Regular Paper

Heuristic Understanding of Three Orthographic Views

CHANGHUN KIM*, MASAHIRO INOUE* and SEIICHI NISHIHARA*

This paper describes a system that reconstructs 3-D models of polyhedrons from three orthographic views represented by binary images. Our main stress is on the description of a new reconstruction algorithm. The proposed algorithm performs a combinatorial search based on the face decision strategy along with two heuristics. The constraint rules are developed on the basis of the properties of orthogonal projection geometry, which are dynamically applied to the combinatorial search process. Further, two heuristics are adopted in the combinatorial search process so that its way of interpretation follows that of human understanding. One is for ordering the search nodes to reduce the search space, and the other is for reconstructing the scenes as naturally as possible. The effectiveness of introducing heuristics is also proved and analyzed experimentally.

It is shown that 3-D scenes can successfully be reconstructed by the proposed algorithms, and that the efficiency of the understanding system is improved considerably by the introduction of the heuristics.

1. Introduction

Engineering drawings composed of three orthographic views have conventionally been used as common tools for representing three-dimensional (3-D) objects on two-dimensional (2-D) sheets of paper. In computer graphics and CAD, modeling 3-D objects is still a laborious and time-consuming task, although it is a fundamental problem that has been studied for thirty years by many researchers. Recently, understanding engineering drawings has emerged as a promising approach to the computerization of 3-D design in manufacturing. Many studies on the reconstruction of 3-D scenes from given 2-D drawings have already been reported. A reconstructed scene is often composed of two or more 3-D objects, since a 2-D drawing can easily express many objects at once.

When the original drawing is prepared as a naive binary image itself, any understanding system should inevitably contain two phases: preprocessing, in which geometrical features and elements are extracted from the original binary image to recover the 2-D structure expressed in the drawing, and reconstruction of the 3-D scene in a way consistent with the 2-D structure recovered in the preprocessing phase. Therefore, while preprocessing mainly uses image processing techniques, reconstruction is primarily a reasoning or search process in artificial intelligence. Most reconstruction systems developed up to now are only for reconstructing drawings whose 2-D structure has been recovered beforehand; they do not include a preprocessing phase.

In general, the reconstruction phase first recovers one by one all of the possible 3-D components such as edges and faces, which we call candidate components. Only some of these candidates, which are called true components, will actually become parts of the final valid objects in the scene reconstructed; the others, which are called false components, do not contribute to the final solution. In other words, therefore, reconstruction is a filtering process that eliminates all the false components from the set of the candidate components. From the viewpoint of filtering technique, reconstruction methods are classified into three main classes: the direct elimination approach [1, 2, 3], the primitive construction approach [4, 5, 6], and the face-oriented approach [7, 8, 9]. One of the earliest reports on the last approach is by Haralick. et al. [7]. We have also proposed a face-oriented approach [9], in which we demonstrated that scene reconstruction can be realized on the basis of several constraint rules that should be satisfied geometrically as 3-D objects, and that the reconstruction process can be expressed ingeniously by a typical tree search algorithm to find a permissible subset of candidate components, or candidate faces.

In this paper, we propose another face-oriented method emphasizing heuristics, whose major characteristics can be contrasted with those of the old method as follows [9]:

1) A new reconstruction algorithm: The method is essentially a combinatorial search that decides whether each candidate face is true or false. To realize the deci-

*Institute of Information Sciences and Electronics, University of Tsukuba, Tsukuba, Japan.

sion procedure, 3-D geometrical constraints that are satisfied by the legal set of true faces are analyzed and summarized as a set of rules.

2) New heuristics: Two types of heuristics are newly introduced. One is for speeding up the search process, and the other is for finding more natural solutions in preference to unusual ones. These heuristics, so to say, try to let the search process simulate the human way of understanding three orthographic views.

3) Improved preprocessing techniques: To improve the accuracy and the efficiency of the preprocessing phase, we developed a modified Hough transformation and techniques for adjusting geometrical features by using inter-view relations. However, we explain the preprocessing only briefly in this paper.

Section 2 gives definitions and an overview of the understanding system. Section 3 is the main part of the paper, and describes the new reconstruction algorithm and heuristics. Preprocessing is also reviewed briefly. Section 4 describes experiments performed by applying the new algorithm, with heuristics, to some actual hand-drawn samples. Section 5 presents our conclusions.

2. Outline of the System

2.1 Definitions

2.1.1 Three Orthographic Views

An original drawing represented by three orthographic views, namely, top, front, and right-side views, should be drawn correctly on a paper by using a ruler. For each view, the viewpoint, from which the projection beam emanates, is placed at an infinite distance from the projection plane. The only geometrical elements included in each view are line segments. No other supplementary elements such as chain lines indicating central axes, or numerals indicating dimensions, appear in the drawing. There are two types of segments: solid and broken. A segment is defined by its two end points. The degree of each point is two or more; that is, at least two segments should meet at each point.

A closure of connected segments in a view is called 'an area.' We call an area 'simple' if all of its boundary segments are solid and it contains no solid segments.

2.1.2 3-D Object Scenes

A 3-D scene to be understood by our system is composed of one or more polyhedral objects, each of which is a closed section of a non-empty subset of 3-D space bounded by a finite number of polygonal faces. A face is a directed finite area in a plane bounded by a closed sequence of edges lying on the same plane. A vertex is a point at which three or more edges meet. The 3-D scene should not contain any volumeless parts such as a floating face having no other contiguous face or a couple of faces lying on the same plane back to back. Simply, the system accepts any object scene composed solely of polyhedrons that are actually arrangeable in a gravity-free space.

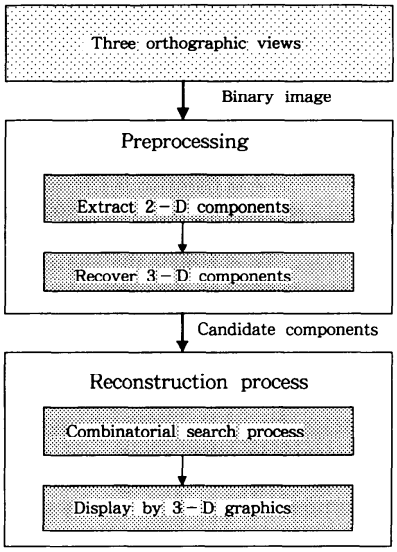


Fig. 1 System Overview.

2.2 System Overview

Figure 1 shows the general flow of the system. This system is divided into two main parts: a preprocessing part and a reconstruction process. In the preprocessing part, three orthographic views drawn on a paper are scanned to be digitized by an image scanner, and feature points and line segments are extracted from the binary digitized image. The geometrical distortion of the extracted geometrical elements is adjusted by finding the correspondence of elements among the three views and by examining the collinearity of line segments. The 3-D candidate components of an object scene are recovered from the 2-D elements extracted. In the reconstruction process, combinatorial search is performed to find all the permissible combinations of candidate faces, or solutions, that can form a valid solid model. The 3-D graphics module displays the reconstructed solid models, and an interactive interface function is also implemented for manipulating displayed models.

2.3 Preprocessing

2.3.1 Extracting 2-D Geometrical Elements

To extract geometrical elements from an input binary image of three orthographic views, we developed a modified Hough transformation, where line fragments (consecutive pieces of segments derived by following the borders of the image) are used as units of Hough transformation [10], to reduce the processing time and

memory space. The transformed line fragments are plotted and make clusters in Hough parameter space; thus, line segments are grouped and recognized as solid line segments and broken line segments. The collinearity of line segments is also computed by using the properties of Hough transformation.

The extraction of 2-D elements is affected by various distortions such as: drawing mistakes, extraction of unwanted elements, and elements not consistent with corresponding views. To eliminate the above distortions, the elements extracted are classified into two groups: the set of elements parallel to a projection beam, and the set of elements not parallel to any projection beam. There are two types of geometrical constraint: inter-group and intra-group constraints. These have two properties. Each element in a view should have corresponding elements in other views, and collinear line segments in the original drawings must remain collinear. Distortion is compensated for by satisfying the above constraints instead of computing the congruence and similarity of geometrical figures in drawings.

2.3.2 Recoverint 3-D Candidate Components

Each 3-D component of a polyhedron—a vertex, an edge, or a face—is orthogonally projected to a point, a segment or a point, and an area or a segment, respectively in the 2-D drawings. All the possible 3-D components constituting the object scene can be recovered from the 2-D elements on the basis of the above orthogonal projection. However, not all of the 3-D components are necessarily used to construct the object scene. Some 3-D components which we call false components, are not included in the real scene. Thus, the 3-D components recovered are called candidate components, namely, candidate vertices, candidate edges, and candidate faces.

To see briefly how a candidate vertex is recovered, let three points, $P_1=(X_1, Y_1)$ in the front view, $P_2=(Z_2, X_2)$ in the top view, and $P_3=(Y_3, Z_3)$ in the right side view, be the orthographic projections of the same vertex V. Then, three equations, $X_1=X_2$, $Y_1=Y_3$ and $Z_2=Z_3$, can be utilized in recovering all of the possible candidate vertices. Here we do not describe the method of recovering 3-D components, whose fundamental idea is based or the one proposed by Haralick et al. [7]. Similar inter-view correspondences are also used to recover candidate edges and candidate faces.

2.4 Reconstruction Process

Here we give a concrete definition of the reconstruction problem. Let Δ be the set of all segments of the original drawing, including solid and broken segments, and let Φ be the set of all candidate faces. Our reconstruction process can be defined as a process of finding each solution that is a subset Γ of set Φ, that is, $\Phi \subseteq \Phi$, satisfying the following two conditions:

1) Γ should form a legal scene composed of one or more polyhedrons, none of which contains a volumeless part such as a floating face having no other contiguous face, or a couple of faces lying on the same plane back to back or intersecting each other.

2) Let Δ' be the set of all segments included in the orthographic views of scene Γ, Then, Δ' should be equal to Δ, that is, $\Delta=\Delta'$.

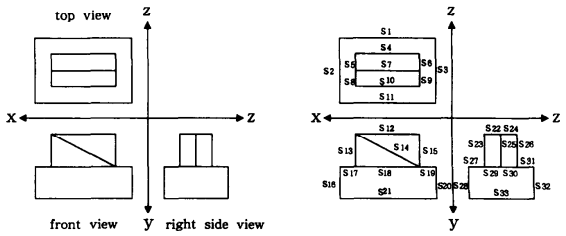
The first condition ensures that Γ is a permissible scene realizable in 3-D space. The second one certifies the coincidence of the scene Γ, even if it is not realizable, with the original drawing.

Figure 2 gives an example of the solution this problem. Figure 2(a) is the original input drawing; Figure 2(b) shows the set Δ of segments extracted from Fig. 2(a): Figure 2(c) shows the set Φ of candidate faces recovered from Δ; and Fig. 2(d) shows three feasible solutions, Γ's, satisfying the above two conditions, where

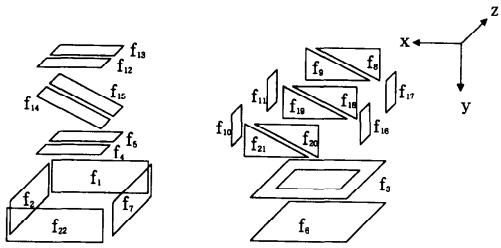
$$\Delta=\{s1, \ldots, s33\},$$
$$\Phi=\{f1, \ldots, f22\},$$

and one of the solutions is

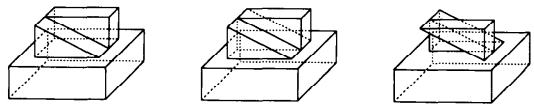
$$\Gamma=\{f1, f2, f3, f6, f7, f8, f9, f10, f11, f13, f14, f17, f18, f21, f22\}.$$



(a) Original three orthographic views (b) Segments extracted from (a)



(c) Candidate faces recovered from (b)



(d) Scenes recovered from (c)

Fig. 2 Example of the reconstruction process.

3. Reconstruction Algorithm

A reconstruction problem can be considered as a combinatorial search problem that finds permissible combinations of faces, each of which forms a valid object scene. Instead of repeating generate-and-test operations to check every combination exhaustively, our algorithm adopts a more sophisticated approach.

In the algorithm, the reconstruction process is basically a face determination process that determines the state, true or false, of each candidate face, on the basis of a combinatorial search. The face state is determined by using face determination rules that satisfy the constraints certifying the realizability of the object scene in 3-D space and the coincidence of the reconstructed scene with the original drawing.

If one or more candidate faces are still left undetermined, recursive combinatorial search is performed on those undetermined faces. In the process of combinatorial search, the current node is always checked, whether or not it is a final node. Thereafter, instead of a simple recursive tree search method, two heuristics are used: one for choosing a face from the set of undecided faces, the other for branching to one of two states, true or false, of the chosen face. The details of the algorithm and the heuristics will be described in the following section.

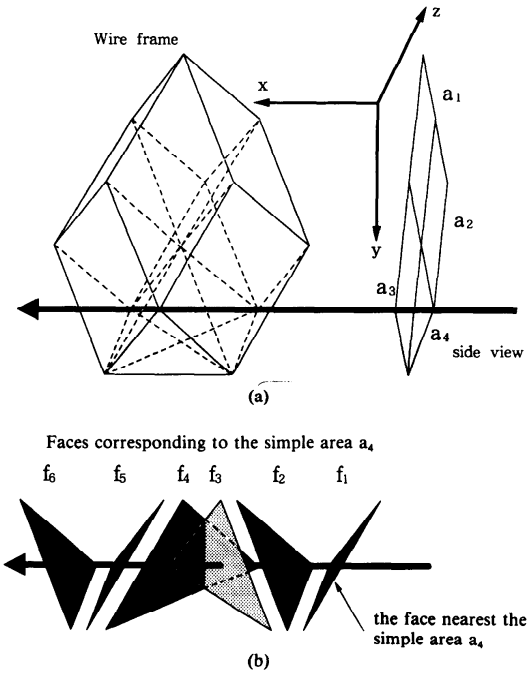


Fig. 3 Corresponding faces and the nearest face.

3.1 Face Determination Rules

To determine the face state and check whether the reconstructed scene is legal, we derive a set of rules based on the properties of 3-D geometry and its local geometrical constraints. Here two terms are introduced to describe the local geometrical constraints: a set of corresponding faces, or CF for short, and the nearest face. A CF is a recovered set of possible candidate faces corresponding to a simple area in the original drawing. The nearest face is the candidate among the corresponding faces that is nearest to the viewpoint and whose face state is not false. For example, the CF to the simple area a4 in Fig. 3(a) consists of six candidate faces, as shown in Fig. 3(b), and the nearest face to the simple area a4 is f1.

(Rule 1) Two or a larger even number of faces should be connected through each edge.

(Rule 2) When two or more faces intersect, at most one of them can actually be adopted as a component of a solution.

(Rule 3) For each segment in a drawing, there must be at least one edge projected to it.

(Rule 4) If a face's boundary contains an edge whose projection appears as a broken segment in a view, there should be at least one other face placed nearer to the viewpoint. (If face f is nearer than face g, it means that face f is placed nearer to the viewpoint than face g, or farther from the projection plane than face g).

Rules 1 and 2 correspond to the condition for making a permissible scene realizable in 3-D space, and are derived from a very fundamental feature of 3-D objects: "The necessary and sufficient condition that one or more objects can actually exist in 3-D space is that any closed curve crosses the surfaces of objects zero times or an even number of times."

While Rules 1 and 2 guarantee the realizability of the scene, Rules 3 and 4 guarantee the coincidence of the reconstructed scene with the original drawing.

The above rules are mainly used to check the validity of a scene in the midst of the reconstruction process. In addition to the above rules, on the basis of more concrete local geometrical constraints, we adopt three further rules:

(Rule 5) If two nearest faces projected into two areas adjacent to each other are coplanar, at most one of them can be true. However, if the above two areas in the drawing are connected with broken segments, both of the two nearest faces with the same face state, true or false, should be assigned to both of the nearest faces.

(Rule 6) For each simple area in a drawing, zero faces or an even number of faces in the CF must be true.

(Rule 7) For each area adjacent to the background, at least two faces in the CF must be true. In particular, if there are only two faces in the CF, both of them are concluded to be true in any solution.

Figure 4 shows examples of local geometrical constraints related to the above rules. Figure 4(a) shows the original three orthographic views, and Fig. 4(b) shows a

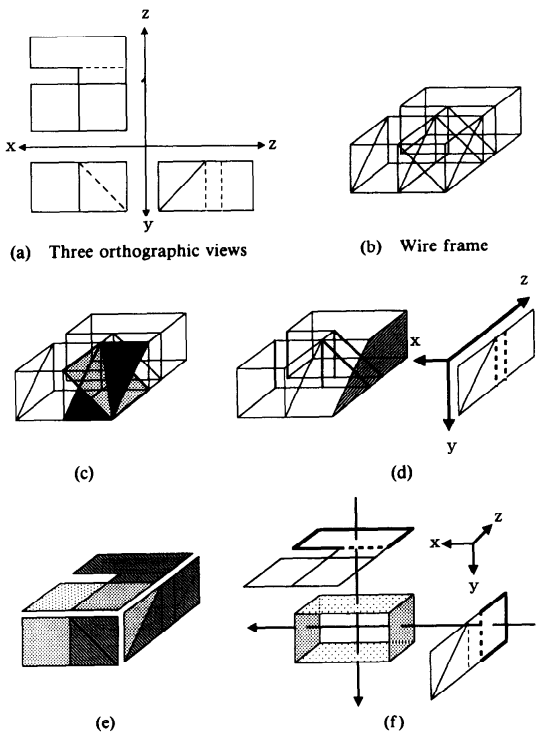


Fig. 4 Various types of local geometrical constraint.

```
begin
    T := F := {} ; U := Φ;                                        ---- (1)
    heuristics 1  - - set the first node                          ---- (a)
    search( T, F, U );
end.
```

(a) Main program

```
procedure search( T, F, U )
 [
   while ( a face f (∈ U) matching with Decision Rule exists ) do   ---- (2)
   [
      determine the state of f using the DR, and move f (∈ U) to T or F:
      if ( any contradiction is found ) then return:               ---- (3)
   ]
   if ( U = {} ) then                                              ---- (4)
      T gives a solution:
   else
   [  clear stack P:
      heuristics 1'  - - get a face f (∈ U)                        ---- (b)
      heuristics 2   - - choose τ of f, and push (¬τ and τ) to P   ---- (c)
      while ( P ≠ {} ) do
         branch( T, F, U, f, pop-up(P) ):                          ---- (5)
   ]
 ].
```

(b) Determination and expansion procedure

```
procedure branch( T, F, U, f, state )
 [
   move f (∈ U) to T or F:                                         ---- (6)
   search( T, F, U ):                                              ---- (7)
 ].
```

(c) Branch operator

Fig. 5 Reconstruction procedure.

wire frame composed of candidate edges. To illustrate Rule 2. Figure 4(c) shows two intersecting faces. In Fig. 4(d), the hatched face is nearer to the broken segments shown in the y-z plane, a situation which is related to Rule 4. Figure 4(e) shows examples of nearest faces projected into two areas adjacent to each other, a situation which is related to Rule 5. Figure 4(f) shows examples of particular cases in which there are only two faces in the CF, a situation which is related to Rule 7.

3.2 Rule-based Algorithm

Figure 5 shows the general structure of the procedure for reconstructing 3-D scenes. Neglecting heuristics statements (a), (b), and (c) in Fig. 5, we get a simple procedure, called the rule-based algorithm, which we describe here.

In the face determination process, the face determination rules are repeatedly applied to the current node in order to determine the state of candidate faces. When some candidate faces are determined to be true or false, the local assignments of geometrical states are updated to correspond to the changed states of candidate faces. After that, other undetermined candidate faces are checked dynamically according to the new local assignments of geometrical states.

If no candidate face is newly determined, even after some local assignments of geometrical states have been changed, one face arbitrarily chosen is forcibly assigned a state value, namely true or false, and the tree search proceeds to branch down the search tree. The above procedure is recursively performed until all the possible solutions have been found.

In Fig. 5, U is a set of under-processing faces, T is the set of faces determined to be 'true,' and F is the set of faces determined to be 'false.' Both T and F, independent of each other, are subsets of candidate faces. As the search proceeds, faces in U are determined to be true or false and moved to T of F one after another. In the search tree, each node represents a subset of candidate faces, and the set for a node is always a superset of the set for its parent node. Thus, each branch-down operation in the tree search is actually an operation for adding one or more candidate faces, as is shown in the branch procedure.

In step (1), each parameter is initialized. T and F are initially set to be empty. U is initially a set of all of the candidate faces. After this, a search process is started by the procedure 'search.' In step (2), a determination process is performed until no more faces can be determined by the face determination rules. In step (3), if

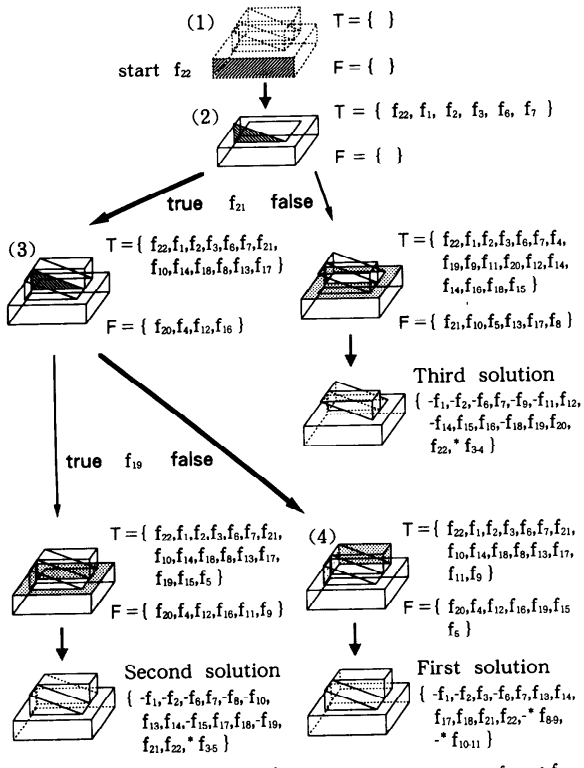


Fig. 6 Example of search process.

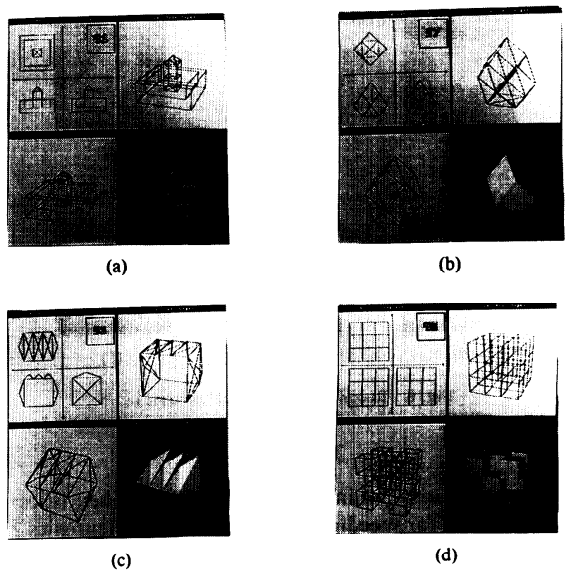


Fig. 7 Sample drawings and reconstructed scenes.

any contradiction is detected by using Rules 1, 2, 3, and 4, backtracking is invoked and the search proceeds to the other remaining branches. When U is found in step (4) to have become empty, T gives a solution. In step (5), even though the local geometrical constraints are changed, if no face matches any face determination rule, then a candidate face still remaining in U is chosen arbitrarily. Both the true and false states of the new face are pushed onto the stack P, and the search branches down to one of two states until the stack P becomes empty. In step (6), the sets T, F, and U are updated according to the state newly popped up from the stack P. In step (7), a search is recursively called for to the new state popped up. Local geometrical constraints are newly generated by forcibly assigning true or false to the face, and the search process continues in this way.

In the algorithm, searching and branching mutually form an indirect recursive procedure. They check every possible combination of candidate faces; therefore, all possible solutions are reconstructed, even when the original drawing contains some ambiguities permitting more than one interpretation.

3.3 Introducing Heuristics

Although we can obtain all the possible solutions with the above algorithm, this kind of exhaustive search involves some difficulties. The processing speed is relatively slow, especially when the set of candidate faces is large. If the original drawing contains any ambiguity permitting more than one solution, there will be a problem in generating the order of solutions. To resolve

Table 1 Experimental results.

case	#candidate faces	#solutions	heuristics	CPU time ($\times 10^{-2}$ sec)	#branches	#face checks
Fig. 7 (a)	42	49	—	23.6	117	1787
			1 & 2	18.2	75	1386
Fig. 7 (b)	32	50	—	14.1	61	1402
			1 & 2	12.9	53	1040
Fig. 7 (c)	55	51	—	20.7	96	1550
			1 & 2	13.2	59	1310
Fig. 7 (d)	108	1152	—	482.1	1222	31474
			1 & 2	393.2	1178	23298

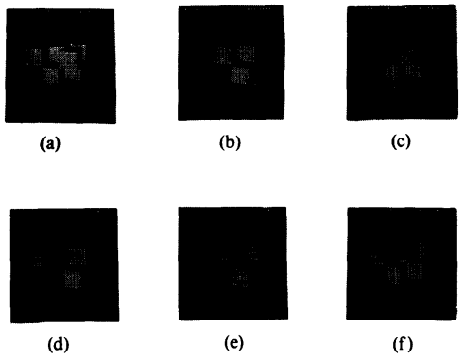


Fig. 8 Reconstructed feasible scenes.

Table 2 Comparison results.

sample id.	#candidate faces	#solutions	CPU time ($\times 10^{-2}$ sec)	
			present method	conventional method [9]
1	12	1	0.3	3
2	12	1	0.3	10
3	19	1	0.3	25
4	25	1	0.4	29
5	30	1	2.2	72
6	28	1	0.4	76
7	36	2	0.4	87
8	30	1	0.6	109
9	35	1	0.6	148
10	20	5	1.8	173
11	47	1	0.8	181
12	20	35	4.5	561
13	33	5	1.8	1025
14	36	3	0.9	1498
15	42	49	18.2	4176

these difficulties, we introduced two types of heuristics: Heuristics 1 for speeding up the search, and Heuristics 2 for reconstructing the most natural scene first among all the feasible scenes. These heuristics try to let the understanding system follow the human way of interpreting three orthographic views.

[Heuristics 1]

When a human tries to understand three orthographic views, he first selects the face that has experience tells him to be true, and then starts the reconstruction from that face. For example, he usually selects first the face nearest the viewpoint or an effective face that has a strong influence on determining the state of the other faces. Heuristics 1 is used as a set of rules for choosing the candidate face that is the most effective for reaching the final solution as fast as possible before branching. In Fig. 5, statements (a) and (b) show where Heuristics 1 works out. In statement (a), heuristics 1 works by ordering simple areas before starting the search: it sets the first node of the search. In statement (b), heuristics 1' works for choosing a face in U; it sets the next node to branch. Heuristics 1 can be restated as the following set of heuristic rules:

(H-1) When a human draws three orthographic views, he usually begins with the front view.

(H-2) A nearest face there is recovered from a simple area placed nearer to the origin of the coordinates in the original drawing contains more information than the nearest faces of the other simple areas.

(H-3) A simple area having fewer corresponding faces should be given a high priority in the processing order.

(H-4) A simple area containing silhouettes bordering on the background of the view is usually considered to have effective corresponding faces.

(H-5) In general, the nearest face is one of the most effective faces.

[Heuristics 2]

When human sees a 3-D object, he usually tends to imagine the invisible part to be as simple as possible. By using this heuristic knowledge, more natural 3-D scenes will tend to be derived before complicated ones. Heuristics 2 is used as a rule for determining the state of a face chosen by using Heuristics 1, instead of deciding it arbitrarily. In Fig. 5, statement (c) shows how Heuristics 2 works to determine the state of the face chosen for branching. If the face is hidden from the viewpoint, Heuristics 2 prefers to take the false state, because a human tends to imagine the hidden part to be as simple as possible. The decision on a hidden face is checked under the following conditions:

1) The face intersects with an other face.

2) The face is located inside of one of the faces in the CF that are not parallel to the projection plane.

3) The face is the nearest face and is connected to an inclined face located inside the projection plane.

4) The face is an inclined face and not the nearest face.

5) The face is not the nearest of the faces in the CF, only two faces remain to be determined, and an odd number of faces already been determined to be true.

[Example]

Let us see how the above heuristics can be applied to an actual example. Figure 6 shows details of the process for obtaining three feasible solutions consistent with the original drawing in Fig. 2(a). At first, by using Heuristics 1, the front view in the drawing is chosen as the starting node of the search. After this, some candidate faces can be concluded to be true or false one after another by using the face determination rules. If some faces are still left undetermined, search and branch procedures are recursively performed to obtain solutions by using the heuristics. The details of the search process are described in the following. The identifier of candidate faces in Fig. 6 refers to Fig. 2(c).

(Step 1) The first search node is settled as a set containing a single face {f22} by using Heuristics 1.

(Step 2) In this node, first of all, faces f22 and f1 are concluded to be true by using Rule 7, because they are the only two faces in the CF. Faces f2, f7, f3 and f6 are also concluded to be true by using Rule 1, because each of them is the only face connected to each edge of the true face f22. When this determination process cannot proceed further in the current node, face f21 is selected as the next search node by using heuristics 1', A search is then performed on both states of face f21: true and false.

(Step 3) The search is made to branch to the true state of face f21 by using Heuristics 2, because f21 is not in the false condition. Face f20 is concluded to be false by using Rule 5, because it is coplanar with and adjacent to f21. Faces f10 and f14 are concluded to be true from Rule 1, because each of them is connected to the edge of the true face f21. By using the same rule, faces f14, f4, f12, and f16, each of which is the only face connected to each edge of the false face f20, are also concluded to be false. Faces f18 and f8 are concluded to be true by using the Rule 6, because they are the only two faces in the CF of their simple area. Faces f13 and f17 are also concluded to be true, because they are connected to each edge of the true face f18. When the determination process cannot proceed further, one face, f19, is chosen by using Heuristics 1. Heuristics 2 branches the search to the false state as to face f19, because f19 is in the false condition of Heuristics 2.

(Step 4) Faces f11, f15, and f9 are concluded to be true or false by using Rule 1.

(Step 5) Steps 1 to 4 are repeated until all feasible solutions have been obtained.

(Step 6) When the final node is reached, face merging operations are performed on the faces in T and the direction of the face normal is examined.

(Step 7) Finally, we obtain three solutions. In the set of solutions, a minus sign shows that the direction of the face normal is opposite to the viewing direction.

4. Experiments

4.1 Experimental Evaluation of the Heuristics

The understanding system described above is implemented on a Sun4 Sparc 1+ workstation in C language. A4-sized drawings representing three orthographic views are scanned by an GT-4000 scanner with a step of 4 pixels/mm. Figures 7(a), (b), (c), and (d) show various examples of three orthographic views and corresponding reconstructed scenes.

An experiment was performed by using two methods: the rule-based method with no heuristics incorporated, and the heuristics-directed method. Table 1 gives the experimental results of the methods applied to the above four examples. Both of the methods succeeded inobtaining all of the feasible solutions for the four samples. The results are compared with respect to the following items: the number of candidate faces, the number of solutions, the search time needed to obtain all the feasible solutions, the number of branches performed, and the number of times the face states were checked. Table 1 shows that the heuristics-directed method most effectively reduces the search space and CPU time needed. Figure 8 shows an example of the effect of Heuristics 2. In Fig. 8, six feasible solutions selected from the 1152 possible solutions are arranged, preserving the sequence of derivation. Although all of the scenes in Fig. 8 are valid solutions, we can easily see that the first scene, Fig. 8(a), is more natural than the other five scenes derived later.

The effects of using heuristics have been proved experimentally: one is an increase in the search speed and the other is earlier reconstruction of more natural scenes. However, heuristics are not effective in all cases; if all of the candidate faces can be determined by using the face determination rules, heuristics need not be used. In a few cases, the results are almost the same when heuristics are used as when they are not.

4.2 Comparison with Other Algorithm

We have presented a face-oriented reconstruction algorithm before [9]. The new algorithm is an extension of this old one. However, there are some differences between them, which can be summarized in three phrases: the tree structure used in the search, local geometrical constraints, and the mechanism of knowledge usage.

1) Tree structure: The new method adopts a binary tree structur that is used to search for two possible face states: true and false. However, the old method takes the multiway tree structure; the number of leaves is determined by the number of candidate faces connected with the current candidate edge being processed. In other words, the new method pays attention to the face states, and the old method pays attention to the edges shared by the faces. In the old method, the combinatorial search can be expressed as a process for finding only a legal subset of candidate components. But the new algorithm tries not only to search for legal candidates, but also to eliminate false ones.

2) Local geometrical constraints: In the old method, the local geometrical constraints are applied to edges only. In the new method, face constraints are also introduced.

3) Mechanism of knowledge usage: The above local geometrical constraints are summarized as face determination rules in both algorithms. In the old method, the rules are applied only once before starting a search; in the new method, on the other hand, the rules are applied at every current node of the search process, and new heuristics are also adopted in the search process.

Table 2 compares the experimental results for several examples. Both the approaches of the new method–rule-based and heuristics-directed—have a much shorter processing time than the old method. However,

it must be noticed that heuristics-directed approach gives the best result. This approach is particularly effective if the problem has many solutions.

5. Conclusions

We have described a system that reconstructs 3-D scenes from three orthographic views represented by binary images. In this system, *a priori* knowledge of the characteristics of the three view drawings and 3-D geometry is analyzed and then applied to the reconstruction process. In the proposed algorithms, this knowledge is synthesized in sets of rules that govern the 3-D geometry and topology of the geometrical elements in drawings. In this paper, the problem of reconstructing 3-D scenes from three orthographic views is set up essentially as a kind of combinatorial search. The rules are applied to the combinatorial search according to the change in the local assignment of geometrical states. This frees us from unnecessary searching. In general, although the combinatorial search process is one of searching for the set of legal combinations, the new approach proposes the effective combinatorial search process that not only searches for legal combinations but that also simultaneously eliminates illegal combinations.

Another emphasis of this paper is on introducing two types of new heuristics: one to reduce the search space and the other to reconstruct more natural 3-D scenes before more complicated ones. The heuristics are concretized as a set of heuristic rules that try to follow the human way of imagining 3-D scenes from 2-D drawings. The efficiency of the heuristics has been proved by applying them to a set of test drawings.

The system based on the algorithm proposed here can serve as an interface between paper-based drawings and solid modeling in CAD. It can also be used as an input tool for a man-machine interaction system.

We now mention some ideas for possible and necessary extensions to make the system more practical. In engineering drawings, there can be various types of geometric elements, and therefore it is necessary to extend the system to geometric elements such as arcs, circles, and curves. To increase the efficiency of the reconstruction process, it would also be desirable to develop more systematic and reliable functions for evaluating the heuristics and to find more efficient heuristics. We believe that the algorithms presented here will be able to cope with the above problems if they are extended, the possibility of which we are now studying.

References

1. SASAKI, Y., ITOH, K. and SUZUKI, S. *Solid generation from orthographic views by nonlinear pseudo-Boolean algebraic solution (in Japanese)*, *J. IPS Japan*, **30**, 6 (1989).
2. IDESAWA, M. *A system to generate a solid figure from a three view, Bull. JSME*, **16** (1973), 216–225.
3. IDESAWA, M. *3-D model reconstruction and processing for CAE*, 8th ICPR (1986), 220–225.
4. EJIRI, E. et al. *A prototype intelligent robot that assembles objects from plan drawings, IEEE Trans. Comput.*, **C-21** (1972), 161–170.
5. MARKOWSKY, G. and WESLEY, M. A. *Fleshing out wire frames, IBM J. Res. Develop.*, **24**, 5 (1980), 582–587.
6. YOSHIURA, H. et al. *Top-down construction of 3-D mechanical object shapes from engineering drawings, IEEE Trans. Comput.* (Dec. 1984), 32–40.
7. HARALICK, R. M. et al. *Understanding engineering drawings, CG and Image Processing*, **20** (1982), 244–258.
8. NISHIHARA, S. and IKEDA, K. *Interpreting engineering drawings of polyhedrons*, 9th ICPR (1988), 869–871.
9. NISHIDA, J., ZHANG, S. and NISHIHARA, S. *Understanding three orthographic views by combinatorial search of faces (in Japanese)*, *Jour. of Japanese Society for AI.*, **6**, 1 (Jan. 1991).
10. DUDA, R. O. and HART, P. E. *Use of the Hough transformation to detect lines and curves in pictures, C. ACM*, **15** (1972), 11–15.
11. CLEMENT, T. P. *The extraction of line-structured data from engineering drawings, Patt. Recog.*, **14**, 1–6 (1981), 43–52.

(Received October 8, 1991; revised February 10, 1992)