the views, and control points were used to transform the top and bottom views. For each view, the required transformation was applied, and the distance between the transformed point and the points already in the list (in the beginning just the 0 degree view points) was used to add previously unrepre-

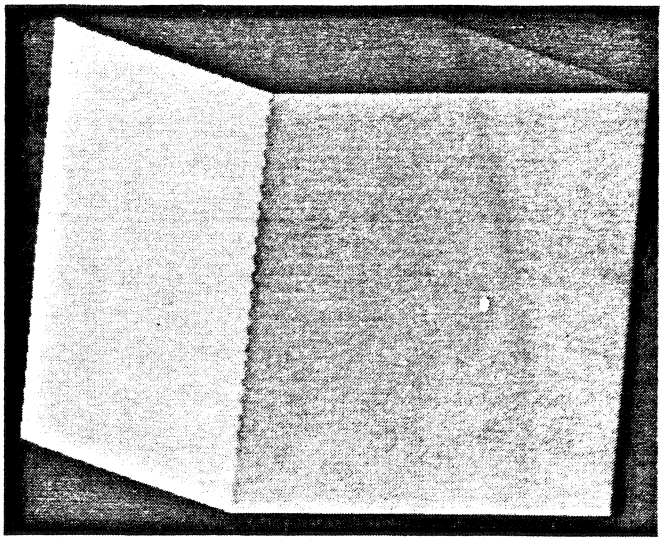


Fig. 9. Faces found for cube.

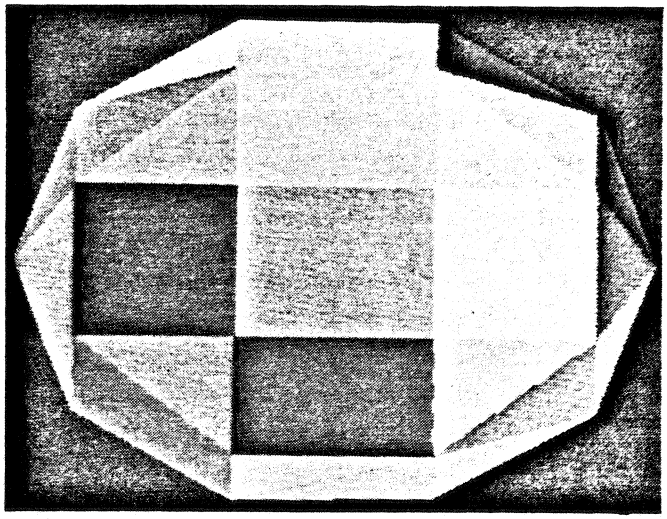


Fig. 11. Faces found for sphere.

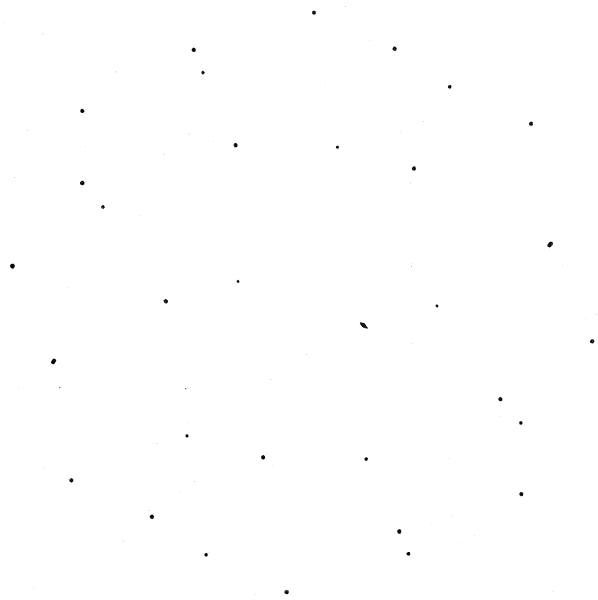


Fig. 10. A set of surface points sampled from a sphere.

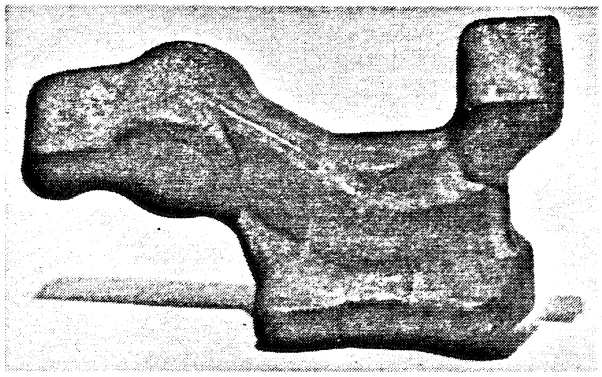


Fig. 12. Industrial part.

sented points to the list. With all the views combined, 8334 points were produced for the composite object.

The 3-point seed method was applied to the 14 individual views and to the composite object. Fig. 16 shows the faces found for the 0 degree view of the object. The points that could not make up a face of at least 20 points were rejected. The area of rejected points falls 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. As can be seen, most of the faces found are reasonable. The object has major curved surfaces that were split symmetrically into different faces. Edge points are not very well detected. A total of 85 faces were found for the composite object, and their distribution fits well with the results from the individual views.

The major advantages of this method include its applicability to a composite object as it is not restricted to single view range data images, and the robust result produced even for noisy data. Moreover, the expected time complexity is very low.

IV. Summary and Future Work

The major goal is to develop efficient surface extraction methods. This requires a thorough investigation of the fundamental issues of surface representation, and in particular, of the notion of "face." Thus, we hope to characterize the definition most suitable for a given class of objects to be modeled. No matter which 3-D object representation is chosen, the spatial proximity graph provides a convenient representation for the spatial organization of the original 3-D data points. Once the representation is chosen, particular methods of surface extraction can be analyzed. We have investigated the graph-based segmentation of 3-D object surfaces. A region growing algorithm which takes advantage of the spatial proximity graph which describes the neighborhood structure of the points in

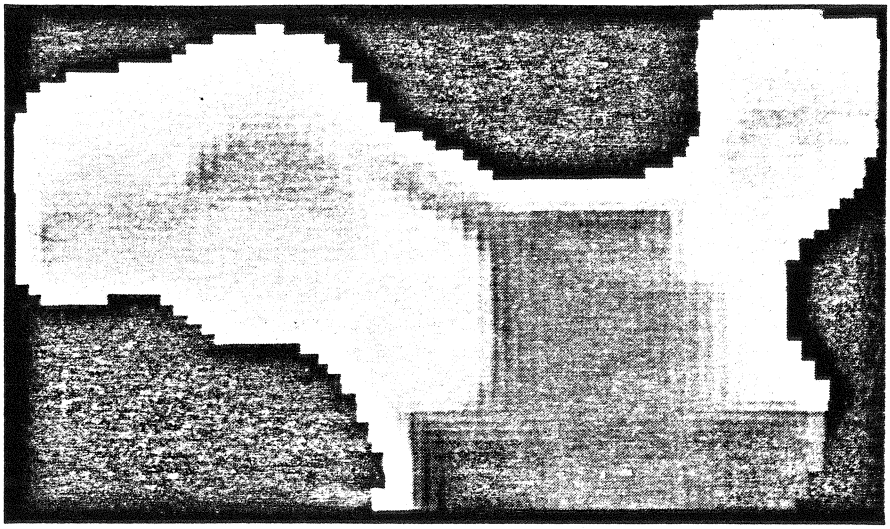


Fig. 13. 0 degree range data image of part.

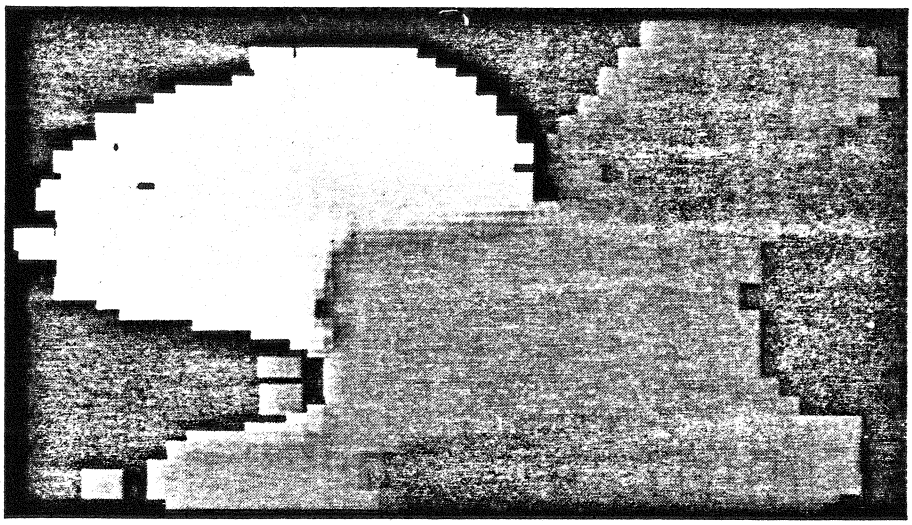


Fig. 14. 60 degree range data image of part.

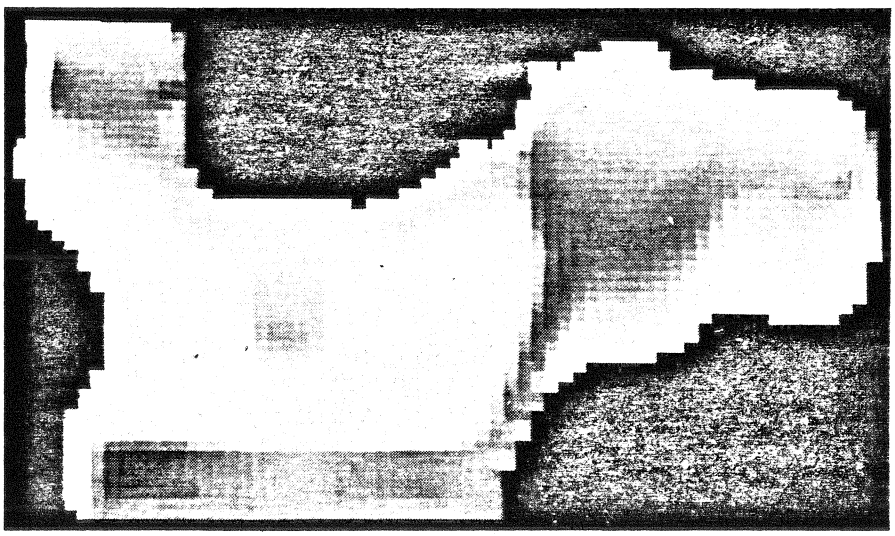


Fig. 15. 180 degree range data image of part.

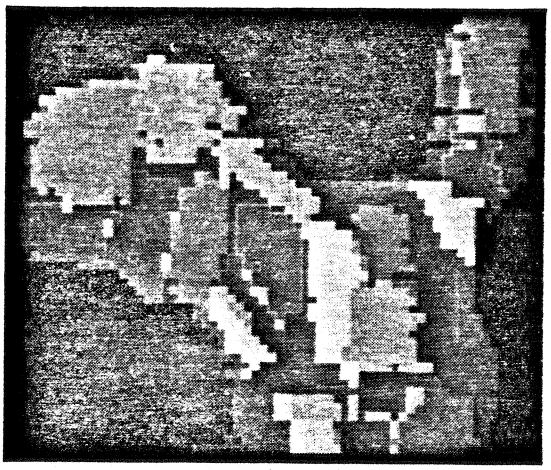


Fig. 16. Faces found for the industrial part.

3-D space has been developed and evaluated. This method has been tested on synthetic and industrial objects.

REFERENCES

[1] G. Agin, "Representation and description of curved objects," Stanford Univ., Memo. AIM 173, Oct. 1972.

[2] N. Ahuja, "Dot pattern processing using Voronoi neighborhoods," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. PAMI-4, pp. 336-343, May 1982.

[3] N. Ahuja, R. T. Chien, R. Yen, and N. Bridwell, "Interference detection and collision avoidance among three dimensional objects," in *Proc. 1st Annu. Nat. Conf. Artificial Intell.*, Aug. 1980, pp. 44-48.

[4] R. Bajcsy, "Three dimensional scene analysis," in *Proc. 5th ICPR*, Miami Beach, FL, Dec. 1980, pp. 1064-1074.

[5] H. H. Baker, "Three dimensional modelling," in *Proc. 5th Int. Joint Conf. Artificial Intell.*, Aug. 1977, pp. 649-655.

[6] D. Ballard, "Generalizing the Hough transform to detect arbitrary shapes," Dep. Comput. Sci., Univ. Rochester, Tech. Rep. TR-55, Oct. 1977.

[7] J. Bentley, "Multidimensional divide-and-conquer," *Commun. Ass. Comput. Mach.*, vol 23, pp. 214-229, Apr. 1980.

[8] J. Bentley and M. Shamos, "Divide and conquer in multidimensional space," in *Proc. ACM Symp. Theory of Comput.*, May 1976, pp. 220-230.

[9] B. Bhanu, "Shape and image segmentation using stochastic labeling," USC-IPI, Tech. Rep. 1030, Sept. 1981.

[10] T. O. Binford, "Survey of model-based image analysis systems," *Robotics Res.*, vol. 1, pp. 18-64, 1982.

[11] H. Blum, "A transformation for extracting new descriptors of shape," in *Models for the Perception of Speech and Visual Form*, W. Wathen-Dunn, Ed. Cambridge, MA: MIT Press, 1967.

[12] J. D. Boissonnat and O. Faugeras, "Triangulation of 3-D objects," in *Proc. 7th IJCAI*, Vancouver, 1981, pp. 658-660.

[13] R. Bolles and M. Fischler, "A RANSAC-based approach to model fitting and its application to finding cylinders in range data," in *Proc. 7th IJCAI*, Aug. 1981, pp. 637-643.

[14] J. W. Boyse, "Interference detection among solids and surfaces," *Commun. Ass. Comput. Mach.*, vol. 22, pp. 3-9, Jan. 1979.

[15] R. A. Brooks, R. Grenier, and T. O. Binford, "Progress report on a model-based vision system," in *Proc. ARPA Image Understanding Workshop*, Carnegie-Mellon Univ., Nov. 1978, pp. 45-151.

[16] C. M. Brown, "Some issues and answers in geometric modeling," in *Workshop on the Representation of Three-Dimensional Objects*, R. Bajcsy, Ed., 1979, pp. F-1-F-35.

[17] P. G. Comba, "A procedure for detecting intersections of three-dimensional objects," *J. Ass. Comput. Mach.*, vol. 15, pp. 354-366.

[18] E. Diday, G. Govaert, Y. Lechevallier, and J. Sidi, "Clustering in pattern recognition," in *Digital Image Processing*, J. C. Simon and R. M. Haralick, Eds. Dordrecht, Holland: Reidel, Oct. 1981, pp. 331-370.

[19] L. Dreschler, "Zur Reproduzierbarkeit von Markanten Bildpunkten bei der Auswertung von Realwelt-Bildfolgen," in *Modelle und Strukturen*, B. Radig, Ed. Berlin: Springer-Verlag, Oct. 1981, pp. 76-82.

[20] R. O. Duda and P. E. Hart, *Pattern Classification and Scene Analysis.* New York: Wiley-Interscience, 1973.

[21] 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.

[22] O. D. Faugeras, F. Germain, G. Kryze, J. D. Boissonnat, M. Hebert, and J. Ponce, "Toward a flexible vision system," in *Proc. 12th Int. Symp. Industrial Robotics*, June 1982, pp. 67-78.

[23] J. H. Friedman, J. L. Bentley, and R. A. Finkel, "An algorithm for finding best matches in logarithmic expected time," *ACM Trans. Math. Software*, vol. 3, pp. 209-226.

[24] E. L. Grimson, *From Images to Surfaces.* Cambridge, MA: MIT Press, 1981.

[25] E. L. Hall, J. B. Tio, C. A. McPherson, and F. A. Sadjadi, "Measuring curved surfaces for robot vision," *IEEE Computer*, vol. 15, pp. 42-54, Dec. 1982.

[26] T. C. Henderson, "An efficient segmentation method for range data," in *SPIE Conf. Robot Vision*, Arlington, VA, May 1982, pp. 46-47.

[27] T. C. Henderson and B. Bhanu, "Three-point seed method for the extraction of planar faces from range date," in *Proc. IEEE Workshop on Industrial Applications of Machine Vision*, Research Triangle Park, NC, May 1982, pp. 181-186.

[28] T. C. Henderson and N. Keskes, "L'analyse des objects tridimensionnels," in *Proc. 3rd Conf. Pattern Recognition and Artificial Intell.*, Nancy, France, Sept. 1981, pp. 277-287.

[29] T. C. Henderson and A. Mitiche, "Modeling 3-D structure," in *Modelle und Strukturen*, B. Radig, Ed. Berlin: Springer-Verlag, Oct. 1981.

[30] B. K. P. Horn, "Obtaining shape from shading information," in *The Psychology of Computer Vision*, P. Winston, Ed. New York: McGraw-Hill, 1970, pp. 115-155.

[31] W. Inokuchi and R. Nevatia, "Boundary detection in range pictures," in *Proc. 5th ICPR*, Miami Beach, FL, Dec. 1980, pp. 1301-1303.

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

[33] C. L. Jackins and S. L. Tanimoto, "Oct-trees and their use in representing 3D objects," *Comput. Graphics Image Processing*, vol. 14, pp. 249-270, 1980.

[34] T. Lozano-Perez, and M. A. Wesley, "An algorithm for planning collision-free paths among polyhedral obstacles," *Commun. Ass. Comput. Mach.*, vol. 22, pp. 560-570, Oct. 1979.

[35] D. Marr, "Representing visual information," MIT, AI Lab. Memo. 415, 1977.

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

[37] R. Nevatia and T. O. Binford, "Structured descriptions of complex objects," in *Proc. 3rd IJCAI*, Stanford, CA, 1973.

[38] J. O'Rourke, "Polyhedra of minimal area as 3-D object models," in *Proc. 7th JCAI*, Vancouver, Aug. 1981, pp. 664-666.

[39] M. Oshima and Y. Shirai, "Object recognition using 3-D information," in *Proc. 7th IJCAI*, Vancouver, Aug., 1981, pp. 601-606.

[40] T. Pavilidis, *Structural Pattern Recognition.* New York: Springer-Verlag, 1977.

[41] R. J. Popplestone, C. M. Brown, A. P. Ambler, and G. F. Crawford, "Forming models of plane and cylinder faced bodies," in *Proc. 4th IJCAI*, 1975, pp. 664-668.

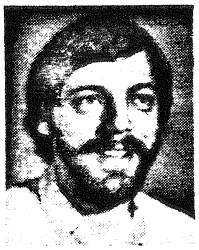
[42] F. Preparata and S. Hong, "Convex hulls of finite sets of points in two and three dimensions, *Commun. Ass. Comput. Mach.*, vol. 20, pp. 87-93, 1977.

[43] A. Requicha, "Representations for rigid solids: Theory, methods, and systems, *Comput. Surveys*, vol. 12, pp. 437-464, Dec. 1980.

[44] C. A. Rosen, "Machine vision and robotics: Industrial requirements," SRI Tech. Note 174, pp. 12-25, Nov. 1978.

[45] M. Shamos, "Computational geometry," Ph.D. dissertation, Yale Univ., 1978.

[46] R. Shapira, "A technique for the reconstruction of a straight-edge, wire-frame object from two or more central projections," *Comput. Graphics Image Processing*, vol. 3, pp. 318-326, 1974.
[47] R. Shapira and H. Freeman, "Computer description of bodies bounded by quadric surfaces from a set of imperfect projections," *IEEE Trans. Comput.*, vol. C-27, pp. 841-854, Sept. 1978.
[48] Y. Shirai, and M. Suwa, "Recognition of polyhedra with a range finder," in *Proc. 2nd IJCAI*, 1971, pp. 80-87.
[49] S. N. Srihari, "Representation of three-dimensional digital images," *ACM Comput. Surveys*, vol. 13, pp. 399-424, Dec. 1981.
[50] J. Tennenbaum, M. Fischler, and H. Wolf. "A scene-analysis approach to remote sensing," SRI Tech. Rep. 173, Oct. 1978.
[51] G. T. Toussaint, "The relative neighborhood graph of a finite planar set," *Pattern Recognition*, vol. 12, 261-268, Aug. 1980.
[52] S. Udupa, "Collision detection and avoidance in computer controlled manipulators," in *Proc. 5th Int. Joint Conf. Artificial Intell.*, Aug. 1977, pp. 737-748.
[53] S. A. Underwood and C. L. Coates, "Visual learning from multiple views," *IEEE Trans. Comput.*, vol. C-24, pp. 651-661, June 1975.
[54] L. T. Watson and L. G. Shapiro, "Identification of space curves from two-dimensional perspective views," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. PAMI-4, pp. 469-475, Sept. 1982.
[55] C. T. Zahn, "Graph-theoretical methods for detecting and describing Gestalt clusters," *IEEE Trans. Comput.*, vol. C-20, pp. 68-86, 1971.
[56] S. Zucher and R. Hummel, "An optimal three-dimensional edge operator," in *Proc. IEEE Conf. Pattern Recognition and Image Processing*, Aug. 1980, pp. 162-168.



Thomas C. Henderson received the Ph.D. degree in computer science from the University of Texas, Austin, in 1979.

He worked for the Deutsche Forschungs- und Versuchsanstalt fuer Luft- und Raumfahrt (DFVLR), Oberpfaffenhofen, West Germany, until 1980 when he went for a one year visit at the Institut National de Recherche en Informatique et en Automatique (INRIA), Roquencourt, France. Since 1981 he has been an Assistant Professor in the Department of Computer Science at the University of Utah, Salt Lake City. Current interests include computer vision, robotics, artificial intelligence, and multisensor systems.

Special Correspondence

GAGESIGHT: A Computer Vision System for Automatic Inspection of Instrument Gauges

MICHAEL L. BAIRD

Abstract—GAGESIGHT is a computer vision system recently integrated into production which demonstrates new flexibility in production automation technology. The vision algorithm accommodates significant misregistration in the position and orientation of objects, and adjusts process control parameters accordingly. The GAGESIGHT system is easily reconfigured for inspecting new types of objects by executing a teach program, and responding to a few instructions at an interactive display console. GAGESIGHT, intially intended for the automatic inspection and assembly of pointers and other components to instrument gauges, is finding new applications as well since the approach employed makes few assumptions regarding the objects being viewed.

Manuscript received September 27, 1982; revised May 23, 1983. This work was performed at General Motors Research Laboratories, Warren, MI 48090.

The author was with General Motors Research Laboratories, Warren, MI 48090. He is now with the Artificial Intelligence Laboratory, Advanced Research and Development, Fairchild Camera and Instrument Corporation, 4001 Miranda Avenue, Palo Alto, CA 94304.

Index Terms—Automation, computer vision, machine vision, robotic vision, visual inspection.

I. Introduction

At the AC Spark Plug Division of General Motors Corporation, various instruments such as fuel gauges and speedometers are manually calibrated and inspected for proper setting. Most of these calibration and inspection tasks are candidates for automation with computer vision techniques. Research was required to find general methods for determining the position of a pointer relative to markings on a gauge where the gauge itself is only roughly fixtured. These vision methods must be general because over several hundred different styles of gauges are produced at AC Spark Plug every year. Fig. 1 illustrates a few of the many styles of fuel gauges and speedometers. A separate vision system for each gauge is thus not practical.

This report describes a system which has been constructed that shows both the technical and economic feasibility of automating this process. Other attempts to automate this process required a special system to be constructed for each gauge style, and worked on only a limited number of styles [1]. Nippondenso Company, Ltd. also has recently demonstrated a system similar to GAGESIGHT [2]. It appears that they have fully integrated the vision capability into production, at the expense of some flexibility in instrument design rules.

0162-8828/83/1100-0618$01.00