MVA'94 IAPR Workshop on Machine Vision Applications Dec. 13-15, 1994, Kawasaki

RECONSTRUCTION OF 3D OBJECTS FROM THREE ORTHOGRAPHIC PROJECTIONS USING A DECISION-CHAINING METHOD

R. E. Marston and M. H. Kuo

Department of Computer Science, The University of Nottingham.
University Park, Nottingham NG7 2RD, UK.
ted@cs.nott.ac.uk, mhk@cs.nott.ac.uk

ABSTRACT

The automatic reconstruction of 3D objects from their 2D projections is briefly reviewed. Many algorithms have been published to tackle this problem in various contexts, the present work concentrates on the reconstruction of 3D mechanical parts from 2D orthographic projections. We present an algorithm that automatically reconstructs 3D polyhedral solids from their 3-view orthographic projections. All possible solutions can be enumerated when the engineering drawing is ambiguous.

Firstly, a 3D skeleton is established from a third-angle projection engineering drawing. Next, all candidate faces are traced in the skeleton. Then pseudo elements are detected and deleted using a decision-chaining method based on constructive solid geometry and the characteristics of engineering drawings. Finally, all true faces are assembled to form an oriented 3D object. The Moëbius rule is used to define face orientations and topological corrections are made using the Euler rule. The entire algorithm has been implemented in C on a UNIX system and experimental results are shown.

INTRODUCTION

The two main approaches to representing 3D objects reconstructed from 3-view orthographic projections use the volumetric model or the surface model. The volumetric approach usually uses CSG methods to represent objects and a translational sweep operation to construct primitives which it combines using set operators to form the final solid object. For example, Ho[1], and Meeran[2] give algorithms that reconstruct planar and curved face objects from 2D orthographic projections using such methods. Their main benefit is that they always produce solid objects. However they seem to require help from the user to identify the primitives' sweep heights or angles.

The surface model (B-rep) approach usually works from the bottom up, starting from vertices and edges, then finds faces to construct a solid. Some of the key methods using this approach are reviewed by Wang[3]. Wesley & Markowsky[4,5] derived a robust method from algebraic topology that ensured the results were always consistent. They overcame the serious problem of ambiguities in wireframe representations by listing all possible solutions. Several others have followed their approach and developed algorithms that can reconstruct planar faced 3D objects. Gujar and Nagendra[6] gave a practical algorithm based on the theory of Wesley & Markowsky, to deduce all 3D polyhedrons from given 2D engineering drawings. Lequette[7] and Dutta[8] included the ability to handle more complex objects, having curved faces.

However, all these methods still suffer from an exponential growth in complexity while reconstructing many of the simpler objects. The problem is that many pseudo or ghost elements can be generated during the back projection from 2D to 3D. The elimination of these can require a large search space and time. Chen et al[9], and Kim et al[10], proposed heuristic rules which reduce the combinational searching time. Unfortunately, some of the rules seem redundant or not always true when applied to certain examples. We follow their methods but use different rules and a decision-chaining algorithm to improve the early elimination of pseudo elements. Before describing our method, we briefly summarise the preceding steps that build a 3D skeleton of candidate vertices and edges and establish a list of candidate faces from which the pseudo elements must be deleted.

THE 3D SKELETON

An object is first recognised in terms of a wireframe of 3D vertices and edges (skeleton), derived from three 2D orthographic projections. This involves identifying triplets of 2D line junctions (one from each view), that have one coordinate in common in each pair of views. All 3D vertices of the object that lie at the intersection of at least three non-coplanar faces are found this way. Candidate edges are inserted into the skeleton between each pair of vertices that have a 2D line segment connecting their corresponding pair of 2D junctions in each view. Each edge is checked for intersection with the other edges and, when this occurs, additional vertices are inserted into the skeleton to subdivide the edges.

Figure 1 illustrates the process. Vertices 1 to 11 in the skeleton were derived from triplets of line junctions from the three views. Vertices 12 to 14 were found by intersecting the edges. The skeleton includes some pseudo elements, but they cannot be detected at this stage.

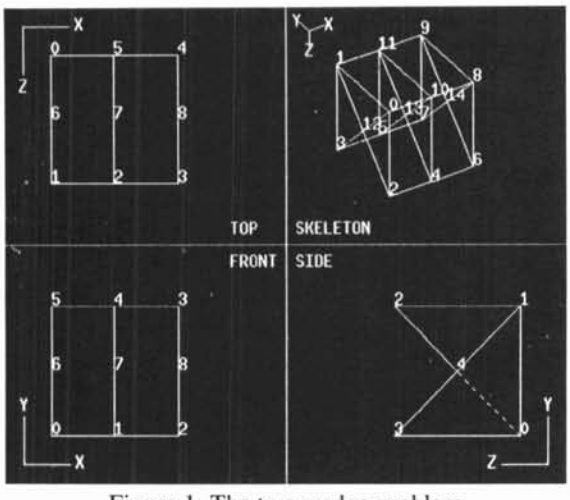

Figure 1. The two wedge problem

CANDIDATE FACES

We use a minimum-internal-angle searching method to trace all planar edge loops in the skeleton. Starting from a convex vertex (at the extreme left), we ensure that loops have no internal edges.

procedure trace_faces( )
  for (each vertex, v_i)
    for (each pair of adjacent edges, e_i,e_j) **do**
      n ← normal to plane, $e_i \times e_j$
      v_k ← vertex at far end of e_j
      while (($v_k \neq v_i$) **and** (e_j exists)) **do**
        e_k ← edge adjacent to v_k, coplanar to n,
          with minimum-internal-angle from e_j
        $e_j = e_k$; v_k = far end of e_k
      end while
      if ($v_k = v_i$) **then**
        list e_i,e_j,e_k, ··· is a candidate face loop
      end if
    next e_i,e_j
  next v_i
end procedure

Wesley and Markowsky[5] show that the true edges and faces of the 3D object are a subset of those found so far. Some pseudo elements may be detected when loops fail to close, but there are usually many more still to be found.

DECISION RULES

Our method of deleting pseudo elements uses the following rules:

[1] When an edge is adjacent to more than two faces, at most two faces can be true, the rest must be pseudo. (From the Moëbius rule)[7].

[2] When two faces intersect, only one of them can be true. (If they were both true, the intersecting edge would have four adjacent faces which contradicts the Moëbius rule).

[3] An edge which projects onto a dashed line in a view, but which has no candidate face to occlude it, is false. (From the definition of dashed lines on engineering drawings).

[4] When an edge is adjacent to only one face, both the edge and face are false. (Another application of the Moëbius rule).

[5] When an edge is adjacent to exactly two coplanar faces, the edge is false and the coplanar faces can be merged. (From the definition of lines on engineering drawings).

[6] If a true edge is adjacent to exactly two non-coplanar faces, then these faces are both true. (If either of the adjacent faces were false, it would contradict rule 4).

[7] A face that is coplanar and adjacent to a true face, when their shared edge projects onto a solid line and is not occluded in a view, is a false face. (The shared edge is needed to project onto the solid line because any other edge would be occluded by the true face. If the candidate face was true, rules 5 and 4 would apply and the shared edge would be deleted).

DECISION-CHAINING ALGORITHM

The decision-chaining method detects pseudo elements in the skeleton and candidate face lists. When undecided edges are found, assumptions are made using rules 1 and 2, then rules 1 to 7 are applied, until either all edges are decided, or no object is found. All possible objects are detected this way, without having to assemble them first. The recursive algorithm is summarised as follows:

procedure decision_chaining( )
  if (all edges obey Moëbius rule)
      and (no faces intersect) **then**
    a solution has been found, add it to the list
  else
    for (each undecided edge, e_i) **do**
      for (each adjacent pair of faces, f_j, f_k) **do**
        assume f_j, f_k are true;
        if (rules 1-7 change anything) **then**
          call decision_chaining( )
        end if
      next f_j, f_k
    next e_i
  end if
end procedure

Using Figure 2 as an example to illustrate the algorithm, there are sixteen candidate faces found by the minimum-internal-angle method. Initially edge e(6,8) is shared by candidate faces 5, 6 and 7. The algorithm first assumes faces 5 and 6 to be true. However, face 7 cannot be deleted because e(6,8) is a visible edge in the side view, which conflicts with rule 5. Next, faces 5 and 7 are assumed to be true. Now rule 1 says that face 6 is false and the chaining algorithm continues until it detects that faces 8 and 9 are false and a solution is found. Finally, the algorithm back-tracks to assume faces 6 and 7 are true. This fails to find a solution because face 5 cannot be deleted without the loss of visible edge e(4,7) from the side view. Thus a single solution is found, as shown in Figure 3.

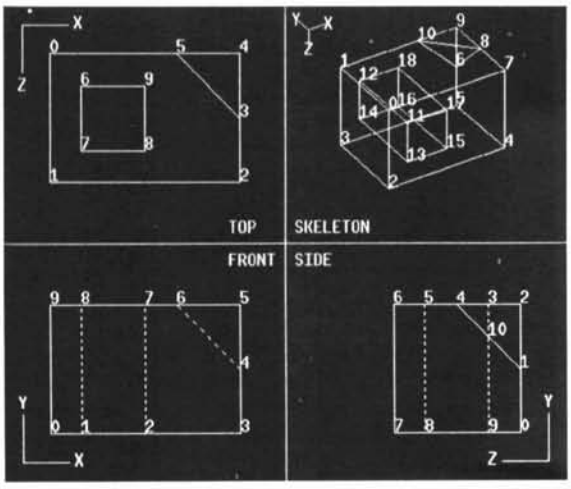

Figure 2. An example with three pseudo faces.

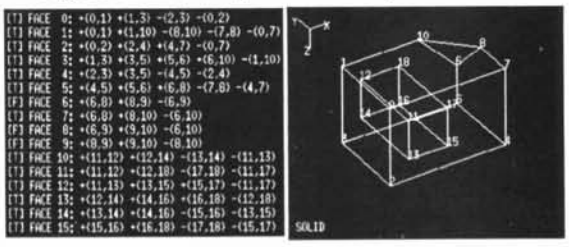

Figure 3. Candidate faces and object for Fig. 2.

CONTAINMENT AND ORIENTATION

Sometimes a 3D object includes holes whose faces are contained in another face. This is determined using a sum-of-angles containment test. In general, faces may contain each other in a hierarchical manner. Once the status of one face has been established the remaining can be deduced, they are alternately face and hole, ie, $f_i(face) \supset f_{i+1}(hole) \supset f_{i+2}(face) \supset \cdots$, where f_i is a true face when i is even, and a hole when i is odd. For example in Figure 3, f_{12} and f_{13} are "hole faces", contained in f_2 and f_3 respectively.

Face orientation consistency is maintained by using the Moëbius rule. A face adjacent to a vertex on the convex hull (eg. the extreme left-hand vertex) is used to decide which side is outside. From this, all other faces are oriented by comparing the direction of traversal of their edge loops. The sign of a face plane normal is reversed if adjacent face loops traverse adjacent edges in the same direction. For example, in Figure 3, face f_0 was initially traced using edges in the order v_0, v_1, v_3, v_2, but the adjacent face f_1 was traced in the order $v_0, v_1, v_{10}, v_8, v_7$. Applying the Moëbius' rule to edge e(0,1), results in face f_1's plane normal being reversed.

EXPERIMENTS AND SUMMARY

Further experimental results are given in Figures 4 to 9. The extracted solids for figures 4-6 are shown in Figure 7. Rule 5 applied to Figure 4 enabled true coplanar faces to be merged and the pseudo edges to be deleted. An example from Gujar and Nagendra[6], is given in Figure 5. The error of their 3-view engineering drawing is detected and a true 3D solid is successfully reconstructed by the decision-chaining algorithm. The famous two-wedge problem in Wesley and Markowsky's work is extended to three wedges in Figure 6. The efficiency of the present algorithm can be judged by the ambiguous 3-views of Figure 8 which has fourteen possible solutions, found in under one second by our algorithm.

It is observed that the decision-chaining algorithm does not establish any subparts during the reconstruction procedure. This reduces both the searching time and space, compared to earlier methods. The "hole faces" are detected by a sum-of-angles test and face orientation consistency is ensured using the Moëbius rule, which are seldom discussed in previous works.

The entire algorithm has been implemented in C on a UNIX system. Experimental results show that the newly developed algorithm has a higher performance than previous works, even though the reconstruction algorithm is limited to planar faced objects. We are currently developing the ability to handle curved surface objects and provide a practical system for the automatic reconstruction problem.

References

1. Ho Bin, "Inputting constructive solid geometry representations directly from 2D orthographic engineering drawings," *Computer-Aided Design*, vol. 18, no. 3, pp. 147-155, 1986.
2. S. Meeran and M. J. Pratt, "Automated feature recognition from 2D drawings," *Computer-Aided design*, vol. 25, no. 1, pp. 7-17, 1993 .
3. Weidong Wang and Georges G. Grinstein, "A Survey of 3D Solid Reconstruction from 2D Projection Line Drawings," *Computer Graphics Forum*, vol. 12, no. 2, pp. 137-158, 1993.
4. G. Markowsky and M. A. Wesley, "Fleshing Out Wire Frame," *IBM J. Res. Develop*, vol. 25, no. 5, pp. 304-311, 1980.
5. M. A. Wesley and G. Markowsky, "Fleshing Out Projections," *IBM J. Res. Develop*, vol. 25, no. 6, pp. 934-954, 1981.
6. U. G. Gujar and I. V. Nagendra, "Construction of 3D solid objects from orthographic views," *Comput. & Graphics*, vol. 13, no. 4, pp. 1-18, 1989.
7. Rémi Lequette, "Automatic construction of curvilinear solids from wireframe views," *Computer-Aided Design*, vol. 20, no. 4, pp. 171-179, 1988.
8. D. Dutta and Y. L. Srinivas, "Reconstruction of curved solids from two polygonal orthographic views," *Computer-Aided Design*, vol. 24, no. 3, pp. 149-159, 1992.
9. Zen Chen, Der-Baau Perng, Chii-Jang Chen, and Chu-Song Wu, "Fast reconstruction of 3D mechanical parts from 2D orthographic views

with rules," *INT. J. Computer Integrated Manufacturing*, vol. 5, no. 1, pp. 2-9, 1992.

10. C. Kim, M. Inoue, and S. Nishihara, "Understanding Three-View Drawings Based on Heuristics," *11th ICPR, Computer Vision and Applications Conf.*, vol. 1, pp. 514-517, 1992.

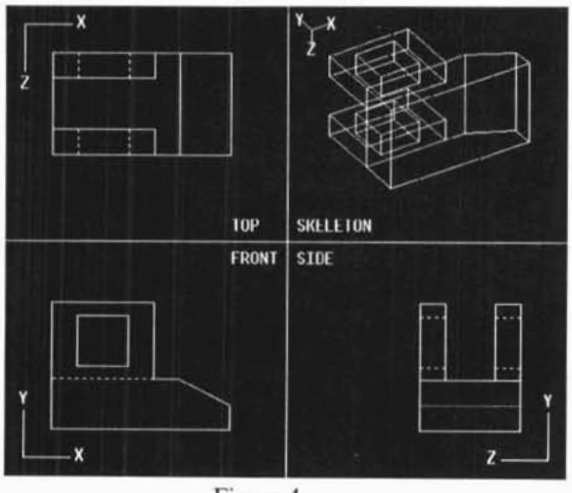

Figure 4.

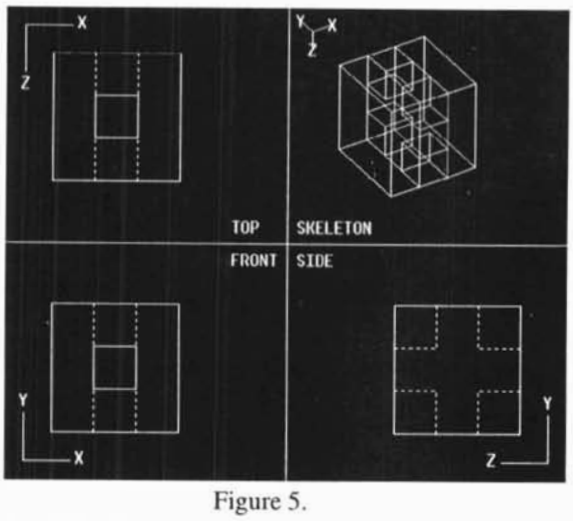

Figure 5.

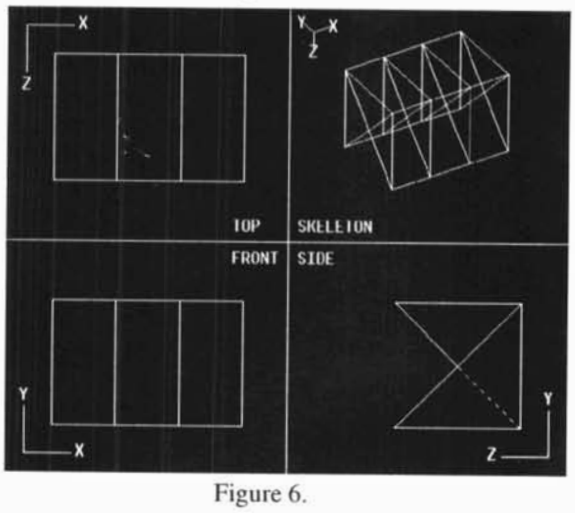

Figure 6.

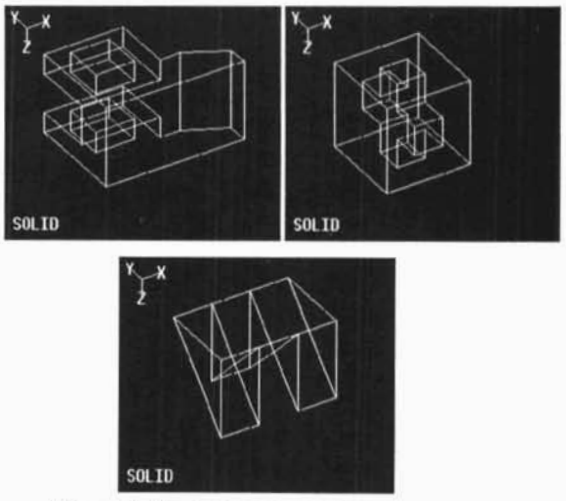

Figure 7. True objects for Figures 4 - 6

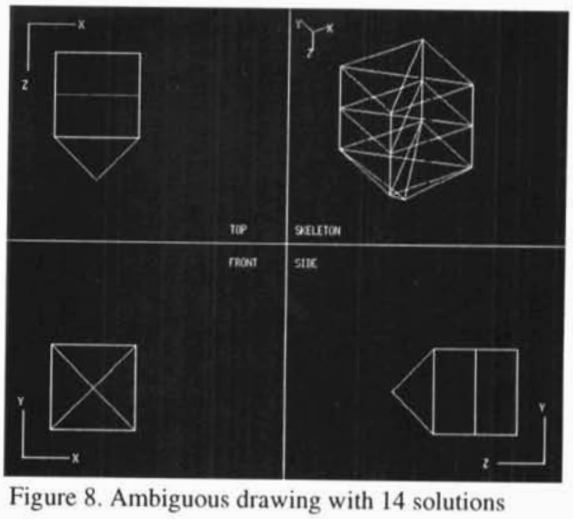

Figure 8. Ambiguous drawing with 14 solutions

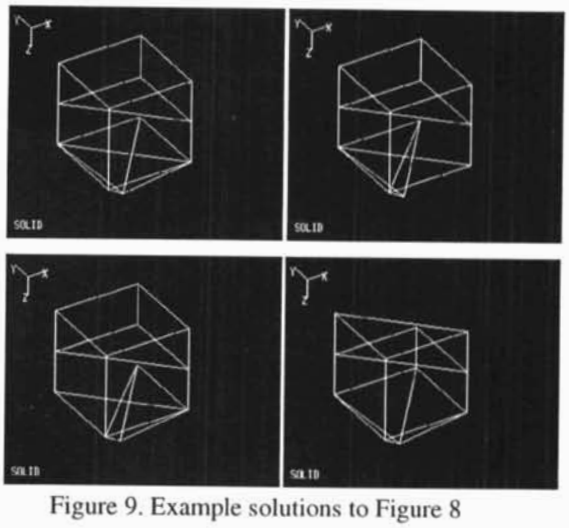

Figure 9. Example solutions to Figure 8