On automatic recognition of 3D structures from 2D representations

B Aldefeld

One possible way of specifying the geometry of mechanical parts in CAD follows a two-step procedure: input of a graphics-oriented 2D representation corresponding to views and cuts; and interpretation of this data to give the explicit 3D description. This paper is concerned with the problem of automating the interpretation step. The method of solution presented is based on viewing a part as consisting of several elementary objects and on recognizing these from their specific patterns in the 2D representation. A detailed algorithm is given for the case of three-view representations of parts whose constituent elementary objects are of the uniform-thickness type.

drafting, interpretation, reconstruction

At present, there are two main approaches in CAD considered for defining the geometry of mechanical parts. The first of these is 3D oriented and is based on description and manipulation of elementary volumes, with which, by such operations as union, difference and intersection, complex structures can be defined. The second approach is 2D oriented and is based on the analogy with conventional engineering drawings. The pieces of data of the internal part description correspond to the graphical elements (straight lines, arcs, etc) of views and cuts depicted in a drawing. A geometric part description, whether 2D or 3D oriented, is typically stored in a database and shown on a graphical display.

In this paper, we consider the 2D-oriented approach, which is preferred in some CAD systems, eg the PHILIKON system[1], because it has several advantages as far as user convenience is concerned. A disadvantage is, however, that although the 3D structure of a part is completely defined by the 2D data, this structure is not explicitly represented as would be desirable for further automatic processing. To build an explicit 3D description requires a second step, the 2D–3D reconstruction step, in which the 2D data is interpreted with regard to their 3D semantics.

Sufficiently general methods for automating this step have not yet been developed despite its great importance for engineering practice. Work on this problem has so far concentrated on the class of polyhedral parts[2–10], with either no capabilities at all, or only limited capabilities for more general shapes. Further investigation aimed at extending shape-modelling possibilities is therefore needed.

This paper presents a new approach to the problem, as a further step towards the desired goal of a general solution. The main underlying philosophy is to view a complex part as being composed of elementary objects belonging to a set of predefined classes, and to recognize these elementary objects by making use of the knowledge about the class-dependent patterns of their 2D representations. This approach has the potential for dealing with a variety of non-polyhedral shapes. In selecting and elaborating the techniques used in this approach, the close analogy with problems from the field commonly known as 'scene analysis', 'picture interpretation', or 'computer vision'[11] served as a guiding principle. This analogy derives from the fact that 2D–3D reconstruction can be regarded as interpretation of an artificial scene, namely an engineering drawing, which is preprocessed by an ideal optical front-end analyser.

Throughout this paper we assume that the 2D input data correspond to a three-view engineering drawing with front, side and top views given in orthographic projection. Further, we impose the simplifying restriction that the 2D representation of each elementary object is complete in the following sense: the input data must include, for each of the elementary objects, the projections of all its edges and the projections of all boundaries where the line of sight is tangent to the surface. This means that each elementary object must be represented as if it were an isolated body, except that the drawing mode of its lines may be changed from solid (visible) to dashed (invisible) or *vice versa* due to interference with other elementary objects. In real engineering drawings, not all of those lines are always given, since edges or tangential surfaces may vanish where two elementary objects are in contact. To handle such cases, we require that appropriate auxiliary lines be specified.

In other respects, we allow a 2D representation to be on its lowest possible semantic level, ie we will not require any information to be present about relationships between the basic graphical elements. Thus we consider the worst case in which there is only a collection of straight lines, arcs, etc stored in arbitrary sequence in the database, but no higher-level structures, such as closed curves consisting of several graphical elements. Also, we do not impose any restrictions on how the mapping between the internal computer representation and the corresponding engineering drawing is performed. For instance, a circle may be specified as a single circle or as two arcs.

For brevity the following notation will be used: the term 'primitive' denotes the basic elements of the 2D representation, such as straight lines, arcs, and circles. The term 'object' is used for the 3D elementary objects of which a complex part is composed. An object can be a solid body or it can be a cavity.

Information Systems Department, Philips Research Laboratory Hamburg, 2000 Hamburg 54, FRG

 0010–4485/83/020059–06 $03.00

BASIC CONCEPTS

The difficulties encountered in the attempt to realize automatic 3D interpretation have their main cause in the loss of semantics occurring when a part is represented with a 2D description: no information is retained regarding the mechanism by which 2D structural elements were generated. Thus while the generation of a 2D representation from a given part is a rather straightforward task (projection of the edges and tangential boundaries from a number of viewing directions), the inverse operation of 3D interpretation will necessarily involve some kind of search in a search space which may be large. Three basic concepts characterize the method developed to solve the problem.

The first is to describe the structural information in terms of a semantic network composed of entities, relationships and attributes (commonly termed a relational structure). Interpretation proceeds by establishing the relational structure of the primitives followed by a gradual building-up of higher-level structures until all objects have been found. Such structural pattern recognition techniques have been successfully applied in the interpretation of visual data[11], and they should also prove suitable for the present problem because of the analogy mentioned earlier.

The second concept is the use of models to guide the interpretation process. In this context a model is a description of the syntactic structure to which the 2D representations of all objects of a given class must conform, in other words, a description of how an object appears in the views of an engineering drawing. The motivation for the use of models is that the problem of recognizing patterns of individual objects in a composite pattern cannot be efficiently solved by a pure bottom-up method, in which primitives would be combined in a more or less exhaustive search until the higher-level structures of objects have been found. Even relatively small-size problems of the kind considered are liable to a combinatorial explosion of possibilities. The use of models reduces the search space because the knowledge about possible structures can be fully utilized.

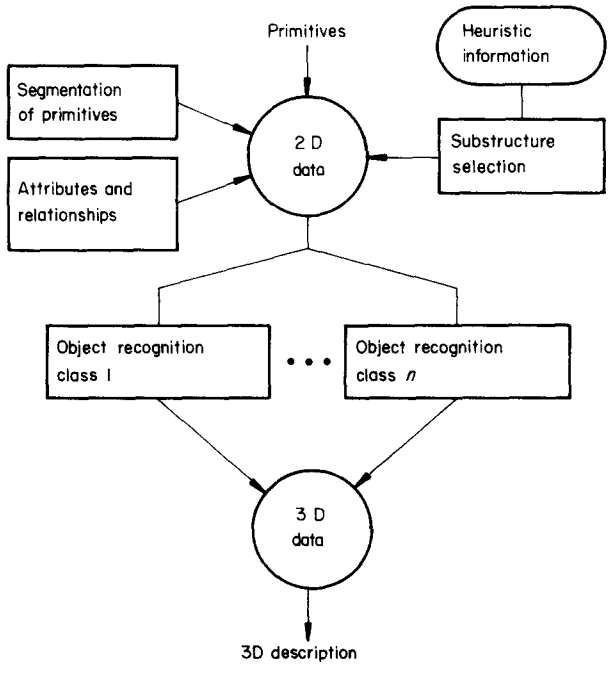

Figure 1. Main components of the system

It is possible to define some classes for the modelling of objects in such a way that a variety of shapes are included within each single class. One such class is, for instance, defined by all objects having a plane base with arbitrary contour and uniform thickness (termed 'uniform-thickness objects' in the following). Another class comprises all objects having rotational symmetry. These two classes are already sufficient for the modelling of many complex structures. Note that they include a large number of the more specific objects frequently used in shape modelling, such as cubes, cylinders, wedges, toroids, rhomboids, spheres and cones.

Finally, the third concept is the use of heuristic information to control the search[12], which, if it were exhaustive, would be much less efficient for the kind of problem encountered here. Heuristic information serves to help select certain substructures of the 2D structure that seem to be promising starting points for a possible interpretation. The patterns of selected substructures are considered hypothetical partial patterns of objects that might be embedded in the structure. The subsequent model-guided search process then leads to either verification or rejection of these hypotheses.

MAIN COMPONENTS

Figure 1 gives a simplified overview of the main components of the reconstruction system, emphasizing the aspect that a number of processes, each having a specific function, operate on the data. Interpretation consists of invoking these processes during repetitive cycles until the 3D structure has been completely established.

Following input, primitives are subjected to a segmentation procedure, which subdivides each primitive at those points where it has an intersection or contact with one or more other primitives. The purpose of this segmentation is to find the basic primitives that serve as building blocks in the formation of higher-level structures. To illustrate the necessity of segmentation, imagine that a circle and a straight line intersecting it have been specified in the input: the circle might be without further semantics, ie might not be part of any object representation, while the two arcs created by the intersection might represent parts of two different objects.

The process termed 'attributes and relationships' scans the data during each cycle, checking if one or more new entities, eg a loop, have been added, and assigns attribute values and inserts relationships as required. Subsequent processes can now rely on the completeness of the relational structure of all entities existing so far.

The task of the 'substructure selection' process is to find substructures whose patterns give evidence that they might represent partial structures of objects. The process makes use of heuristic information to realize a 'best first' strategy, aimed at finding the most promising substructures as fast as possible. An important responsibility of this process is to guarantee that all possible substructures are considered, so that no object is overlooked.

Substructures serve as input for the modules which recognize objects. Each module knows about exactly one object class, and each module accepts a substructure under the hypothesis that it is a partial structure of one or more objects. It then tries to find the complete structures and, if the search is successful, adds new objects to the set of data. All modules for the recognition of objects could in principle operate in parallel.

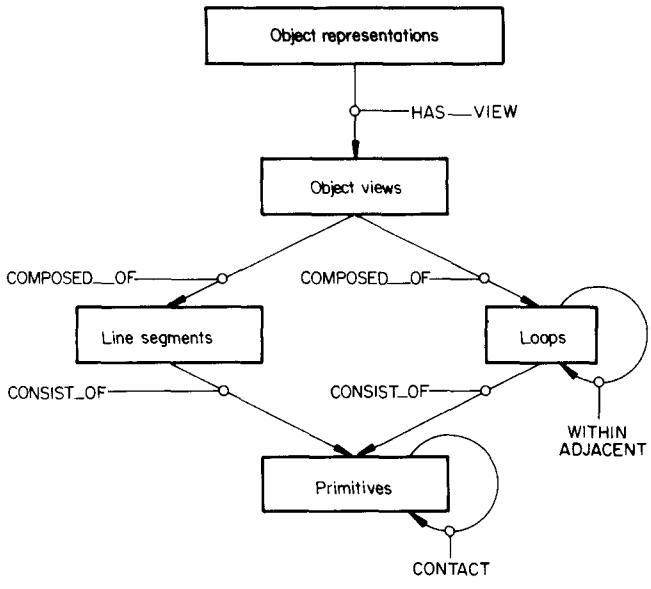

Figure 2. Types of entities and relationships defining the data structure

DETAILS OF THE METHOD

To avoid being confronted with the whole complexity of possible geometries from the beginning, closer elaboration of the method was confined so far to a subset of structures. Accordingly, it will be assumed in the following that the structure to be interpreted is composed of uniform-thickness objects only, with each object oriented such that its base is parallel to one of the coordinate planes. We describe the details of the method for these simpler cases and mention possible extensions later on.

Representation of structure

Figure 2 shows the structural graph defining the types of entities and relationships used to build all structures occurring under the restricted conditions stipulated. The lowest-level elements, of which all higher-level structures are composed, are the primitives, which define straight lines, arcs and circles, depending on their attribute values (see below). There is one possible relationship, CONTACT (p, q), between primitives, meaning that p and q have at least one common point.

A line segment is a grouping of concatenated collinear primitive straight lines. A (single) loop is a closed curve that does not intersect itself. It can be made up of a single primitive circle, or of an arbitrary number of primitive straight lines and arcs. Line segments and loops are related to primitives via the CONSIST-OF relationship. Possible relationships between loops are ADJACENT(a, b), meaning that a and b have a common primitive, and WITHIN(a, b), meaning that the region defined by a is included in the region defined by b. Object views are partial (single-view) 2D representations of objects. They are COMPOSED-OF line segments and loops. Objects are related to object views via the HAS-VIEW relationship.

Some of the entities and relationships have one or more attributes assigned to them. Attributes of primitives are 'subtype' (possible values LINE, ARC, CIRCLE), 'drawing-mode' (possible values SOLID, DASHED), and 'coordinates' (specifying start point, end point, mid point, radius). Attributes of loops are 'shape' (possible values RECTANGULAR, CIRCULAR, SQUARE, IRREGULAR), 'extrema' (specifying minimum and maximum values for each coordinate), and 'visibility' (possible values VISIBLE, INVISIBLE, PARTLY-VISIBLE, according to whether all primitives of the loop are SOLID, all primitives are DASHED, or some are SOLID and some are DASHED, respectively). Primitives, line segments, loops and object views each have an attribute 'viewing-direction' (possible values FRONT, SIDE, TOP), and the CONTACT relationship has an attribute 'angle' (specifying the angle between two primitives at their point of contact).

The actual structure, ie the occurrences of the types of entities and relationships, is build up step by step as the interpretation process proceeds.

Model-guided recognition

The 2D representation of an object takes the form of a triple of specific patterns, representing front, side and top views, respectively. For a uniform-thickness object, restricted in its spatial orientation as stipulated earlier, the patterns can be described as follows (see Figure 3): one of the patterns takes the form of a loop of arbitrary shape. It represents the silhouette of the base (front view in Figure 3). Each of the other two patterns is given by the union of a rectangular loop and a number of line segments subdividing the rectangular region, with the sides of the rectangles and the line segments being parallel to a coordinate axis. These patterns represent lateral silhouettes, edges and projections of surfaces tangential to the line of sight (side and top views in Figure 3). Subparts of the total pattern are interrelated via some matching conditions, which follow from the object's shape and from the rules governing the preparation of engineering drawings. For instance, a match between loops L_1 and L_2 in two different views can be defined by MATCH(L_1, L_2)=TRUE if $\min(L_1)=\min(L_2)$ and $\max(L_1)=\max(L_2)$, where min and max denote the minimum and maximum coordinate values in the coordinate direction that the two views have in common.

Given a partial pattern representing part of one or more objects, the model knowledge can be used to find the complete patterns very efficiently. The following algorithm guarantees that all uniform-thickness objects will be

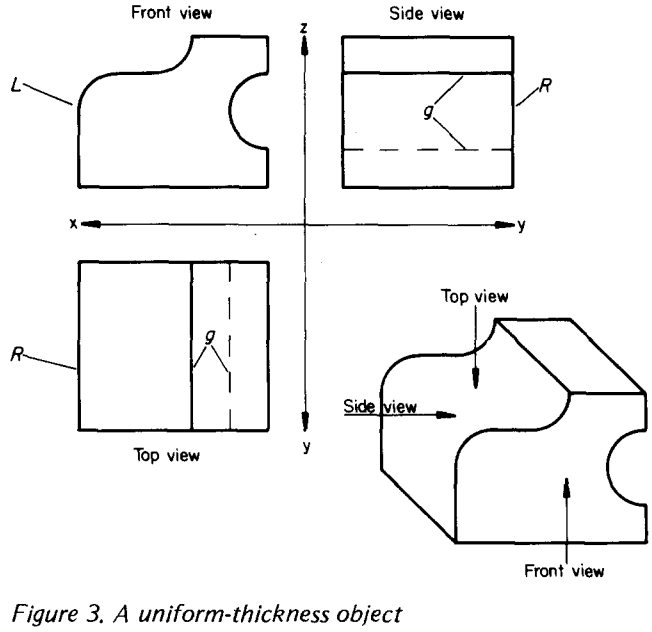

Figure 3. A uniform-thickness object

recognized for which a given loop, L, in a given view, ν_1, represents the base silhouette (see Figure 3):

- Search in view ν_2, where ν_2 can be arbitrarily chosen amongst the views $\nu \neq \nu_1$, and find all rectangles $R_1^{\nu_2}$, $R_2^{\nu_2}$, . . . matching L. Exit if no rectangle is found.
- Search in the remaining view ν_3 and find for each $R_n^{\nu_2} \in \{R_1^{\nu_2}, R_2^{\nu_2}, \ldots\}$ the rectangle $R_n^{\nu_3}$ matching both $R_n^{\nu_2}$ and L. Exit if no such rectangle is found.
- Scan loop L and find all features requiring the presence of line segments in one or both of the views $\nu \neq \nu_1$. (These features are corners formed by primitives of L, relative extreme values of coordinates and lines parallel to a coordinate axis, according to the rules for engineering drawings.)
- For each pair $(R_n^{\nu_2}, R_n^{\nu_3})$, find the complete set of line segments $\{g_{n,1}^{\nu_2}, g_{n,2}^{\nu_2}, \ldots g_{n,1}^{\nu_3}, g_{n,2}^{\nu_3} \ldots\}$ required by the features. If this is successful, a complete object pattern, given by the union of $R_n^{\nu_2}$, $R_n^{\nu_3}$, and $\{g_{n,1}^{\nu_2}, g_{n,2}^{\nu_2}, \ldots g_{n,1}^{\nu_3}, g_{n,2}^{\nu_3}, \ldots\}$, has been found.

A similar algorithm can be designed for the case where a lateral silhouette, R, of an object is given.

Heuristic search

A model-guided recognition algorithm as given above can be successful only if the patterns offered to it are in fact partial patterns of objects. However, it is not possible, in general, to know with certainty whether a given pattern is a partial pattern of an object or whether it is an accidental composition of subpatterns from two or more objects and thus without meaning. Subpatterns can therefore be selected only on the basis of a hypothesis, which is open to both verification and rejection.

Suitable subpatterns for our purpose are the patterns of closed loops, L, which lend themselves to the hypothesis 'L represents the base silhouette of an object'. This is a reasonable hypothesis since we assume that the silhouette of each object is present in each view. However, since pattern components of different objects may be in contact or overlap each other, many spurious loops may be found in the 2D structure. It is not known *a priori* which of the loops really represent silhouettes. Each of the loops is a potential candidate, but only a (possibly very small) subset of them will finally lead to recognition of one or more objects. It is therefore an essential requirement of an efficient solution to select loops in a well-considered order, the aim being to find the overall solution with as few hypotheses as possible.

The 'best first' search strategy devised to satisfy this requirement is based on a selection of loops according to a score assigned to them by an evaluation function that tries to estimate the promise of success for each loop. Two main ideas have been incorporated in the design of this function. The first is to focus the attention of the search process on those pieces of data that have not yet found an interpretation. This is taken into account in the evaluation function by letting the score of each loop depend on the number of its primitives that have not yet been recognized as parts of an object's representation.

The second idea is to use some rough estimate of the probability that a given loop represents the silhouette of at least one object. This is realized by letting the scores depend on the occurrences of relationships and on the values of attributes. For instance, a loop with no ADJACENT relationship established to other loops (an isolated loop) is assigned a higher score than a loop for which one or more such relationships holds, because an isolated loop must necessarily represent a silhouette of at least one object. A loop with a value CIRCULAR of its shape attribute is assigned a higher score than a loop with a value IRREGULAR because an accidental occurrence of a CIRCULAR loop is deemed less probable than an accidental occurrence of an IRREGULAR loop.

Reconstruction algorithm

A complete solution algorithm, devised along the lines discussed above, comprises the following major steps:

- Establish the relational structure of primitives, ie store all occurrences of the CONTACT relationship of primitives.
- Find and store all elementary loops (loops that do not envelop other loops). Assign their attribute values and insert their relationships. Mark all loops as 'open'.
- Assign a score to each open loop according to the evaluation function. Select the loop, L, with the highest score.
- Assume that L represents the base silhouette of one or more objects. Try to verify or reject this hypothesis by looking for all possible interpretations of L, ie call the model-guided recognition algorithm. If complete patterns are found, establish the corresponding 3D structure.
- 'Expand' loop L, ie generate all loops that include both L and one of the loops ADJACENT to L. For each such loop, check whether it already exists. If not, add it to the set of data, calculate its attributes and insert all new relationships. Mark new loops as 'open', and mark L as 'closed'.
- Check, using the rules for engineering drawings, whether the objects found so far explain all input data. If yes, terminate the procedure, else continue with the third step.

Some examples of automatic 3D interpretation, obtained with a programmed version of this algorithm, are shown in Figure 4. The average number of cycles (the last four steps) per example was 24, and the average CPU time was 15 s when run in PASCAL implementation on a VAX 11/780 computer. The heuristic parameters assessing the relative importance of the individual contributions in the evaluation function were chosen tentatively and will not be discussed here because more experience is needed for a well-founded valuation. But to give an impression of their importance, it is mentioned that, in comparison with the case of all parameters set to zero, a reduction in the number of cycles between 2 and 10 was achieved with the tentative parameter adjustment.

CONCLUSIONS AND POSSIBLE EXTENSIONS

The application of the method to some practical cases has shown that automatic interpretation can be achieved at high speed, satisfying requirements of an interactive CAD environment. However, in the light of the problem as it confronts us in practice, the method presents only a first step. We will therefore mention some possible and necessary extensions.

An obvious shortcoming so far is that the full semantics of the 2D representation is not entirely recovered. An essential piece of information not yet provided concerns the question of whether an object is a solid body or whether it is a cavity. Also, the list of recognized objects may, in some cases, include spurious objects, which are due

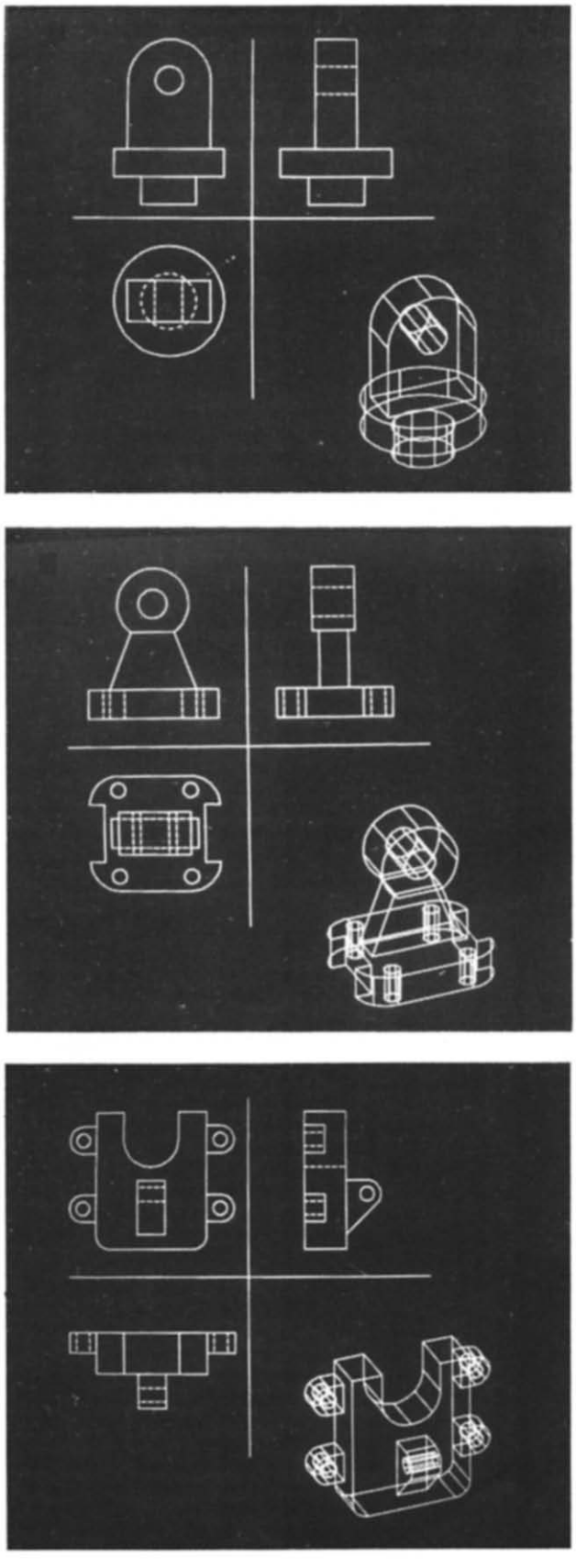

Figure 4. Examples illustrating automatic 3D interpretation

to accidental pattern compositions. The reason for these shortcomings is that the interpretation method so far works on a local basis only, ie it focuses on patterns of single objects but ignores the global context. A remedy would thus be to add a further interpretation step to the alogorithm described above in which global properties would be exploited to find a consistent set of objects with full 3D semantics. (Such an extension is presently being worked on.)

Another possible extension of the method would be to use more classes of objects, so that a broader range of geometries can be covered. First of all, rotationally symmetric objects should be included. Further, the addition of less frequently used objects such as pyramids might be useful. A new object class can be incorporated by adding a corresponding object-recognition module, tailored to the special class. In general, any class whose 2D pattern can be described by some kind of formal rules could be dealt with by the same technique. Since a variety of geometries can be described using only a small number of object classes, the philosophy of having a special module for each class remains reasonable for a wide range of practical applications.

A very important possible extension, in view of practical needs, would be to relax the restrictions imposed so far on the spatial orientation of the objects. It would also be desirable to allow an arbitrary number of views and cuts. The basic concepts presented above should still be applicable to this more general case, but the patterns of objects will be considerably more complex and more difficult to recognize, so that more sophisticated heuristic techniques will have to be developed for a computationally efficient solution.

Finally, a somewhat different philosophy of how reconstruction should be accomplished should also be worth investigating. So far, it was always assumed that the reconstruction process is invoked after the 2D structure has been completely defined. In an interactive design environment, however, where a description of a new part is created by the designer, it seems to be natural to embed the reconstruction process within the data-definition process. The reconstruction could then proceed in the background of an interactive session, advancing incrementally as data become available. A significant advantage of such an approach would be that the process could request additional information or help from the human partner in the man–machine dialogue whenever it encounters situations for which no algorithmic solution has yet been developed. Using the concepts presented in this paper, or some of them, in the design of such an interactive system seems to be a very promising approach.

ACKNOWLEDGEMENTS

The author would like to thank P Blume, who gave the impulse for this study, and W E Fischer for the discussions on the data-structure aspects of the problem.

REFERENCES

1 **Blume, P** 'Computer-aided design' *Philips Tech. Rev.* Vol 36 No 6 (1976) pp 162–175

2 **Idesawa, M** 'A system to generate a solid figure from three view' *Bull. JSME* Vol 16 No 92 (February 1973) pp 216–225

3 **Idesawa, M and Shibata, S** 'Automatic input of line drawing and generation of solid figure from three-view data' *Proc. Int. Comput. Symposium 1975* Vol 2 pp 304–311

4 **Shapira, R** 'A technique for the reconstruction of a straight-edge, wire-frame object from two or more central projections' *Comput. Graphics Image Proc.* Vol 3 (1974) pp 318–326

5 **Lafue, G** 'A theorem prover for recognizing 2D representations of 3D objects' *Proc. IFIP TC-5 Working Conf. Artif. Intell. Comput. Aided Design* Grenoble, France (March 1978) pp 391–401

6 **Liardet, M, Holmes, C and Rosenthal, D** 'Input to CAD systems: Two practical examples' *Proc. IFIP TC-5 Working Conf. Artif. Intell. Comput. Aided Des.* Grenoble, France (March 1978) pp 403–414

7 **Preiss, K** 'Algorithms for automatic conversion of a 3-view drawing of a plane-faced part to the 3D representation' *Comput. Industry* Vol 2 (1981) pp 133–139

8 **Sugihara, K** 'Mathematical structures of line drawings of polyhedrons — toward man–machine communication by means of line drawings' *IEEE Trans. Pattern Analysis Machine Intell.* Vol PAMI-4 (September 1982) pp 458–469

9 **Farny, B** 'Geometrisches Modellieren technischer Objekte durch Rekonstruktion der dreidimensionalen Werkstueckgeometrie aus zweidimensionalen Zeichnungsrissen' *Preprints Geom. Modelling Conf.* Technical University of Berlin, FRG (November 1982) pp 23–31

10 **Jansen, H and Meyer, B** 'Rekonstruktion von volumenorientierten 3D-Modellen aus handskizzierten 2D-Ansichten' *Preprints Geom. Modelling Conf.* Technical University of Berlin, FRG (November 1982) pp 33–44

11 **Barrow, H G and Tenenbaum, J M** 'Computational vision' *Proc. IEEE* Vol 69 No 5 (May 1981) pp 572–595

12 **Nilsson, N J** *Problem-solving methods in artificial intelligence* McGraw-Hill (1971)