Proc. SPIE Advances in Intelligent Robotics Systems:
Intelligent Robots and Computer Vision, Cambridge, MA., October 1986, 292-298.

Merging 3-D Symbolic Descriptions Obtained from Multiple Views of a Scene

Martin Herman

Robot Systems Division
National Bureau of Standards
Gaithersburg, MD 20899

ABSTRACT

An algorithm is presented for merging three-dimensional symbolic descriptions of static polyhedral scenes obtained from multiple viewpoints. The 3-D descriptions are assumed to be partial because of occlusions. The merging algorithm treats the topology of the descriptions (i.e., connections between vertices, edges, and faces) separately from the geometry (i.e., physical dimensions and relative locations). This paper describes one part of a larger project whose goal is to obtain an integrated symbolic description of a scene from range data obtained from multiple viewpoints.

1. Introduction

An important problem for robotics vision is that of generating a 3-D description of an unknown scene from range maps obtained from multiple views. This paper describes part of a project whose goal is to obtain a full symbolic description of a polyhedral scene from range data obtained from multiple viewpoints. Each view is processed in sequence, and the 3-D information extracted from each view is used to incrementally construct a model of the scene. Three main steps are followed in this project. First, a scene description, in terms of faces, edges, and vertices, is generated from each view. This step is described in [5]. Next, descriptions from separate views are matched to obtain corresponding elements and to obtain the six parameters of the global coordinate transformation. This process is described in [6]. Finally, the separate descriptions are merged, resulting in a more complete overall description of the scene. This step is the topic of the current paper.

The approach of the overall project is useful for two applications: (1) a mobile robot or sensor incrementally learning its environment as it moves through it [3, 7, 8, 9, 10, 11], and (2) incrementally learning the 3-D shape of an object presented via multiple views to a sensor [2, 12, 13].

2. Approach

Each new view of the scene undergoes analysis which results in a symbolic scene description. This description is then merged with the current model. The description extracted from the first view forms the initial state of the scene model. This model is incrementally updated with information from each new view.

Both the symbolic descriptions extracted from each viewpoint, as well as the scene model, are represented in the form of a graph. Since this graph represents 3-D structure in the scene, we call it a *structure graph* [7]. The nodes and links represent topological and geometric constraints. The structure graph representation models surfaces in the scene as polyhedra. The components of a polyhedral surface are the face, edge, and vertex. We distinguish the topology of the polyhedral components from their geometry [1, 4]. The geometry involves the physical dimensions and location in 3-space of each component, while the topology involves connections between the components.

Each node in the structure graph represents a face, edge, or vertex. Associated with each face node is its plane equation, with each edge node its line equation, and with each vertex node its coordinate values. The links in the structure graph capture the topology between nodes. A vertex node has a link to each edge and face node on which it lies, an edge node has links to its vertex nodes and to the face nodes on which it lies, and a face node has links to its vertex and edge nodes.

A vertex must lie on at least two edges. If it lies on only one edge, then it is not really a vertex, and is tagged as a *non-vertex end point* of the edge. This comes about when a portion of the edge has been occluded, and the edge is therefore tagged as *incomplete* (see Fig. 1). Any face that contains incomplete edges, or that is not closed by a ring of edges, is also tagged as incomplete.

3. Merging

Given a structure graph description extracted from a new view, it is incorporated into the current model structure graph as follows. First, the two descriptions are matched as described in [6]. The output of this matching process is a set of corresponding pairs of elements, i.e., vertices, edges, and faces. Those elements in one description that have no match in the other description because of occlusion are matched with NIL. Non-vertex end points are not included in the corresponding pairs of elements, since the matching process does not match such points.

In order to merge the two descriptions, we require not only the corresponding elements, but also the global coordinate transformation that maps one description into the other. To obtain the coordinate transformation, we use an algorithm that obtains the best rigid transformation (rotation and translation) between two sets of corresponding points in 3-space. Suppose a point $\bar{v}_{1_i}$ in description 1 is transformed into the point $\bar{v}_{2_i}$ in description 2 by the equation:

$$\bar{v}_{2_i} = R\bar{v}_{1_i} + T,$$

where R is a rotation matrix and T a translation vector. Then R and T may be estimated by minimizing the sum of the square norms:

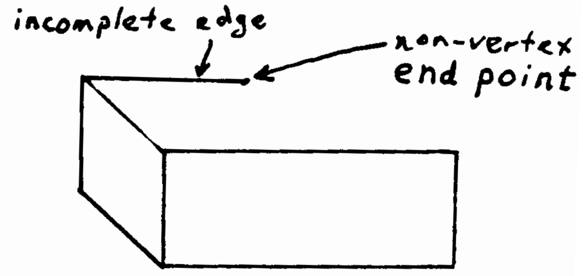


Fig. 1

$$||R\bar{v}_{1_i}+T-\bar{v}_{2_i}||^2.$$

The vertex matches returned by the matching module are used to obtain this transformation.

At this point, both descriptions are expressed in the same coordinate system. To merge the two descriptions, pairs of corresponding elements are first merged, then elements for which no correspondences were found are included.

Basically, the merging process consists of two main steps. First, the topology of the two descriptions are merged, then the geometry is merged. Topological merging involves generating new elements (vertices, edges, and faces) for pairs of corresponding ones, and then generating appropriate links among these new elements and between these elements and those that have no correspondences. Geometric merging involves determining the appropriate geometry (i.e., coordinates of vertices, line equations of edges, and plane equations of faces) of all elements modified by the topological merging.

4. Merging Edges and Faces Within a Description

Before topological and geometric merging between elements in the two descriptions can occur, we want to merge multiple elements in each description that correspond to a single element in the other description. For example, in Fig. 2, the face f_1 will match both faces f_2 and f_3, while the edge e_1 will match both edges e_2 and e_3. The final goal of the merging process is to merge f_1 with f_2 and f_3, and e_1 with e_2 and e_3. By first merging elements within each description, e.g., f_2 with f_3, and e_2 with e_3, the later process of merging elements across the two descriptions can be more uniform, since it will only have to merge pairs of, rather than multiple, corersponding elements.

Notice that our approach for merging elements within a description is different from the typical approach. Typically, edges e_2 and e_3 in Fig. 2 would be merged using heuristics such as that they are colinear and have opposing T junctions (i.e., junctions A and B). Such heuristics often will lead to errors, since we can only guess at what happens in occluded regions. A more reliable way to determine what is occurring in occluded regions is to move to another viewpoint where the occluded region becomes visible. This is in effect what is occurring here. After the totally visible edge e_1 has been determined to match with the partially complete edges e_2 and e_3, we know that e_2 and e_3 are portions of the same edge.

Merging of edges and faces within a description occurs as follows. If an edge in one description matches multiple edges in the other (which will be multiple incomplete edges), the multiple edges are merged into a single edge by connecting the two vertex end points (e.g., C and D in Fig. 2) of these edges. This new edge, which is complete, then replaces the multiple ones. Topological pointers are then modified, so that the new edge is given pointers to all the vertices and faces of the old edges (except for the non-vertex end points, e.g., A and B in Fig. 2, which are deleted), and all the vertices and faces that had pointers to the old edges are given pointers to the new edge instead.

The merging of faces within a description occurs after the edges have been merged. If a face in one description matches multiple faces in the other (which will be multiple incomplete faces), the multiple faces are merged into a single face. This new face then replaces the multiple ones. It is tagged as incomplete if it has at least one incomplete edge. Topological pointers are then modified, so that the new face is given pointers to all the vertices and edges of the old faces, and all the vertices and edges that had pointers to the old faces are given pointers to the new face instead.

5. Dealing with Non-Vertex End Points

At this point, we want to find vertices in one description that "correspond" to non-vertex end points in the other description (e.g., J and L in Fig. 3), and "corresponding" non-vertex end points in the two descriptions (e.g., B and D in Fig. 4). Such points do not really correspond because the point on the edge which divides the visible from the occluded portions is in general different for different viewpoints. However, it is convenient to treat the points as corresponding pairs, since they must somehow be combined to produce a single point in the merged description.

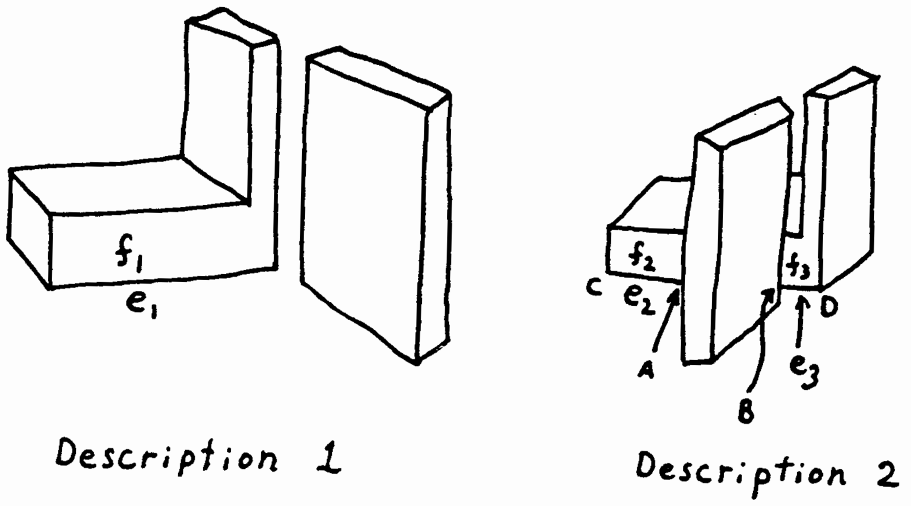
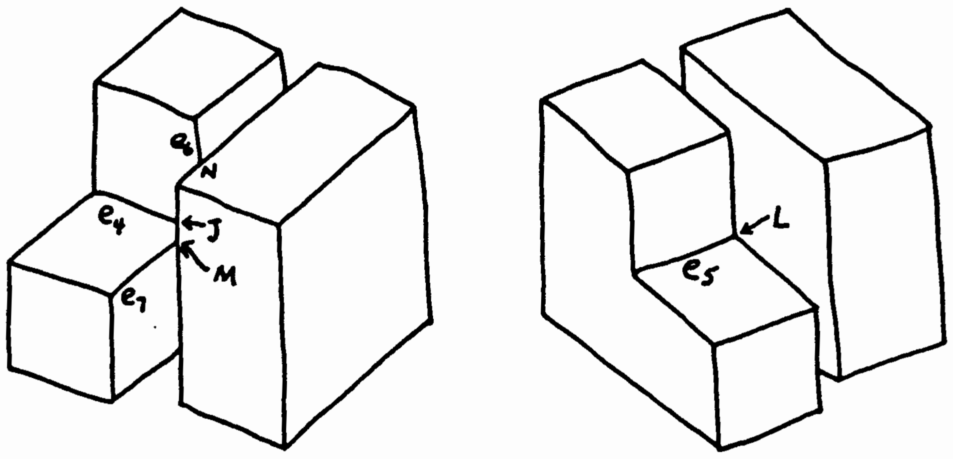
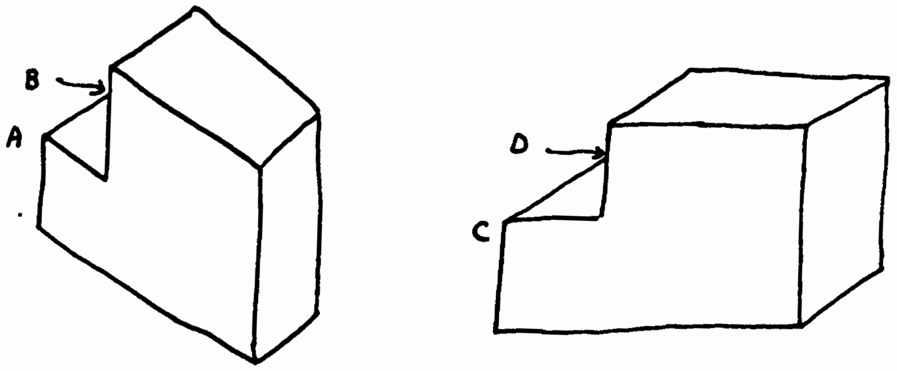


Fig. 2



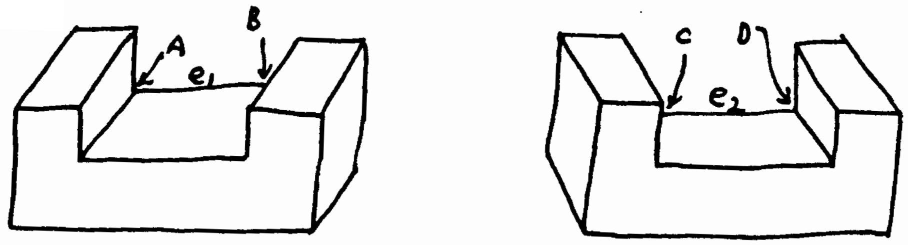
Description 1 Description 2

Fig. 3



Description 1 Description 2

Fig. 4



Description 1 Description 2

Fig. 5

Notice that in both these cases, the edges on which these points lie have been found to match by the matching algorithm. Also, in both cases, the matching edges have one pair of matching vertices. Therefore, to find corresponding pairs of points involving non-vertex end points, each matching pair of edges is examined. If at least one of the edges is incomplete, then one pair of end points will not appear in the list of vertex matches obtained by the matching algorithm. This pair is then included in the list. The non-vertex end points are tagged as such so that when merging occurs later, these points will be merged differently from those returned by the matching algorithm.

The current algorithm for finding correspondences for non-vertex end points does not handle the situation depicted in Fig. 5, where it appears that A corresponds to C and B to D. This situation is not handled because it is not known at this point in the processing that the edges e_1 and e_2 should match. Our matching algorithm matches vertices first, and then finds edge and face matches based on the vertex matches [6]. Since e_1 and e_2 have no matching vertices, they themselves are not found to match. (In fact each of them will be matched with NIL.)

Notice that it is possible for a vertex point in one description to match with multiple non-vertex end points in the other description. In Fig. 3, for example, the vertex point L in description 2 corresponds to the non-vertex end points J, M, and N in description 1.

As described earlier for cases involving multiple edges or faces in one description that match single edges or faces in the other, we want to merge multiple end points in one description that match a single vertex in the other. By doing this first, the later process of merging elements across two descriptions can be simpler and more uniform.

The procedure we use to merge multiple non-vertex end points is to intersect all the edges on which these points lie. In Fig. 3, for example, the edges e_4, e_6 and e_7 are intersected (in 3-space) in pairs, resulting in three intersection points. The average position of the intersection points is used to form a new vertex, which is given topological pointers to all the edges and faces of the old end points. All edges and faces with pointers to the old end points are then given pointers to the new vertex. All edges on which the old end points were lying are then tagged as complete, and all faces that contained these end points are checked for completeness. Any of these faces containing only complete edges is tagged as complete.

6. Merging the Topology of the Two Descriptions

At this point, all mergers within each description have been done, so that each element in one description will match at most one element in the other description. To merge elements across the two descriptions, it is simpler to merge the topology separately from the geometry.

The first step is to create a new empty vertex, edge, or face node for each matching pair of vertices, edges, and faces, respectively. The new nodes will represent the merging of each original pair.

The next step is to merge the topology of the two initial descriptions. This is accomplished by substituting, in the two structure graphs, each new node for its matching pair. For example, each new vertex node is given a topological pointer to each of the edges and faces pointed to by its matching pair, and all edges and faces with pointers to either of the pair are given a pointer to the new vertex instead. New edge and face nodes are handled in a similar manner.

7. Merging the Geometry of the Two Descriptions

The next step is to merge the geometry of matching pairs of elements. For each new vertex node previously generated, the original matching pair of vertices is examined. If both are not non-vertex end points (i.e., both correspond to real vertices in their descriptions), then the position of the new vertex is the average position of the original pair. Notice that this algorithm gives equal weight to the positions of the original vertices.

Next, for each new edge node previously generated, the original matching pair of edges is examined. Three different merging techniques are used depending on whether both original edges are complete, both are incomplete, or one is complete and one is complete. If both original edges are complete (Fig. 6a), the 3-space line of the new edge merely connects the two new vertices previously obtained. If one of the original edges is complete and the other is incomplete (Fig. 6b), we must obtain the line equation of the new edge and the position of the new vertex on the edge. (Because the new vertex is formed by merging a vertex and a non-vertex end point, its coordinates have not yet been determined.) To see how this is done, consider Fig. 6c. Let $newv_1$ be the new vertex formed by merging v_1 and v_3 in Fig. 6b. Let p be the average position of the other two end points of the edges, i.e.,

$$\vec{p} = (\vec{v}_2 + \vec{p}_1)/2.$$

The line of the new edge is a line containing $newv_1$ and p. The position of the other new vertex, $newv_2$ lies on this line at a distance from $newv_1$ equal to the length of the original complete edge, and in the same direction from $newv_1$ as $newv_2$, i.e.,

$$\vec{newv}_2 = \vec{newv}_1 + |e_1| [(\vec{p} - \vec{newv}_1)/|\vec{p} - \vec{newv}_1|].$$

To see intuitively why the resulting new edge should pass through the midpoint of p_1 and v_2 in Fig. 6c, consider the two extreme cases for the relative lengths of e_1 and e_2. If the length of e_2 equals the length of e_1, then we want the resulting edge to lie along the angle bisector of e_1 and e_2, which it will under our formula. On the other hand, if the length of e_2 approaches 0, then we want the resulting new edge to lie along the edge e_1, which it will under our formula.

If both original edges are incomplete (Fig. 6d), the line of the new edge and position of the new vertex is obtained exactly as for the case of one complete edge and one incomplete edge, except that the length of the new edge is the same as that of the longest of the original edges e_1 and e_2. Also, the new vertex is tagged as a non-vertex end point and the new edge is tagged as incomplete.

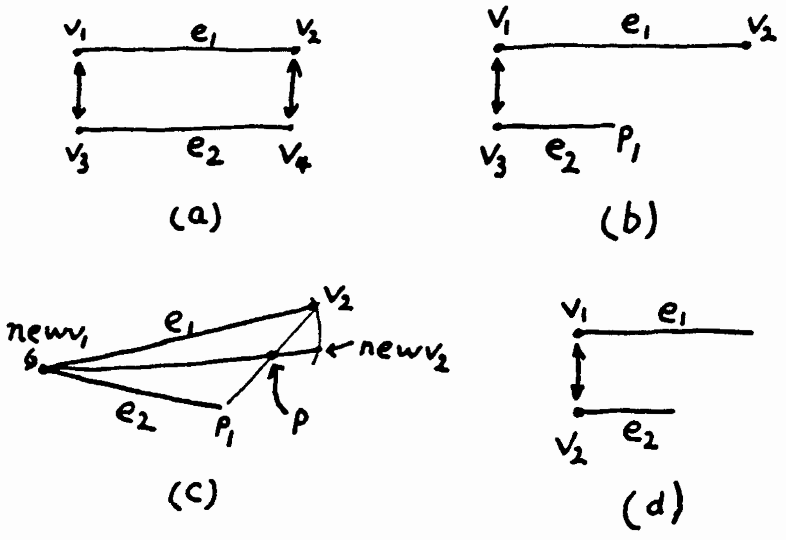


Fig. 6

8. Adding Elements with No Matches

Both description 1 and description 2 will often have elements that have no matches. Merging these elements into the final structure graph involves two steps: (1) modifying their topological pointers so that they obtain links to the newly created nodes formed by merging pairs of matching elements, and (2) modifying their geometry so that they are represented in the same coordinate system as the final description. But these two steps have already been performed. All topological pointers were modified immediately after the new elements were created, and the geometry of all elements were modified when the coordinate transformation between the two original descriptions was calculated.

For each new face node previously generated, its plane equation is calculated by a least squares fit to all the new vertices of the face. If any new edge of the face is incomplete, the face node is tagged as incomplete. Otherwise, it is complete.

9. Experimental Results

This section presents results of the merging algorithm on some examples. The first example involves 3-D descriptions automatically generated from range data [5]. Fig. 7 shows the descriptions generated from two views of an object. This might be an example of the application mentioned earlier of learning the shape of an object presented via multiple views. Fig. 8 shows two views of the resulting description after matching and merging the descriptions in Fig. 7.

Our next example involves a complicated office scene containing chairs and a table. This might be an example of the application mentioned earlier of a mobile robot learning its environment by moving through it. The 3-D descriptions in Fig. 9 were manually generated from synthesized depth maps. Fig. 10 shows the resulting description after matching and merging the descriptions in Fig. 9. Fgi. 11 shows an additional view of the office scene. Fig. 12 shows the resulting description after matching and merging the descriptions in Figs. 10 and 11. Note how the scene model is incrementally updated with 3-D information from each new view.

References

1. Baer, A., Eastman, C., and Henrion, M. "Geometric modelling: a survey." *Computer-Aided Design*, 11, 1979, 253-272.

2. Bourne, D.A., Milligan, R., and Wright, P.K. "Fault Detection in Manufacturing Cells Based on Three-Dimensional Visual Information." *Proc. SPIE* 336, May 1982, 29-36.

3. Crowley, J.L. "Dynamic World Modeling for an Intelligent Mobile Robot." *Proc. Seventh International Conference on Pattern Recognition*, July, 1984, 207-210.

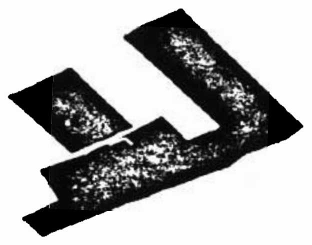
4. Eastman, C.M. and Preiss, K. "A Unified View of Shape Representation and Geometric Modeling." *Israel Conference on CAD*, January 1982.

5. Herman, M. "Generating Detailed Scene Descriptions from Range Images." *Proc. 1985 IEEE International Conference on Robotics and Automation*, St. Louis, Missouri, March 1985, 426-431.

6. Herman, M. "Matching Three-Dimensional Symbolic Descriptions Obtained from Multiple Views of a Scene." *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, San Francisco, California, June 1985, 585-590.

7. Herman, M. "Representation and Incremental Construction of a Three-Dimensional Scene Model." In *Techniques for 3-D Machine Perception*, Rosenfeld, A., ed., Elsevier Science Publishers B.V. (North-Holland), 1986, 149-183.

8. Herman, M. and Kanade, T. "The 3D Mosaic Scene Understanding System." In *From Pixels to Predicates: Recent Advances in Computational and Robotic Vision*, Pentland, A. P., ed., Ablex Publishing Corporation: Norwood, NJ, 1986, 322-358.


(a)

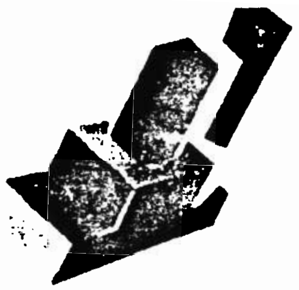
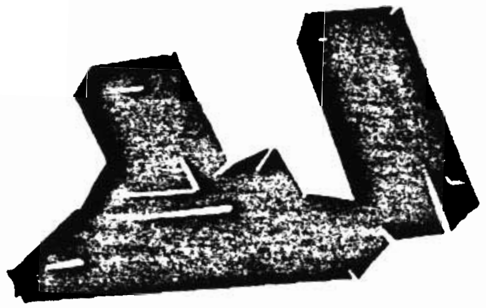

(b)

Fig. 7


(a)

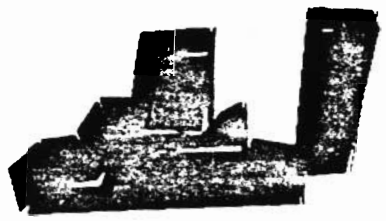
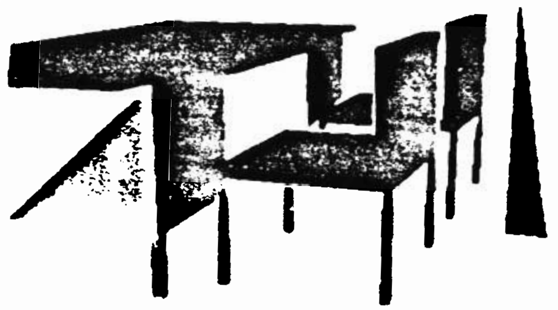

(b)

Fig. 8


(a)

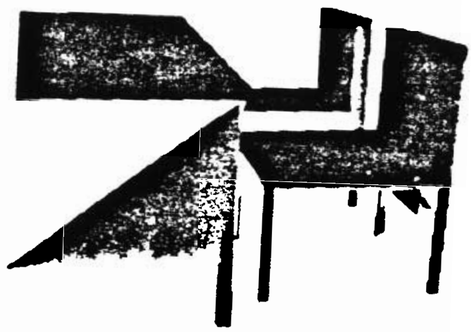

(b)

Fig. 9

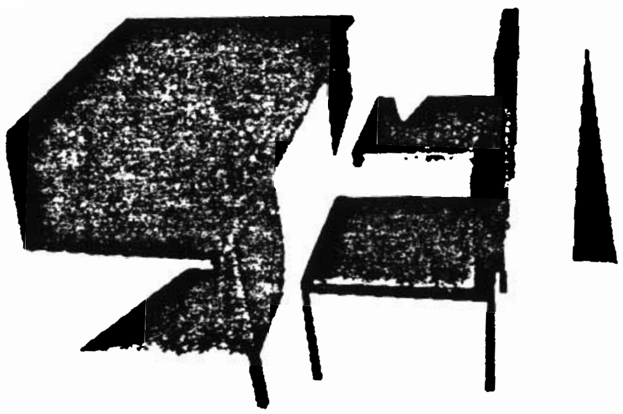


Fig. 10

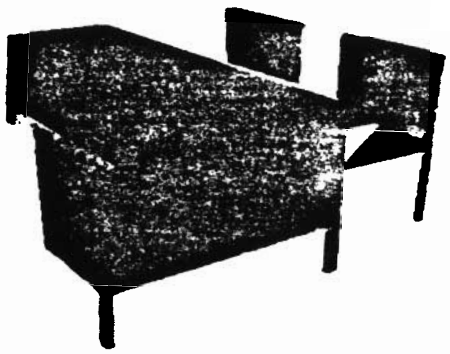


Fig. 11

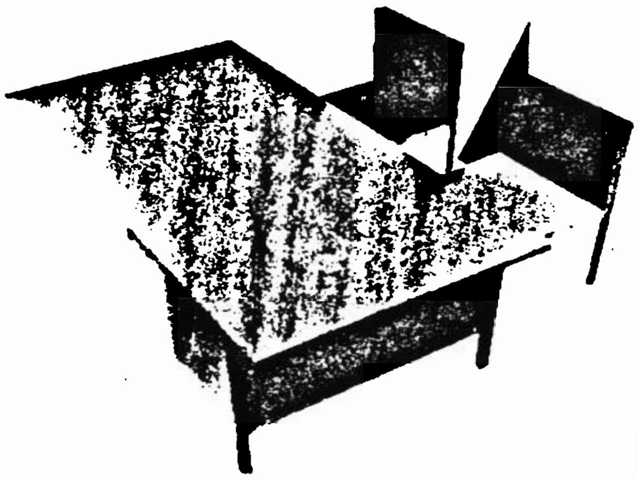


Fig. 12

9. Herman, M. and Kanade, T. "Incremental Reconstruction of 3-D Scenes from Multiple, Complex Images." *Artificial Intelligence*, 1986, to appear.

10. Moravec, H.P. "Obstacle Avoidance and Navigation in the Real World by a Seeing Robot Rover." Tech. Rept. CMU-RI-TR-3, The Robotics Institute, Carnegie-Mellon University, Pittsburgh, PA., September 1980.

11. Shneier, M., Kent, E., and Mansbach, P. "Representing workspace and model knowledge for a robot with mobile sensors." *Proc. Seventh International Conf. on Pattern Recognition*, Montreal, Canada, July 1984, 199-202.

12. Tomita, F. and Kanade, T. "A 3D Vision System: Generating and Matching Shape Descriptions in Range Images." *The First Conference on Artificial Intelligence Applications*, December 1984.

13. Underwood, S.A. and Coates, C.L. "Visual Learning from Multiple Views." *IEEE Transactions on Computers*, 6, June 1975, 651-661.