Computer-Aided Design, Vol. 30, No. 1, pp. 37–46, 1998

Printed in Great Britain
0010-4485/98/$19.00+0.00

ELSEVIER

PII: S0010-4485(97)00051-1

Decomposition of a 2D assembly drawing into 3D part drawings

Masaji Tanaka*†, Kenzo Iwama‡, Atsushi Hosoda§ and Tohru Watanabe§

This paper shows a method of automatically decomposing a 2D assembly drawing into 3D part drawings using a set of solid element equations. Solid elements are virtual elements that are constructed from orthographic views and which become the components of solid models of all of the parts described in the orthographic views. This method organizes the relationships between a 2D assembly drawing and solid elements as a system of solid element equations that can classify solid elements into true elements of some parts and false elements that do not actually exist in any parts. The method generates ways of solving the system of solid element equations and the combination of true elements. If there is more than one solution, the method can generate all of the solutions. This method was tested on various 2D assembly drawings. © 1998 Elsevier Science Ltd. All rights reserved

Keywords: assembly drawing, part drawing, orthographic view

INTRODUCTION

CAD/CAM systems have advanced automated design and manufacturing. In particular, solid modeling enables the manipulation of 3D models of parts. However, much information regarding design and manufacturing is still represented by 2D drawings. Various input devices provide engineers with choices in 2D drawings and improve flexibility for designers. In general, product design proceeds from conceptual to detailed design. Designers have been known to first draw 2D assembly drawings of products and then draw each 2D part drawing, despite the fact that there are methods for designing products by composing designed parts in CAD systems by superimposing layers of 2D part drawings. The 2D part drawings are transformed into solid models for CAM systems to generate NC programs and assembly planning among other applications. It is routine and takes a lot of time to decompose 2D assembly drawings into 2D part drawings and transform them into solid models. Therefore, these transformations are very often processed by operators other than designers.

This paper shows a method of automatically decomposing a 2D assembly drawing into 3D part drawings. A method is developed here to automatically construct solid models from orthographic views[1,2], and the method is extended for the decomposition. Though there are many methods to automatically construct solid models from orthographic views[3], nobody has attempted to construct solid models of more than one part from orthographic views to the knowledge of the authors. Nevertheless, such decomposition is important to designers for the following reasons:

(1) It takes a lot of time to draw each 2D part drawing from a 2D assembly drawing in proportion to the number of parts.
(2) If operators rather than designers process the decomposition, the operators may fail to correctly recognize each part of the 2D assembly drawing.

In this method, wireframe models, surface models and solid elements are constructed from orthographic views in order. A solid element is a closed region of faces that are elements of the surface models. Solid elements become the components of the solid models of all of the parts. The method presented here generates solid models of each part by classifying solid elements into elements of some parts and false elements that do not actually exist in any parts. If there is more than one solution, this method can generate all of the solutions. The method was implemented on a PC and tested on various 2D assembly drawings. Four examples are shown in this paper.

The domain of the 2D assembly drawings is limited to orthographic views consisting of front, top and side views, and cross-sectional views. Also, the types of faces are limited to planar, cylindrical, conical and spherical faces.

TERMINOLOGY

Drawing layout

The input to this method consists of orthographic views of an assembly that include cross-sectional views and part numbers, each of which is given to some surfaces in

*To whom all correspondence should be addressed.
†Okayama University of Science, Department of Mechanical Engineering, 1-1 Ridai-cho, Okayama 700, Japan.
‡Advanced Software Technology and Mechatronics Research Institute of KYOTO (ASTEM RI), Kyoto Research Park, 17 Chudoji Minami-machi, Shimogyo, Kyoto 600, Japan.
§Ritsumeikan University, 1916 Noji, Kusatsu, Siga 525, Japan.
Paper Received: 22 January 1997. Revised: 22 June 1997

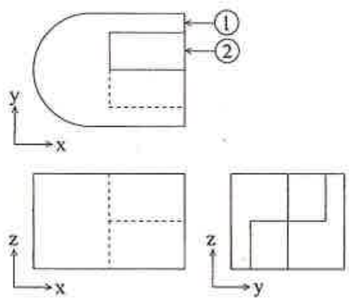

Figure 1 Example 1

the orthographic views. The coordinate system of the front, top and side views are x–z, x–y and y–z. *Figure 1* illustrates Example 1 which consists of two parts (part 1 and part 2).

Geometric elements

2D vertex and 2D edge

2D vertices are defined in 2D drawings as the following three types:

(1) crossing points of straight lines and arc lines;
(2) tangency points between straight lines and arc lines, and tangential points between arc lines;
(3) maximum or minimum points of arc lines in the vertical or horizontal directions in orthographic views, and the center points of circles.

The 2D edges are defined in the 2D drawings as the following two types:

(1) straight lines and arc lines existing between two 2D vertices;
(2) vertical or horizontal straight lines that correspond to 2D vertices selected from tangency points and maximum or minimum points described in the other views. Those edges are called silhouette edges.

Figure 2 illustrates all of the 2D vertices and 2D edges in Example 1.

Vertex and edge

A vertex is defined from the three 2D vertices in each view. Let a front 2D vertex be (fx,fz), a top 2D vertex be (tx,ty) and a side 2D vertex be (sy,sz). If $fx = tx$, $ty = sy$ and $fz = sz$, a (3D) vertex (fx,sy,fz) can exist as a virtual vertex of the solid.

An edge is defined from two vertices. Let the two vertices be $(x1,y1,z1)$ and $(x2,y2,z2)$. If a 2D edge $(x1, z1)$–$(x2, z2)$

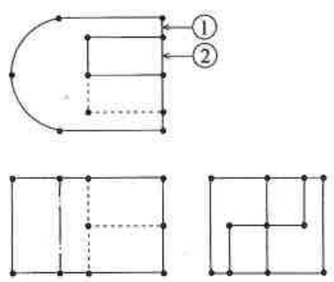

Figure 2 The 2D vertices and 2D edges in Example 1

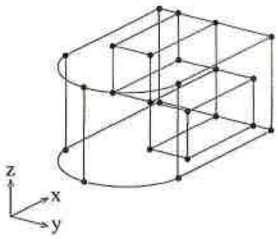

Figure 3 The vertices and edges in Example 1

exists, or $x1 = x2$, $z1 = z2$ and the 2D vertex $(x1,z1)$ exists in the front view, $(x1,y1,z1)$–$(x2,y2,z2)$ has the projection on the front view. In the same way, if an edge $(x1,y1,z1)$–$(x2,y2,z2)$ has projections on each view, the edge can exist as a virtual edge of the solid. *Figure 3* illustrates all vertices and edges obtained from *Figure 2*.

2D face and face

Except for the silhouette edges, a 2D face is a closed loop of 2D edges. *Figure 4* illustrates all of the 2D faces obtained from *Figure 2*. A face is a closed loop of edges. *Figure 5* illustrates all of the faces obtained from *Figure 3*.

Solid element

A solid element is a closed region of faces. *Figure 6* illustrates all of the solid elements ($S1$,$S2$,$S3$) obtained from *Figure 5*. As a result, part 1 is $S1$ and part 2 is $S2$. Therefore, $S3$ does not actually exist. A solid element that actually exists is called a true solid element, and a solid element that does not exist is a false solid element. $S1$ and $S2$ are true solid elements, but $S3$ is a false solid element.

SOLID ELEMENT EQUATION

Background

The method presented here classifies solid elements into true and false elements of some parts. The simplest way to classify them is to develop all of the combinations of the solid elements and project each of them onto 2D views to check if the projection results in the 2D assembly drawing. However, using this method, an explosion of the amount of processing information occurs. When the number of parts is M and the number of generated solid elements is N in a 2D assembly drawing, the number of all combinations of solid elements is $(M + 1)^N$. For example, if $M = 5$ and $N = 25$ (Example 4, see *Figure 21*), the number of combinations is

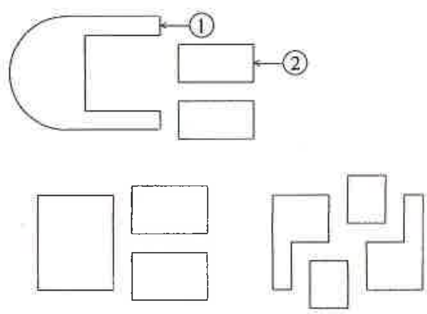

Figure 4 The 2D faces in Example 1

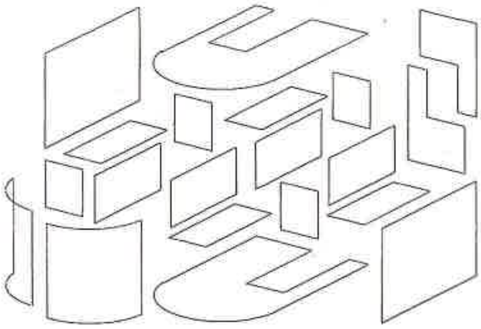

Figure 5 The faces in Example 1

$6^{25} \simeq 2.843 \times 10^{19}$. To avoid this information explosion, the relationships between a 2D assembly drawing and solid elements are organized as a set of solid element equations as described in the following.

Four conditions

When the direction of a view in a 2D assembly drawing is observed, a 2D face corresponds to the face of one solid element. For example, *Figure 7* illustrates the number of corresponding solid elements on each 2D face in Example 1. The following three conditions are obtained from the relationships of the correspondences.

Solid_edge_condition
If a 2D edge existing between two 2D faces is a solid line, the corresponding two faces of two solid elements must not be tangential to each other at the edge that corresponds to the solid 2D edge except where the two solid elements are not both elements of a part. If the two faces are the same, the solid element is a false element. If the two faces are tangential to each other, the two solid elements are neither both true nor both elements of a part. When the two solid elements are *Sx* and *Sy*, the relationship is expressed as a *Solid_edge_condition*, $Sx \times Sy$.

Dotted_edge_condition
If a 2D edge existing between two 2D faces is a dotted line, the corresponding two faces of two solid elements must be tangential to each other at the edge that corresponds to the dotted 2D edge. Since the two faces are tangential to each other, the two solid elements are both elements of a part or both false. If the two faces are not tangential to each other, the two solid elements are not both true. For a dotted 2D edge to exist, solid elements must exist that make edges corresponding to the dotted 2D edge. When the two solid elements are *Sx* and *Sy*, the relationship is expressed as a *Dotted_edge_condition*, $Sx - Sy$.

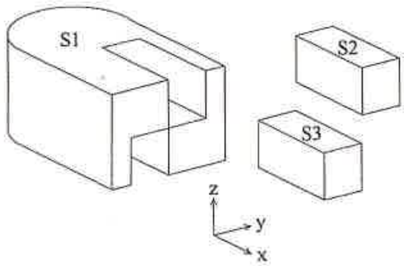

Figure 6 The solid elements in Example 1

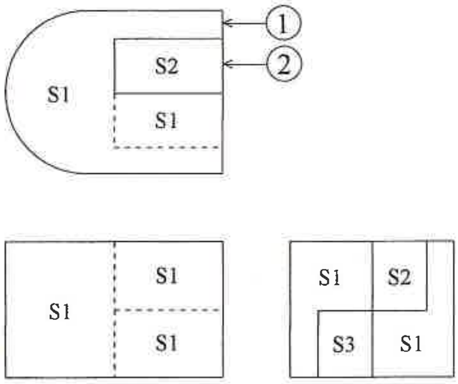

Figure 7 The number of solid elements on each 2D face in Example 1

Existence_condition
For each 2D edge, there must be one true solid element in all of the solid elements that have edges corresponding to one 2D edge. In addition, for each 2D face marked by some part numbers, there must be one solid element in all of the solid elements that have faces corresponding to one 2D face marked by the part number.

A system that represents all of the relationships of solid elements is called the '×' and '−' system of solid element equations. The method classifies solid elements using solid element equations and may also generate a part that consists of two or more separate solids by applying the three conditions *Solid_edge_condition*, *Dotted_edge_condition* and *Existence_condition*. Therefore, the following condition is added.

Mass_condition
Solid elements that form a part cannot be separated into two or more solids, nor can the solid elements be connected simply through vertices or edges. This condition is called the *Mass_condition*.

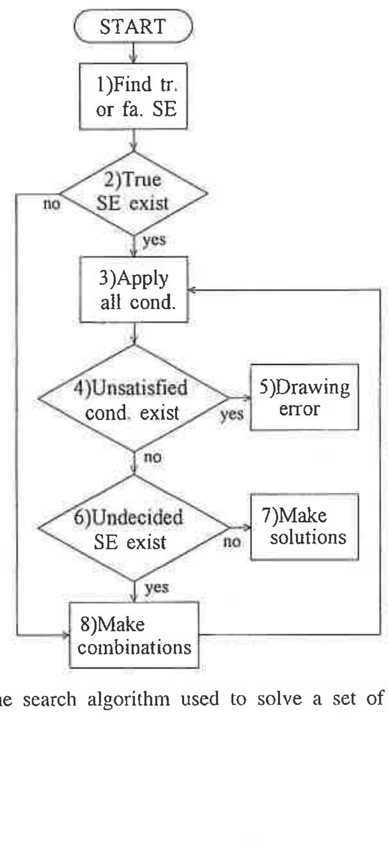

Figure 8 The search algorithm used to solve a set of solid element equations

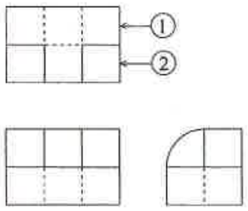

Figure 9 Example 2

Three conditions related to 2D edges are necessary to establish the correspondence between a 2D assembly drawing and the projected drawing of solution solids. When one picks up any solid line segment in any view, one will find the condition (*Solid_edge_condition*) between two solid elements. The same can be said of any dotted line segment (*Dotted_edge_condition*) and of any line segment (*Existence_condition*). *Mass_condition* is a necessary condition to generate real solids. On the other hand, steps 6–8 of the search algorithm described below provide a sufficient condition for the 2D edges in 2D drawings. Since the algorithm generates all of the possible combinations of the true solid elements, every edge in each combination has a corresponding 2D edge (solid or dotted) in the original 2D drawing.

In Example 1, $S1 \times S2$ is obtained from the top view, and $S1 \times S2$, $S1 \times S3$ are obtained from the side view. As a result, $S2 \times S1 \times S3$ is the solid element equation. It is found that *S1* is an element of part 1, and *S2* is an element of part 2 due to the existence of two 2D faces marked part 1 and part 2. *S3* is false due to $S1 \times S3$ and the *Mass_condition*. As a result, part 1 is *S1* and part 2 is *S2*.

Search algorithm

Figure 8 illustrates the search algorithm used here to solve a system of solid element equations. The input is a set of conditions that consist of every *Solid_edge_condition* from the solid 2D edges, every *Dotted_edge_condition* from the dotted 2D edges and every *Existence_condition* from the 2D edges and part numbers on some 2D faces (the proposed program gets the part numbers). The abstract of the algorithm is as follows:

(1) Find true or false solid elements.

Find solid elements (SE) that are trivially false by *Solid_edge_conditions* and *Dotted_edge_conditions*, and find true solid elements by *Existence_ conditions*.

(2) True solid elements exist.

If true solid elements do not exist, develop combinations of all solid elements.

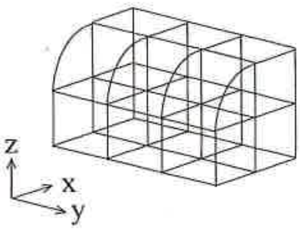

Figure 10 The wireframe model in Example 2

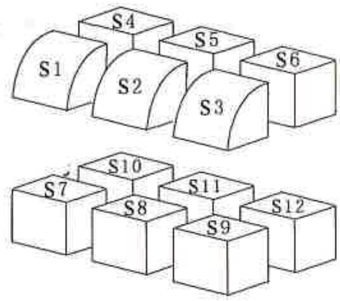

Figure 11 The solid elements in Example 2

(3) Apply all conditions.

Apply *Solid_edge_conditions* and *Dotted_edge_conditions* to find true or false solid elements. After the application of these two kinds of conditions, select a false element and check to see if there is a true element where the true element has an edge of the false element. Finally, apply *Mass_condition* to check and see if a part includes more than one separated solid.

(4) Unsatisfied conditions exist.

If unsatisfied conditions exist, the drawing must have mistakes.

(5) Drawing error.

Output a message 'Drawing error'.

(6) Undecided solid elements exist.

If solid elements that are undecided in terms of belonging to parts or truthfulness exist, combine them.

(7) Make solutions.

Solutions are formed by connecting the solid elements in each part.

(8) Make combinations.

If the number of parts is M and the number of undecided solid elements is N, make $(M + 1)^N$ combinations.

Undecided solid elements cause the generation of multiple solutions. Therefore, if designers introduce cross-sectional views into 2D assembly drawings to reduce the number of cases where one solid element can be in more than one part, the number $(M + 1)^N$ can be minimized.

EXAMPLES

An example of two parts

Figure 9 illustrates Example 2 which consists of two parts.

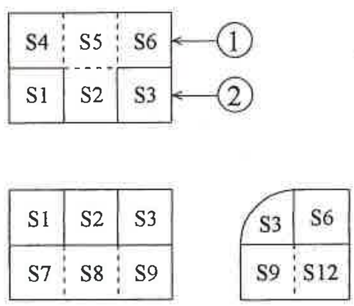

Figure 12 The number of solid elements on each 2D face in Example 2

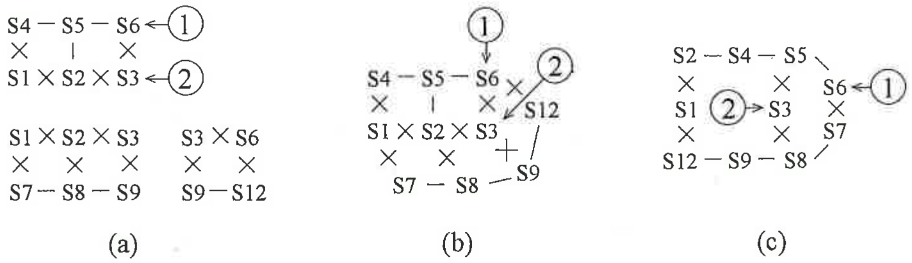

Figure 13 The solid element equation in Example 2

Figures 10–12 illustrate a wireframe model, solid elements and the number of solid elements on each 2D face. *Figure 13a* illustrates the solid element equations in each view. *S6* is marked by part 1, and *S3* is marked by part 2 on the top view. The three equations can be connected to form one equation as in *Figure 13b* by *S1*, *S2*, *S3* and *S6* which are common elements in the three views. The equation can be further translated into that in *Figure 13c* (for example, either *S1* × *S2* or *S1* × *S4* can be removed because of *S2* − *S5* − *S4*).

The solution is calculated as shown below. Since there are top and horizontal 2D edge of the 2D face corresponding to *S2*, and bottom and horizontal 2D edge of 2D face corresponding to *S9* in the front view, *S2*, *S5* and *S9*, *S12* are true. Therefore, *S2*, *S4*, *S5*, *S6*, *S7*, *S8*, *S9*, *S12* are true. Since *S6* is true, *S2*, *S4*, *S5*, *S6* are elements of part 1. So *S7*, *S8*, *S9*, *S12* are elements of part 2 because of *S6* × *S7*, and *S1*, *S3* are false. *S10* and *S11* are undecided solid elements. Each of them is an element of part 1 or part 2 or is false. When all conditions are applied to $3^2 = 9$ combinations, four solutions are obtained as in *Figure 14*.

Actually designers draw 2D assembly drawings as unambiguously as possible. In Example 2, designers can draw cross-sectional views. When section *AA'* is added as in *Figure 15a*, the relationships at section *AA'* are obtained as in *Figure 15b* and *c* by cutting solid elements at the cross-section. As a result, *S10* is false and *S11* is an element of part 1. Therefore, *Figure 14a* is the solution.

A linear table

Figure 16 illustrates Example 3 which shows a linear table. *Figures 17–19* illustrate a wireframe model, solid elements and the number of solid elements on each 2D face. The solid element equations are *S2* × *S5*, *S3* × *S8* and *S2* × *S11* that are obtained from *Dotted_edge_conditions*. When *Existence_conditions* are applied to the solid elements, it is found that *S1* is an element of part 1, *S9* is an element of part 2, *S3* is an element of *part 3*, *S6* is an element of part 4, *S2* is an element of part 5, and *S4*, *S7*, *S10*, *S12* are true.

Since *S4* only makes contact with *S5* by a face, *S5* is not false by applying the *Mass_condition*. Since *S5* makes contacts with *S4* and *S6*, *S4* and *S5* are elements of part 4. Since *S11* is an element of part 1 or false, *S10* and *S12* are elements of part 1. As a result, the undecided solid elements are *S7*, *S8* and *S11*.

S7 can be an element of parts 2, 3 or 4. *S8* can be an element of parts 2 or 4 or is false. *S11* can be an element of part 1 or is false. Therefore, $3 \times 3 \times 2 = 18$ combinations are candidates for the solutions as (*S7*,*S8*,*S11*) = {(2,0,0), (2,0,1), (2,2,0), (2,2,1), (2,4,0), (2,4,1), (3,0,0), (3,0,1), (3,2,0), (3,2,1), (3,4,0), (3,4,1), (4,0,0), (4,0,1), (4,2,0), (4,2,1), (4,4,0), (4,4,1)} (false is zero). Since the combinations of (2,0,0), (2,0,1), (2,4,0), (2,4,1), (3,4,0) and (3,4,1) are not consistent with the *Mass_condition*, the remaining 12 combinations become candidates of the solutions.

However, if (*S7*,*S8*,*S11*) equals (2,2,0) or (2,2,1), it is impossible to assemble parts 2 and 3. As a result, 10 solutions are obtained as in *Figure 20*. In this way, assembly knowledge may reduce the number of possible solutions. Unlike knowledge about the functions of parts, some assembly knowledge is already formalized to generate assembly sequences (for example, see Refs [4,5]), and this knowledge can be integrated into the method.

An example of five parts

Figure 21 illustrates Example 4 which consists of five parts. *Figures 22–24* illustrate a wireframe model, solid elements and the number of solid elements on each 2D face. *Figure 25* illustrates the solid element equations in each view. However, there are a few false elements such as *S18* × *S18* in the top view. These trivial false elements are *S10*, *S12*, *S13*, *S14*, *S18* and *S21*. After those false elements are removed, *Figures 24 and 25* change into *Figures 26 and 27* (*Figure 27b* is translated from *Figure 27a*).

It is found that *S1*, *S2*, *S3*, *S9*, *S11*, *S17*, *S19* are elements of part 1, *S4*, *S6*, *S7*, *S8* are elements of part 3, *S15* is an element of part 4, *S22* is an element of part 2, *S25* is an element of part 5, and *S20*, *S24* are true by the *Existence_conditions*. Since *S5* only makes contact with *S2*, *S4* and *S6*, *S5* is false, and *S25* is part 5, but *S16* is not an element of part 3. These three relations are found by *Solid_edge_conditions*.

As a result, *S16*, *S20*, *S23* and *S24* are undecided

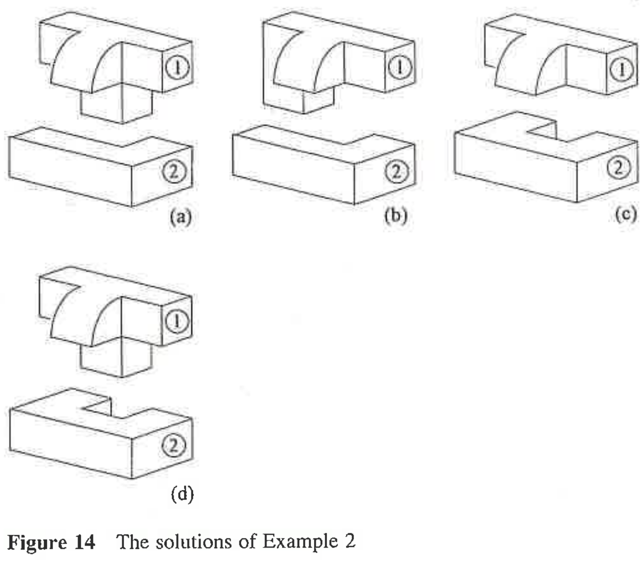

Figure 14 The solutions of Example 2

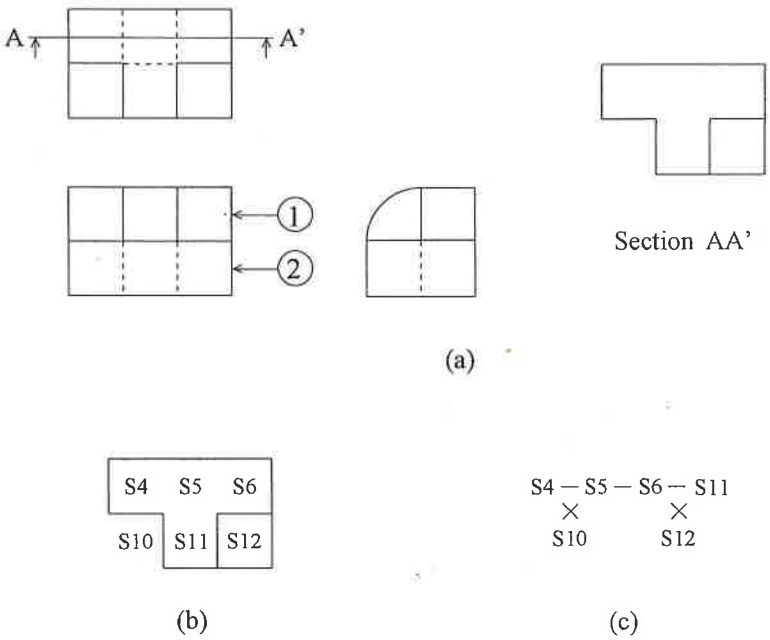

Figure 15 The usage of a cross-sectional view in Example 2

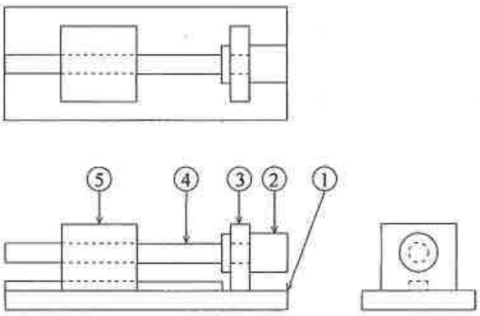

Figure 16 Example 3

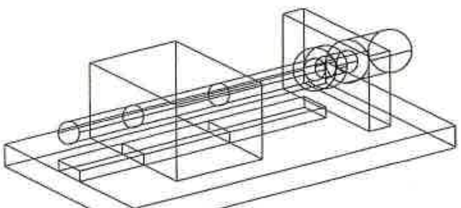

Figure 17 The wireframe model in Example 3

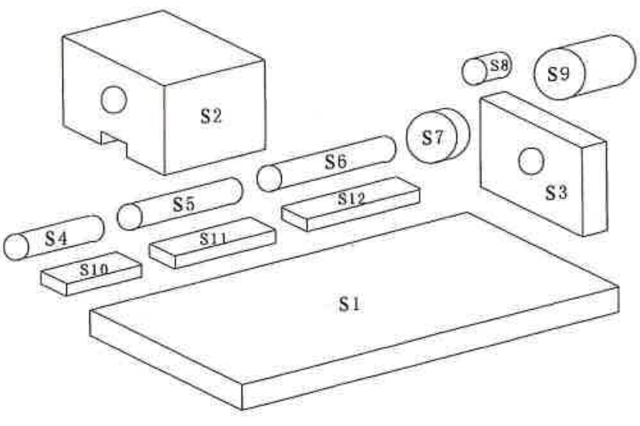

Figure 18 The solid elements in Example 3

elements. Thus, $4 \times 4 \times 5 \times 4 - 60 = 260$ combinations are candidates of the solutions (reduced by 60 due to *S16* × *S24*). By applying all of the conditions to the combinations, five solutions are obtained as in *Figure 28*.

DISCUSSION

The method proposed here is based on research of automatically constructing solid models from orthographic views of one solid. The research was initiated by Idesawa [6]. He made a wireframe model and a surface model, and introduced the generation method of solids. Wesley and Markowsky [7] proposed virtual blocks which are called solid elements. Though their method is applied only to planar faces, Sakurai and Gossard [8] extended the method on order to apply it to curved faces. A method to construct solids using solid element equations [1,2] was also proposed by the authors. Though this method uses solid elements, it is based on a new idea described in the following paragraph.

If the lines drawn are all straight and vertical or horizontal in orthographic views, it is possible to translate these three views into three lattice faces, whose outlines are all

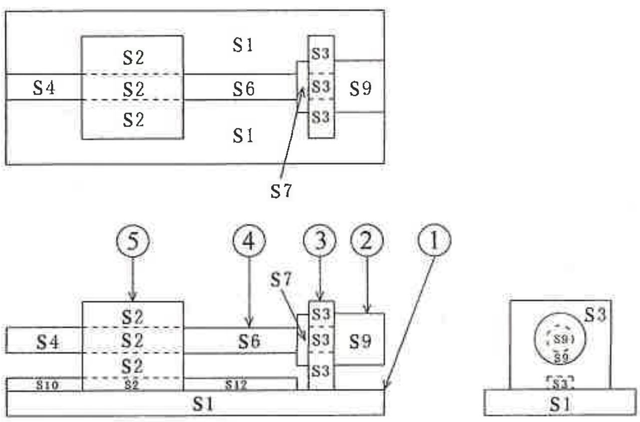

Figure 19 The number of solid elements on each 2D face in Example 3

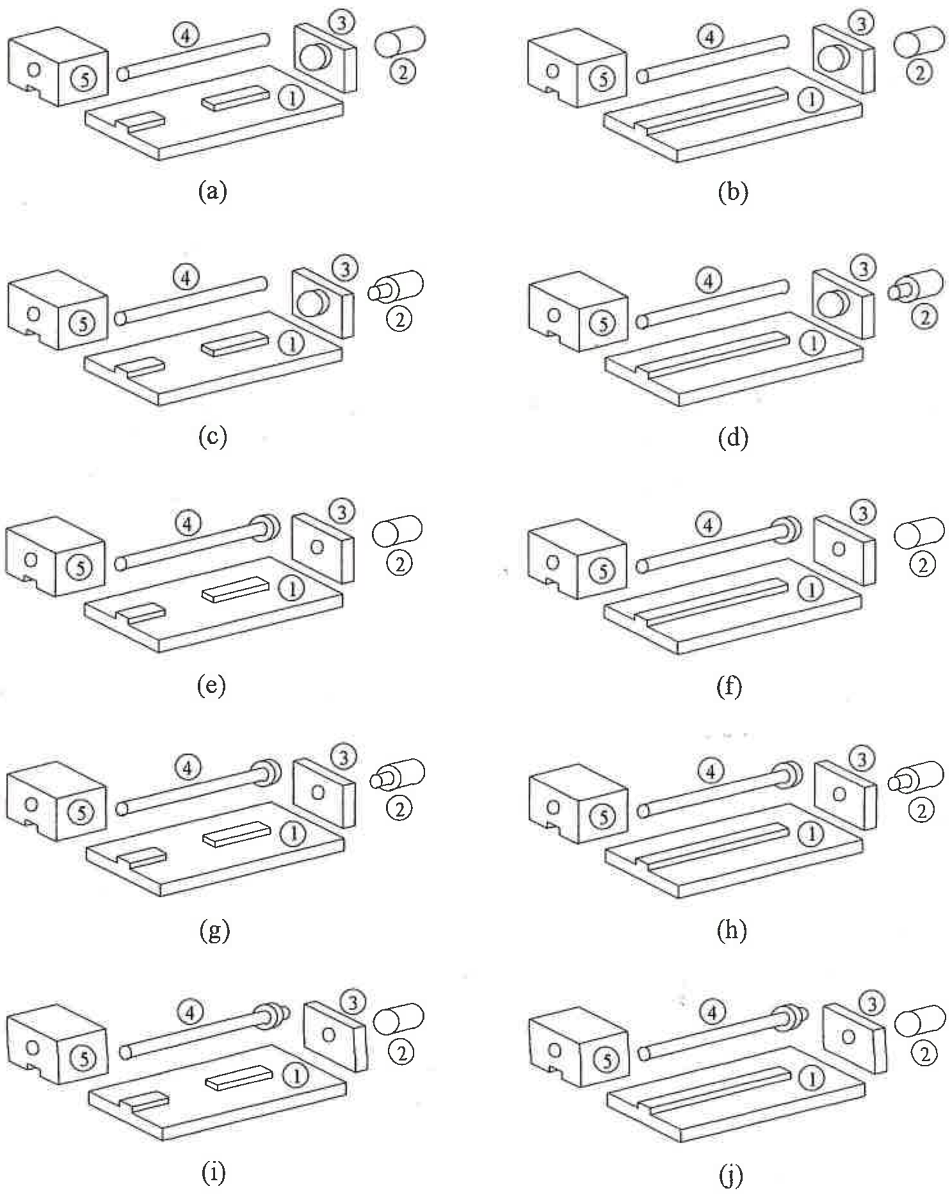

Figure 20 The solutions of Example 3

rectangles, by extending the lines drawn. A matrix of cubic elements is made from those three lattice faces. The relationships between the 2D faces and the faces of the cubic elements are trivial, and the solutions are simply obtained as combinations of the cubic elements.

Figure 29 shows an example of the application of a matrix of cubic elements. *Figure 29a* illustrates one part drawing, which can be translated into three lattice faces as in *Figure 29b*. These faces form the matrix of cubic elements as in *Figure 29c*. From the figure, one is able to observe that both *S3* and *S4* cannot be true in the top view, and that both *S5* and *S6* are true or false in the front view. In this way, one can extract all of the relationships between the 2D faces and the cubic elements that are described as in *Figure 29b* and *d* by using the 2D edge conditions. *Figure 29d* can be translated into *Figure 29e*. By applying *Existence_conditions*, *S1*, *S2*, *S4*, *S8* are false,

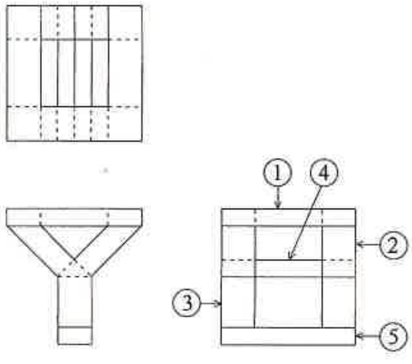

Figure 21 Example 4

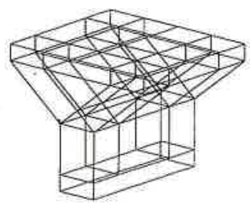

Figure 22 The wireframe model in Example 4

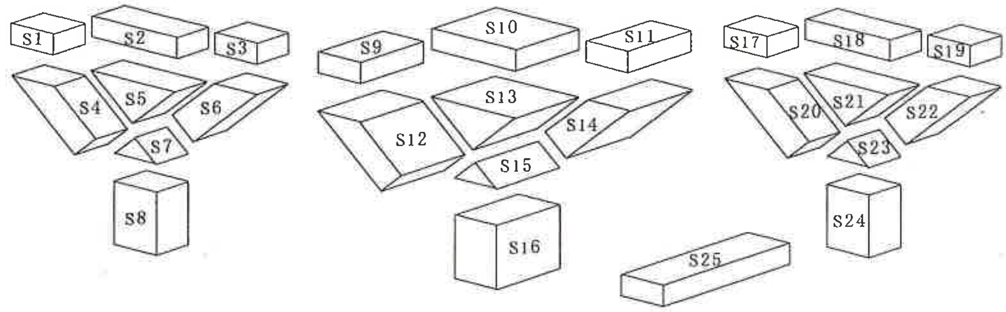

Figure 23 The solid elements in Example 4

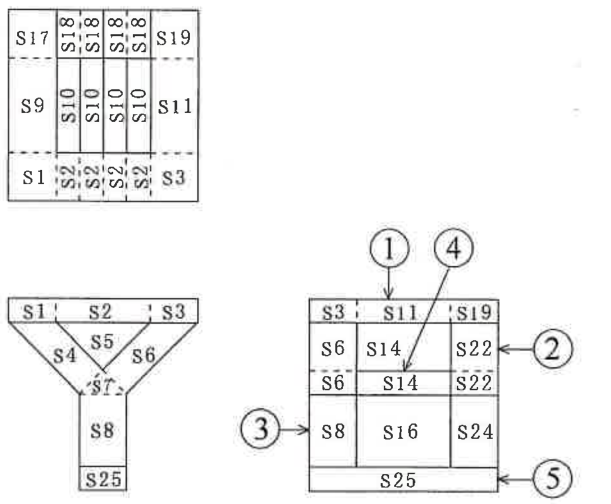

Figure 24 The number of solid elements on each 2D face in Example 4

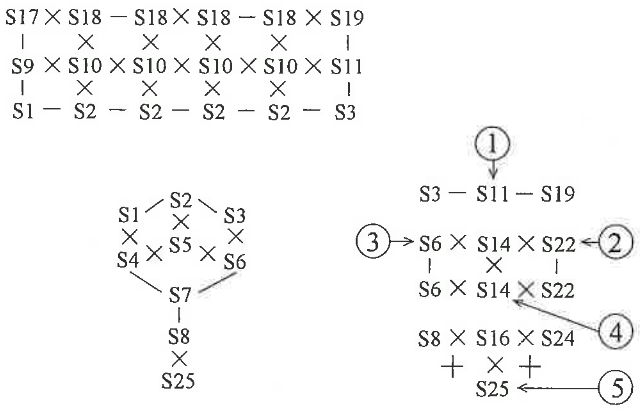

Figure 25 The solid element equation in Example 4

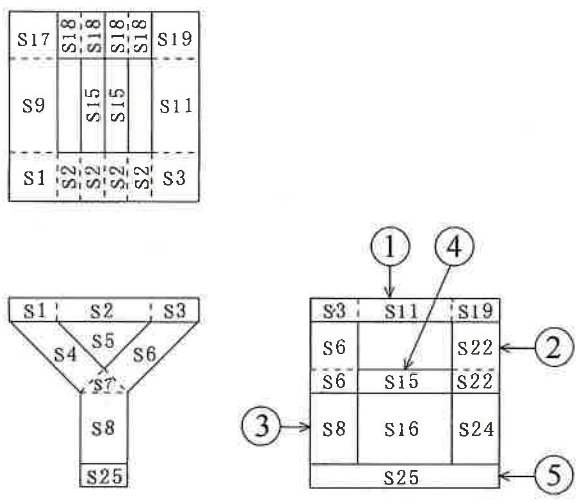

Figure 26 Modified *Figure 24*

and *S5*, *S6* are true. Since *S1* is false, *S3* is true by *Existence_conditions*. *S7* is true by a *Dotted_edge_condition*. As a result, the solution is constructed of *S3*, *S5*, *S6*, *S7* as in *Figure 29f*. However, this method can be applied only to orthographic views that can be translated into lattice faces. Therefore, solid elements are adopted to deal with various shapes of solids.

Since solid element equations directly represent the semantics of every line segment in every orthographic view, it is simple to deal with the cases where more than one part is drawn and/or cross-sectional views are drawn in the orthographic view. However, it is not clear whether or not previously proposed methods can be easily extended to be applied to construct more than one part from orthographic views with cross-sectional views.

Solid element equations can also quickly determine true or false solid elements and find solutions because combinations of solid elements are employed with only

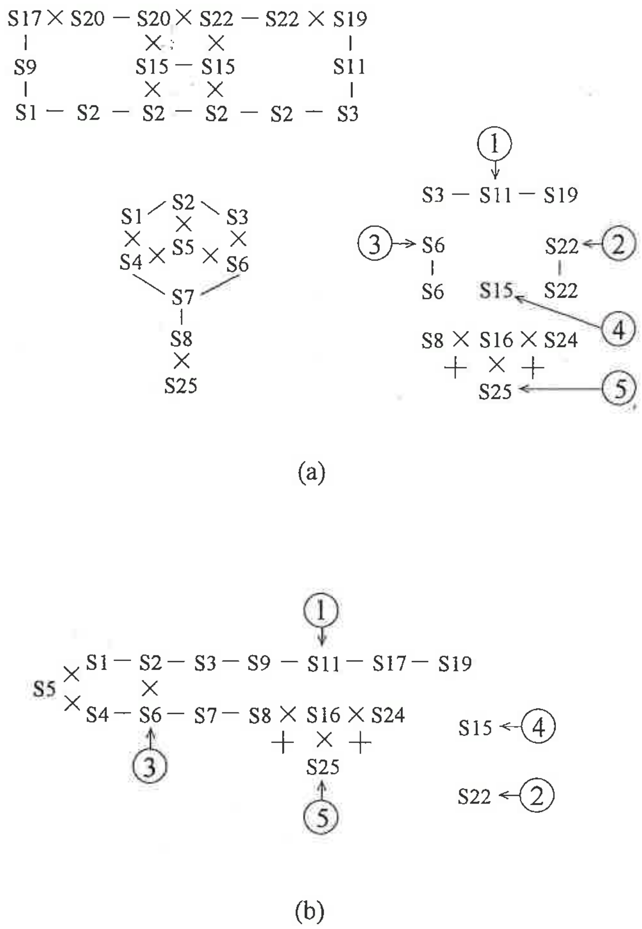

Figure 27 The modified solid element equation in Example 4

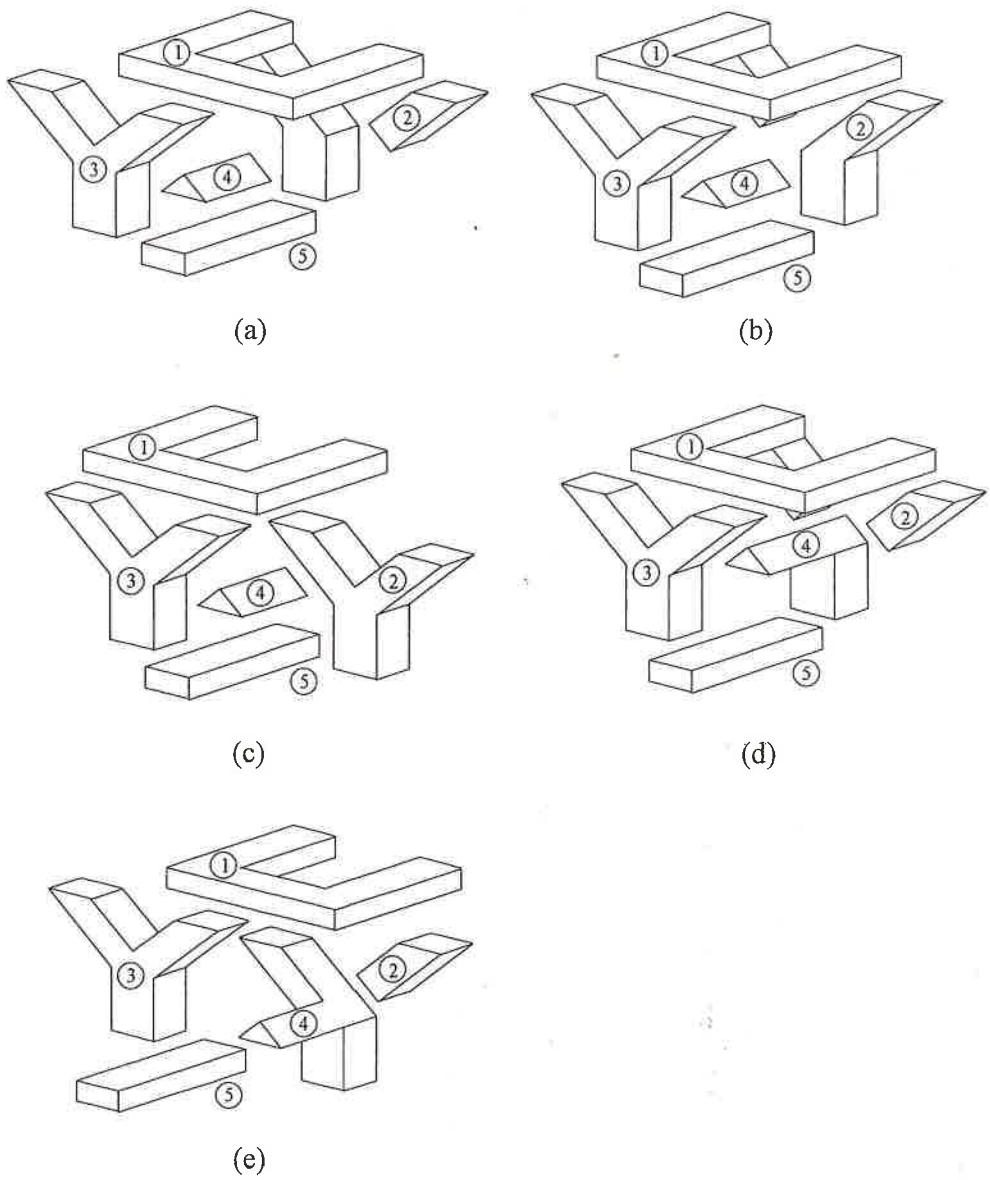

Figure 28 The solutions of Example 4

undecided solid elements. Previously proposed methods for constructing solid models from orthographic views that use solid elements basically employ a mechanism of searching for suitable combinations of all of the elements (for example, see Refs [7,9–11]).

Sub-views such as cross-sectional views as in Example 2, and assembly knowledge as in Example 3 are effective for reducing the number of solutions. Additionally, knowledge of design is also effective. In Example 3, if *S10*, *S11* and *S12* represent a rail, it is found that *S11* is not false. Therefore, the use of design knowledge is a subject for future research.

CONCLUSION

A method was developed for automatically decomposing a 2D assembly drawing into 3D part drawings by a system of solid element equations. Every solid element is classified into one of three types: a true element of a part, a false element and an undecided element. Though the amount of processing information to exponentially classify all the solid elements grows in proportion to the number of undecided elements, the solid element equations can minimize the number of undecided elements. The proposed method was tested on various 2D drawings, and it was shown that this method can generate all of the possible solutions. Cross-sectional views and assembly knowledge were shown to be capable of reducing the number of solutions.

REFERENCES

1. Tanaka, M., Construction of solid models from orthographic views by solid element equations. *IPSJ Trans.*, 1993, **34**(9), 1956–1966.
2. Tanaka, M., Iwama, K. and Watanabe, T., A method of constructing solid models from orthographic views by using solid elements. In *Proc. of the 1st ISICIMS 94*, pp. 219–224.
3. Nagendra, I. V. and Gujar, U. G., 3-D objects from 2-D orthographic view — A survey. *Comput. & Graphics*, 1988, **12**(1), 111–114.
4. Homem de Mello, L. S. and Lee, S., *Computer-Aided Mechanical Assembly Planning*. Kluwer Academic Publishers, 1991.
5. Tanaka, M. and Iwama, K., A method of generating assembly plans

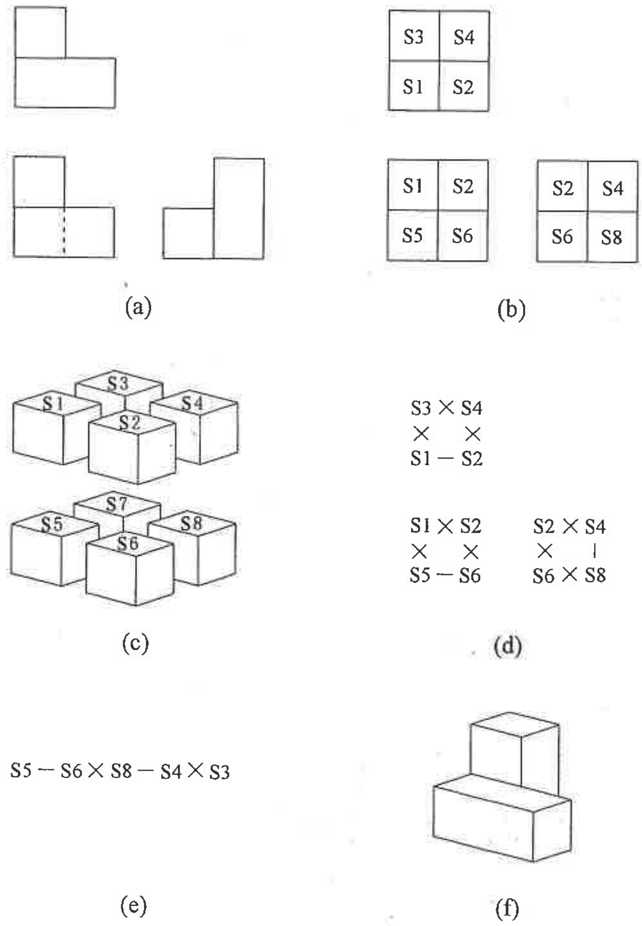

Figure 29 An example of an application of a matrix of cubic elements

from assembly drawings by disassembly equations. *IPSJ Trans.*, 1994, **35**(9), 1912–1921.

6. Idesawa, M., A system to generate a solid figure from a three view. *Bull. JSME*, 1973, **16**(February), 216–225.
7. Wesley, M. A. and Markowsky, G., Fleshing out projections. *IBM J. Res. & Develop.*, 1981, **25**(6), 934–954.
8. Sakurai, H. and Gossard, D. C., Solid model input through orthographic views. *Computer Graphics*, 1983, **17**(3), 243–252.
9. Gujar, U. G. and Nagendra, I. V., Construction of 3D solid objects from orthographic views. *Comput. & Graphics*, 1989, **13**(4), 505–521.
10. Yan, Q., Philip, C. L. and Tang, Z., Efficient algorithm for the reconstruction of 3D objects from orthographic projections. *Compter-Aided Design*, 1994, **26**(9), 699–717.
11. Masuda, H. and Numao, M., A cell-based approach for generating solid objects from orthographic projections. *Compter-Aided Design*, 1997, **29**(3), 177–187.

Masaji Tanaka is a researcher at Okayama University of Science in Okayama, Japan. He graduated from Saga University in 1986. He worked for the Ricoh Company from 1986 to 1990, and ASTEM RI from 1990 to 1996. His main research topics are geometric processing and database systems in CAD/CAM.

Dr Kenzo Iwama is a senior researcher at ASTEM RI in Kyoto, Japan. He received his undergraduate degree from the University of Tokyo in 1974, and his Ph.D. degree in Computer Science from Pennsylvania State University in 1988. His current research interests include algorithm development in CAPP, CACSD, CAD and robotic motion planning.

Atsushi Hosoda received his B.S. in Computer Science from Ritsumeikan University in 1995. Currently, he is completing a master course in Computer Science at Ritsumeikan University, where he is studying the development of computer tools to realize Concurrent Engineering, approaching the problem from the view of CAD systems.

Tohru Watanabe is a Professor of Computer Science and Systems Engineering at Ritsumeikan University. He graduated from Kyoto University in 1965 and acheived his Doctor of Engineering from Kyoto University. His main research topics are flexible automation, CAD/CAM, and the application of neural networks and genetic algorithms to manufacturing scheduling and robot control.