Active Model Matching in Range Images

by

Paul F. Hemler
Griff L. Bilbro
and
Wesley E. Snyder

Center for Communications and Signal Processing
Department of Electrical and Computer Engineering
North Carolina State University

November 1986

CCSP-TR-86/26

Active Model Matching in Range Images

Paul F. Hemler, member IEEE
Griff L. Bilbro, member IEEE
Wesley E. Snyder, senior member IEEE

Center For Communications and Signal Processing
North Carolina State University
Raleigh, North Carolina

Abstract

A general three dimensional image analysis environment is presented. The environment proved extremely flexible, and a three dimensional object recognition system was designed using this environment. The recognition system addresses all the issues associated with view independent object recognition. The particular concentration is of this paper is the development of an active model matching approach called a procedural model. The procedural model uses the surface geometry available in a range image to limit the amount of searching that must be done. The surface geometry is determined by solving an eigensystem, and utilized to determine the objects pose by solving a second eigensystem.

1. Introduction

We have designed and built a flexible image analysis environment. The structure of our system is modular so that individual parts of the system can be independently researched. The system is limited in the domain of objects that can be recognized to those objects made up of quadric surfaces. The system will recognize images that are either synthesized, or actually determined range data. Once an image is input into the system, it is processed by a segmentation program that breaks the image into homogeneous areas called patches. An abstract representa-

tion of the segmented image called the scene graph is then formed. The scene graph is a knowledge interface between the higher level object recognition program and the lower level range data. The scene graph has both unary properties and binary relations and can be altered to reflect hypotheses about the image. Any changes made can be undone by backtracking.

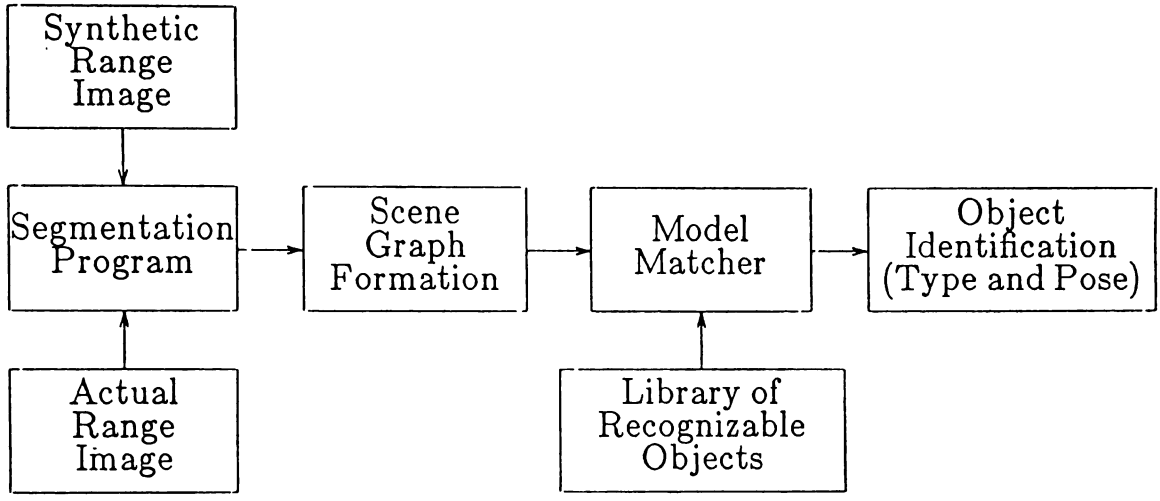
We have successfully developed a active model matching technique called a procedural model. Each procedural model is an expert at finding an instance of itself in the image. The procedural model uses any information either *a priori* or gained while the searching for the object. In particular, the procedural model uses the surface geometry of the object to guide its search. The procedural model is fast, robust, and easy to maintain.

The surface geometry of a patch is determined by fitting the data points in the patch to a quadric surface by solving an eigensystem. We classify the patches into certain recognizable quadric primitives based on a second eigensystem that determines the canonical form of the quadric coefficients. This eigensystem generates the pose of the primitive surface type. The procedural model pieces together the primitive surface types into objects. Object consistency is continually checked as the procedural model continues its search.

2. System Overview

The image analysis environment was designed as a tool to assist the research and development of new image analysis algorithms. The system is modular so the

individual sections of the system may be developed individually. A block diagram of the system is:



Block diagram of a three-dimensional object recognition system.

The synthetic range image generator[13] allows us to generate any object in the limited domain of objects we are working with, that is, objects made up quadric surfaces. The image generator also allows the object to be rotated and translated so it can be viewed from any direction. Figure 1 and 2 show two cylinders generated with the system. We have designed the system to generate noise free images (random noise), however, realistic range data may be fabricated by adding gaussian noise. Ultimately, the algorithms developed will be executed using real range data, so we have allowed for actual range images to be input to the system for analysis.

After the image is in the system, the first processing step is to segment the image. This step can be thought of as breaking the image up into "patches" where

each patch has a homogeneously calculated property over its surface. See[11] for details of the segmentation process. The segmentation program produces an image referred to as a label image where each patch is assigned a unique label number. The label image together with a graphical representation, and the original range image are used in the recognition phase of the system.

The model matching step must compare an input instance to a library of recognizable objects. Many methods of matching have been reported in the literature. These methods have ranged from a correlation measure to feature detection and feature matching. Using a correlation measure in a view independent object recognition system, requires all possible views of the object to be stored and then compared. These types of systems are obviously limited in the context of general view independent object recognition, however, they have proven useful in well defined environments. Feature detection and matching systems use a statistical approach to object recognition. The feature vector, which is a set of features of an image are determined. After the features are detected, a statistical match between the image feature vector and a library of feature vectors of the recognizable objects is done. The object in the library corresponding to the feature vector with the best statistical match to the image feature vector is regarded as the object. The output of a recognition system using this approach is the type object, and the probability that the match is correct.

Another technique for matching an image instance to a model library is referred to a a model based or active model matching approach. With this

technique, the image is searched for instances of one particular object.[7,2,14,15] This is the type of approach used in our system. Each model in the library of recognizable objects is a procedure that is an expert at recognizing one particular object. We call these models procedural models. A procedural model is written to utilize any information about an object whether it is known *a priori* or determined from the image being analyzed.

Any information known about the object can be used to limit the amount of searching required to determine an instance of that object. Range images preserve the surface geometry, which can later be determined and used during object recognition. The surface geometry of all the parts and the geometrical relationships between these parts is "hard-wired" into the procedural model. The procedural model then trys to correctly identifying the specified parts and establish the correct relations among the parts.

A successful object recognition system is built using several levels of abstraction. The different levels not only correspond to the processing being done, but also to the format of the data used in that level. The lower levels of the system use a low level form of the data such as the range image. The higher levels of the system use a more abstract representation of the data. The procedural model is one of the highest level in our system and consequentially uses the most abstract form of the data. This high level representation of the image data is referred to as the scene graph, and can be thought of as an ordered region adjacency graph. More details of the scene graph are given later in the next section.

3. Scene Graph

The scene graph is an abstract representation of the segmented image. It is a multiply traversed graph structure that consists of nodes having unary properties and binary relations. Each node in the scene graph corresponds to a patch in the segmented image. The unary properties of the nodes record information of the represented patch that are useful in identifying a correspondence between the patch and a model part. The unary properties consist of: label number, priority, bounding window, area, volume, etc. The binary relations between nodes consist of: adjacency, nearness, next lower priority. The adjacency relation is defined for all patches that are in direct contact with other patches. We use the an eight neighborhood around a pixel to determine which patches a particular patch is adjacent to. The nearness relation is defined for all patches that have intersecting extended bounding windows. A bounding window is simply the minimum and maximum X and Y coordinates of the patch, and the extended bounding window is obtained by enlarging the window in all directions by a specified amount δ. The nodes of the scene graph are ordered by the patches priority, and the next lower priority relation is defined as the patch with the next lower priority. To summarize, a scene graph is an ordered region adjacency graph where the ordering is done by the priority of the patch, and the priority is a function of the cardinality of the patch.

The task of object recognition is thought of as matching the scene graph to a model library. The model in the library which in some sense is the closest to the

scene graph is then taken as the interpretation of the image. Many papers have dealt with this problem[7,14,1,8,15,17,16] and some systems have addressed the problems associated with model matching with view independence. Our system is different from these other systems in that we don't match the scene graph to the model library directly. Instead, we allow a model to investigate the scene graph and attempt to understand it. We consider view independent object recognition as scene graph understanding.

It would be unrealistic to believe that a segmentation program will produce a perfect segmentation for all objects in the domain. A perfect segmentation is defined such that each patch in the image has a corresponding region in the model. In practice, perfect segmentation is not possible and a segmentation results in: 1) undersegmentation, meaning several regions in the model are merged into one patch in the image, 2) oversegmentation, meaning region in the model is broken into several patches in the image. These two definitions are correct if two assumptions are included. The first is that there is no occlusion and the second is that both types of segmentation do not occur in the same image. When designing the segmentation program,[11] we made it such that if an error did occur, it would tend to be an oversegmentation error. See figure 3 for a segmentation of the range image in figure 1. It is logically simplier to merge patches to form regions, instead of inferring one patch is really two, and splitting it.[3]

3.1. Patch Merging

As the object recognition task progresses, it may become necessary to merge two patches to form a region. The scene graph allows for this by: 1) forming a composite node, 2) establishing relations between the composite node and its constituent nodes. The composite node has the same structure as a patch node, but its unary properties are determined from its two constituent nodes. The binary relation of parent is established so that any data references to the constituent nodes access the data in the composite node. Two other relations of child1 and child2 are established to permit unmerging. Patch merging is also done with patches that are too small for other analysis programs to work with. That is, we use patch merging for small patch suppression. Figure 4 shows the label image after patch merging to suppress the small patches.

3.2. Point Merging

One other approach to the small patch suppression problem is that of point merging. This is a point by point region growing technique that uses the geometry of nearby surfaces to determine which surface the noise point, or small patch is a part of. One method we use is:

```
For all pixels that are either noise or a small patches
        Determine the quadric coefficient of all nearby surfaces.
        Calculate the expected Z value of the point given the X and Y
        coordinates and the quadric coefficient of a nearby surface.
        If the expected value is within the quantization noise
                Merge the point into the region.
        Else repeat calculation using another nearby patch.
```

This approach works well with synthetic range images where the only noise is the quantizing noise, but did not work as well when gaussian noise was added or when real range data was used. We have developed a point merging technique for actual range images using a maximum likelihood approach.

The maximum likelihood approach uses the X, Y, and Z values of the noise point to determine which nearby surface that point belongs to. In this approach, we use the rms fit error of the surface to determine the best patch to assign the noise pixel to.

Let

$$\hat{v} = \begin{bmatrix} x \\ y \\ z \end{bmatrix}$$

be the noise point. Then the probability that $\hat{v}$ is on surface S_i is:

$$P(S_i|\hat{v}) = Cp(\hat{v}|S_i)P(S_i)$$

where C is a normalization constant. We assume a Gaussian probability density function for the probability that a point v is on the surface S_i.

$$p(\hat{v}|S_i) = \frac{1}{\sqrt{2\pi}\sigma_i} e^{\frac{-1}{2}\left(\frac{f_i^2(\hat{v})}{\sigma_i^2}\right)}$$

Where

$$\sigma_i^2 = \frac{1}{N_i}\sum f_i^2(ubar)$$

defines the RMS error of the fit, where the sum is taken over all points, $\bar{u}$ in the *ith* segment. We note, if a point x, y, z is on the surface, then

$$f(x,y,z) = 0$$

The noise point is assigned to the nearby surface that has the highest probability of containing that point. This approach will assign a noise point to a nearby surface, and works well in the presence of noise. Figure 5 shows the range image after the small patch suppression operation, and the point merging technique described above.

4. Procedural Models

The procedural model is an active approach of matching an instance of an image to a model. The model is a procedure that is an expert at finding one specific object in the scene. For example, a procedural model might be defined to locate broken pipes in a scene. Lets call this procedural model *find_broken_pipe*. The model possess all *a priori* knowledge of broken pipes, and attempts to locate a broken pipe in the scene. If a broken pipe exists in the scene, the patches in the scene graph corresponding to the broken pipe are returned, otherwise NULL is returned. For example, *find_broken_pipe* is defined as:

```
find_broken_pipe (scene_graph)
   if (find_cyl1 (scene_graph) == SUCCESS)
      return (SUCCESS)
   else
      return (FAILURE)

find_cyl1 (scene_graph)
   if (cyl1 = get_next_cyl (scene_graph) == NULL)
      return (FAILURE)

   while (cyl1 != NULL)
```

```
    {
    if (find_cyl2 (scene_graph) == SUCCESS)
       return (SUCCESS)

    cyl1 = get_next_cyl (scene_graph)
    }

find_cyl2 (scene_graph)
   if (cyl2 = get_next_cyl (scene_graph) == NULL)
      return (FAILURE)
   else
      return (SUCCESS)
```

Example procedural model to find a broken pipe.

For clearity, *get_next_cyl* is not explicitly stated. Its purpose is to return the next lower priority cylindrical patch on the scene graph. The structure of the procedural model can be seen from this example. For a broken pipe to be in the image, two sections of pipe must be found. If only one section can be found, the pipe is not broken and FAILURE is returned. This implies the procedural model failed to find an instance of itself in the image.

A procedural model is ideal for an automated inspection system, since all parts and the relations between the parts are known. The inspection task simply is to verify: 1) the presence of all required parts, 2) part tolerance, 3) the relation between all the parts. A procedural model is also ideal for view independent object recognition using range images because the geometry of the surfaces assists in the search for the parts comprising the object.

We have identified several classes of procedural models. They consist[4] of 1) linear, 2) recursive, 3) iterative, 4) recursive/iterative. Some advantages of using

procedural models for either inspection or recognition are: 1) all have a similar program structure, 2) easy to maintain, 3) easy to add heuristics, 4) faster then a depth first search. The first three of these features are obvious from the structure of the procedural model. Furthermore, we can show that:

Theorem:

> The worst case time bound of a procedural model is equal to a depth first search on a graph with the number of nodes in the graph equal to the number of parts the model.

5. Quadric Fitting

We have limited the domain of objects we wish to recognize/inspect to those objects made up of only quadric surfaces. About 85% of all manufactured parts can be described in this way.[5] Such objects are therefore, of great interest in a industrial environment.

A general quadric surface can be described by:

$$F(x,y,z)=ax^2+by^2+cz^2+2fyz+2gzx+2hxy+2px+2qy+2rz+d=0 \quad (1)$$

The set of coefficients that minimizes the difference between the surface generated by those coefficients and the actual data is determined in the minimum mean square sense. In another paper[10] this is shown to be equivalent to solving an eigenvalue problem. In fact, they find the minimum value of λ for which

$$R\alpha=\lambda K\alpha$$

where R is the scatter matrix for the data, K is a constraint matrix, α is the vector (of quadric coefficients), associated with the minimum eigenvalue λ.

The result of this analysis provides both the coefficients, a, b, ... of equation 1 and a measurement of the RMS error in the fit. In particular, the ϵ mentioned in a previous section may be shown to satisfy:

$$\frac{1}{N} \Sigma \epsilon^2 = \frac{1}{N} \lambda_{\min}$$

where $\lambda_{\min}$ results from the eigensystem of the fit.

6. Quadric Analysis

This section is concerned with determining the object represented by the quadric coefficients found in a previous section. We will define a "quadric section" to be a bounded portion of a quadric surface such that it satisfies a finite number of quadric inequalities. In addition, we define a "quadric polyhedron" as a closed solid, bounded by a finite number of quadric sections. The problem domain of primary interest to us is a subset of quadric polyhedra: We will mostly restrict our attention to quadric polyhedra composed of four types of quadric sections with sharp joins. The four types of sections comprise our primitive set: planes, cones, cylinders, and spheres. Of course the additional "nonanalytic" section type is treated so that system robustness is not compromised unduly by these restrictions. A sharp join[6] is a "roof edge" boundary between two quadric sections across which the object surface is continuous, but the surface normal is discontinuous. A "step edge" is an occlusion boundary across which the image value (range, depth, or in our case, object surface altitude) is discontinuous. We restrict ourselves to quadric polyhedra with sharp joins because they are common in industrial applications and

because segmentation [11] into connected components can be successfully based on invariant local surface properties (curvature). Smoother joins can result in erroneous segmentations in which two regions are merged into a single segment. Such a segment will typically be misclassified as nonanalytic. This is tolerated by our system but with a degradation of performance.

6.1. Canonical Form

A crucial feature of our recognition system is that it exploits information defined by the quadric fit to a segment to hypothesize the primitive type of the quadric as well as partially determine the pose of the segment. This is possible because a "canonical" or standard form for a quadric can be associated with any collection of quadric coefficients. The canonical coefficients determine the primitive type of the quadric. A rigid motion (which is not unique for the cylindrically symmetric primitives in our problem domain) relates the original quadric coefficients to the canonical coefficients. The motion defined by a segment can be used in a model based vision system to define (at least partially) the pose of the entire object.

A motion of a quadric is taken to be a rotation followed by a translation. For a plane, we define the canonical coefficients as all zero and we define its canonical position as in the xy plane. We define the associated motion as any that carries the original plane into the xy plane. For the other primitives, we rewrite the quadric equation in matrix form,[18]

$$F = \mathbf{x}^T A \mathbf{x} + 2B^T \mathbf{x} + C \ ,$$

where the superscript T indicates matrix transposition, $\mathbf{x} = (x, y, z)^T$ is a point on the surface, A is a real symmetric matrix of second order coefficients, B is a real vector of first order coefficients, and C is a real constant. A rotation is determined from the eigensystem

$$A = U \Lambda U^T \ ,$$

where U is the matrix of eigenvectors and Λ is the diagonal matrix of eigenvalues. After this rotation, it is straightforward to determine a translation which causes the linear terms to vanish. The imposition of Faugeras' condition on the eigenvalues normalizes the canonical coefficients but does not fix the overall sign of F. We now choose that sign so that the canonical form of the quadric surface can be written

$$ax^2 + by^2 + cz^2 = d \ ,$$

with a positive d, at least for our set of primitives. These four constants[12] are the desired canonical coefficients.

6.2. Quadric Classification

In our system, recognition depends on both local (bottom up) actions and global (top down) actions, resulting in both tactical and strategic components of image analysis. The tactical component depends on the section poses defined above as well as on several other locally determined properties of quadric sections. We have found that the sense of the segment (whether convex or concave) allows a

procedural model to distinguish the inside from the outside of hollowed or cup-shaped objects. The surface sense is determined from the sign of the mean curvature averaged over the segment.

In addition to this extrinsic property, an analysis routine in our system provides the procedural model with a "fuzzy" or coarse classification of the intrinsic surface primitive type and a single real parameter elaboration of that type. For cones, the parameter is the half angle at the apex of the cone, which is typically not in the image but instead is determined from the canonical coefficients. For cylinders and spheres, the parameter is the radius, which is also calculated from the canonical coefficients. This type-and-one-parameter characterization has also been recently used by Bolle and Cooper[5] to classify segments of range images. Planes are simply characterized as planes; this is done at fit time.

The analysis is restricted to cones, cylinders, and spheres. Any surface that does not fall into one of these classes is classified as a nonanalytic surface and is not analyzed further in the current system.

We will use four "fuzzy" or coarse predicates, *ISPOS*, *ISZER*, *ISCIR*, and *ISSPH* to help classify this canonical quadric. The first two involve thresholds that characterize the quality of our data and the numerical precision of our analysis. These thresholds are arbitrary, but not vague: they are dimensionless quantities that characterize fractional errors in our system and are typically one or two percent in the boolean expressions

$$ISPOS(x) = (x > .01) ,$$

and

$$ISZER(x) = (|x| < .02) .$$

We distinguish circular cylinders and cones from elliptical cylinders and cones (currently classified and tolerated as nonanalytic) with the Boolean expression

$$ISCIR(x, y) = ISZER(x - \frac{1}{\sqrt{2}}) \text{ and } ISZER(y - \frac{1}{\sqrt{2}}) ,$$

where and is the usual Boolean conjunction and the arguments are assumed to be second order quadric coefficients normalized by the Faugeras' condition.[9] The final fuzzy predicate is used to distinguish spheres from more general ellipsoids

$$ISSPH(x,y,z) = ISZER(x - \frac{1}{\sqrt{3}}) \text{ and } ISZER(y - \frac{1}{\sqrt{3}}) \text{and} ISZER(z - \frac{1}{\sqrt{3}}) ,$$

The final accuracy of our system is not limited by these thresholds, since any measurement made at this stage of the recognition procedure can be refined globally after the identity of the object has been determined and the correspondences between segments and primitives has been completed.

A section is classified as a cone if the constant d satisfies *ISZER*, none of the second order coefficients are zero, two of the second order coefficients satisfy *ISPOS*, and those two satisfy *ISCIR*. In this case, the descriptive parameter of the segment is the half angle. It is calculated from the canonical coefficients as the arc tangent of $-c/a$.

A section is classified as a cylinder if the constant is positive, one second order coefficient is zero, and two are positive, and those two satisfy *ISCIR*. In this case,

the descriptive parameter is the radius, which is estimated as

$$\sqrt{d\sqrt{2}},$$

for a cylinder under the Faugeras condition.

A section is classified as a sphere if the constant is positive, all three second order coefficients are positive, and they satisfy *ISSPH*. The descriptive parameter is the radius

$$\sqrt{d\sqrt{3}},$$

for a sphere under the Faugeras condition.

Any section (not previous classified as a plane) that does not fall into one of these classes is classified as a nonanalytic surface.

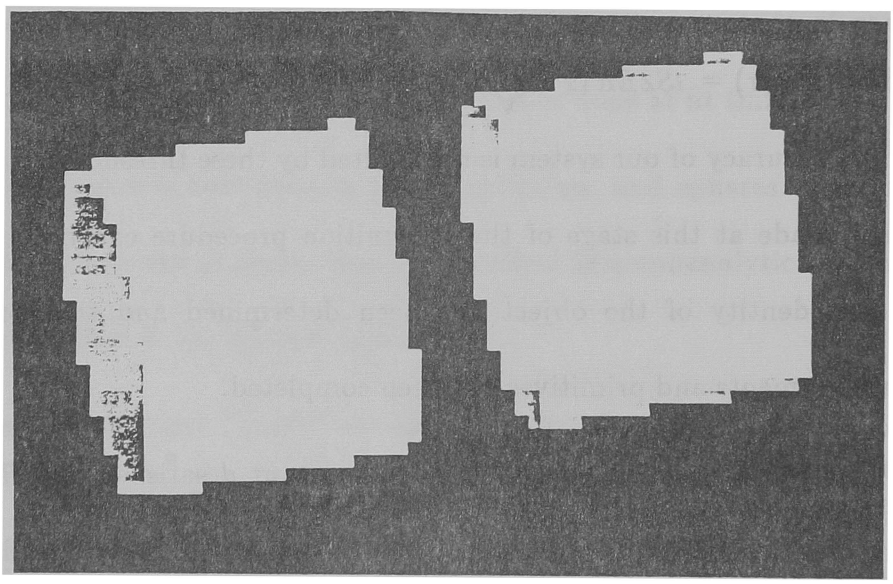
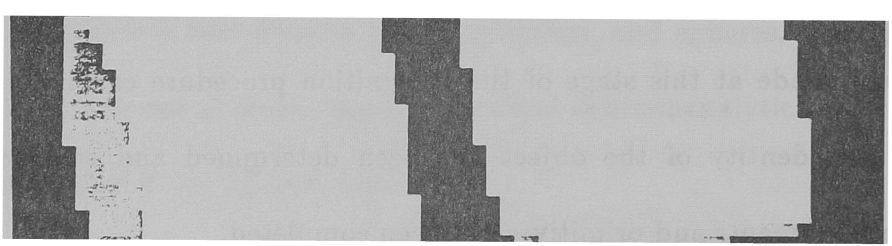


Figure 1. Synthetic range image of two cylinders.



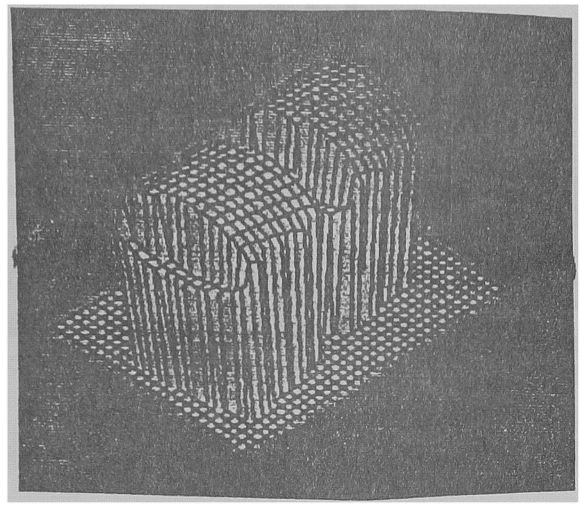


Figure 2. Synthetic range image of two cylinders, 3-D view.

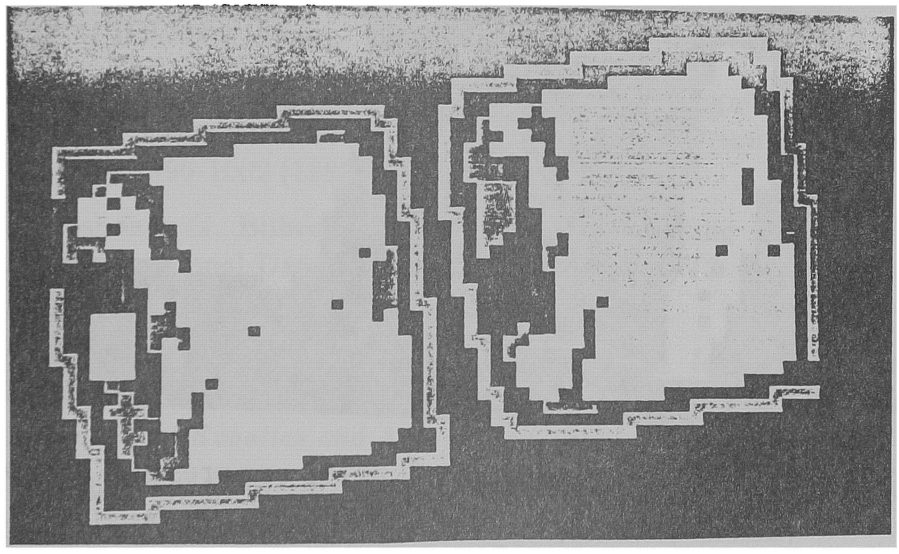
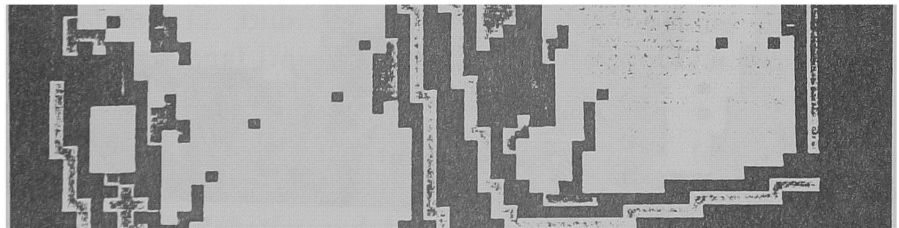


Figure 3. Results of segmentation, the label image.



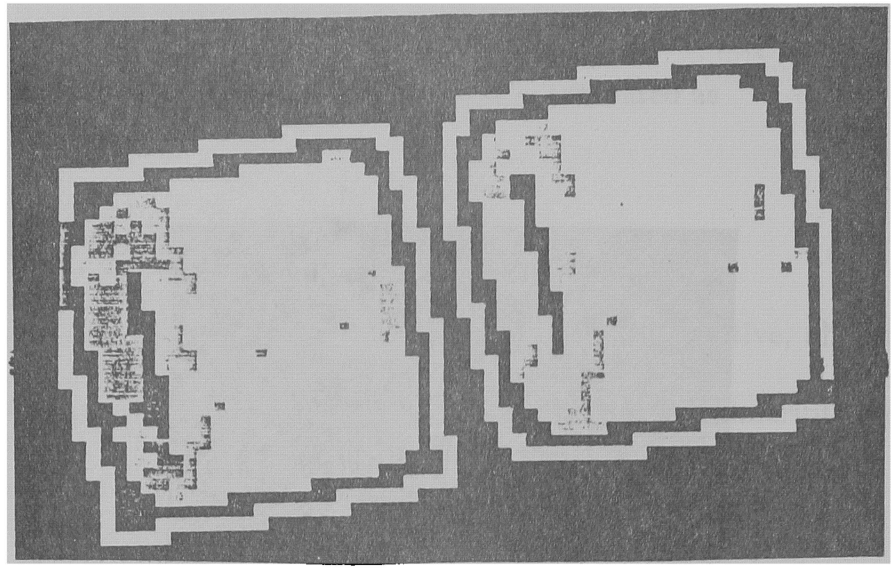


Figure 4. Label image after small patch suppression.

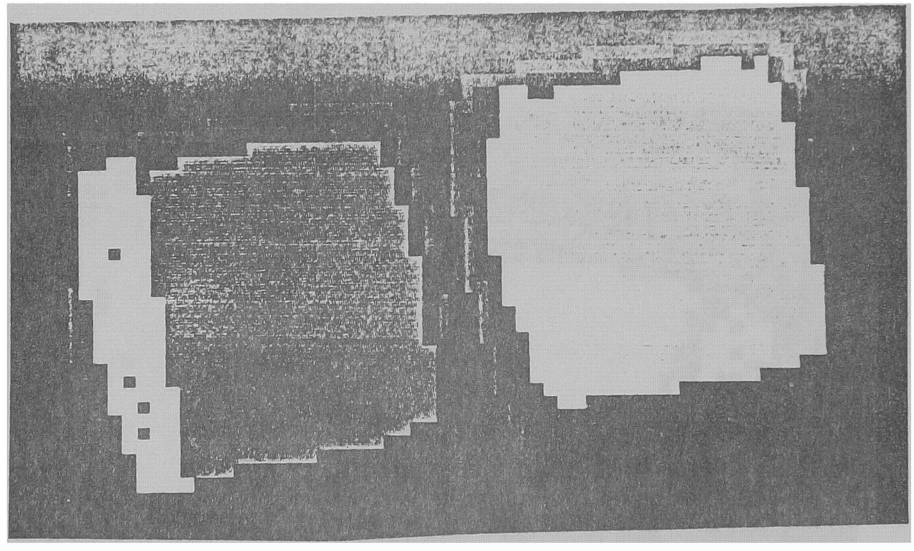
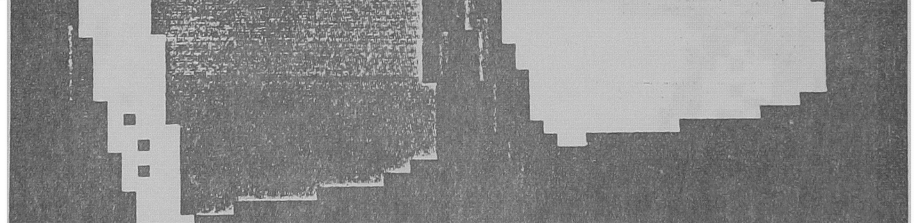


Figure 5. Label image after small patch suppression and point merging.

1. W.A. Perkins , " "A Model-Based Vision System for Industrial Parts"," *IEEE Transactions on Computers* **C-27**(2)(February 1978).

2. P. Besl and R. Jain, "An Overview of Three-Dimensional Object Recognition," RSD-TR-19-84, University of Michigan, College of engineering, Center for Research on Integrated Manufacturing Robot Systems Division



3. G.L. Bilbro, C.D. Savage, and W.E. Snyder, "Identifying Region Adjacency Graphs Degraded by Feature Blending," *CCSP-WP-85/12*, Center for Communications and Signal Processing, NCSU, (July 1985.).

4. G. L. Bilbro, W. E. Snyder, and P. F. Hemler, "Recognition of Objects in Range Images," *SME World Conference on Robotics Research*, (August, 1986).

5. R.M. Bolle and D.B. Cooper, "On Optimally Combining Pieces of Information, with Application to Estimating 3-D Complex-Object Position from Range Data," *IEEE PAMI* **PAMI-8**(September, 1986).

6. M. Brady, J. Ponce, A. Yuille, and H. Asada, *Describing Surfaces,* MIT Press, Proceedings of the Second International Symposium on Robotics Research (1985).

7. R. A. Brooks, "Symbolic Reasoning Among 3-D Models And 2-D Images," *Artificial Intell.* **17** pp. 285-348 (1981).

8. C. Dane and R. Bajcsy, *An Object-Centered Three-Dimensional Model Builder.* October, 1982.

9. O. D. Faugeras and M. Hebert, "A 3-D Recognition and Positioning Algorithm Using Geometrical Matching Between Primitive Surfaces," *Proc. IJCAI,* pp. 996-1002 (1983).

10. B. Groshong, G. Bilbro, and W.E. Snyder, "Fitting a Quadric Surface to Three Dimensional Data," *NCSU CCSP-TR-85/17, also submitted to IEEE PAMI,* (1986).

11. B. R. Groshong, "Range Image Segmentation for Object Recognition," *submitted for publication in the proceedings of the 1987 International Conference on Robotics and Automation,* (September 1986).

12. P. Hemler, G. L. Bilbro, and J. Franke, "Use of Quadric Coefficients to Classify Surface Types," *CCSP - WP - ??,* (in Preparation).

13. M. Lanzo, "Qsyn: Instructions and Operation," *Technical Report 86/10/1 (in preparatation),* Communication Unlimited, (October, 1986).

14. P.Rummel and W. Beutel, "Workpiece Recognition And Inspection By A Model-Based Scene Analysis System," *Pattern Recognition* **17**(1) pp. 141-148 (1984).

15. W. A. Perkins, "Model-Based Inspection System for Component Boards," *Pattern Recognition* **17**(1) pp. 135-140 (1984).

16. L. G. Shapiro and R. M. Haralick, "Organization of Relational Models for Scene Analysis," *IEEE Trans. on PAMI* **PAMI-4**(6) p. 595 (November, 1982).