Constraint-based solid modeling with geometric features

Zhang, J., Middleditch, A., Latham, R.

J. J. Zhang, A. E. Middleditch, R. S. Latham, "Constraint-based solid modeling with geometric features," Proc. SPIE 2644, Fourth International Conference on Computer-Aided Design and Computer Graphics, (22 March 1996); doi: 10.1117/12.235510

Event: Fourth International Conference on Computer-Aided Design and Computer Graphics, 1995, Wuhan, China

Constraint based solid modelling with geometric features

J. J. Zhang A. E. Middleditch R. S. Latham

Department of Computer Science, Brunel University

Uxbridge, Middlesex UB8 3PH, UK

ABSTRACT

Engineering design practice usually follows a design hierarchy. Constraints more or less reflect engineering requirements. But constraints are practically useful only when they are correctly embedded into design hierarchies. The concept of *geometric features* is introduced to serve the following purposes: to allow a designer to work with the conventional *design hierarchy*; to control the effect of constraints; and to decompose a complicated constraint based geometric modelling problem into simpler ones. Geometric features are constructed hierarchically. Although constraints are applied to low level geometric entities (points, lines, etc.), the impact on a design model is usually imposed on higher level features. In terms of constraints processing, each feature is viewed as an independent problem and its constraints are solved independently. All constraints in a design model are therefore solved hierarchically bottom up.

Key words: Solid modelling, geometric feature, constraint based modelling

1. INTRODUCTION

The birth of solid modelling advances CAD from 2D modelling to 3D. However, the set operations (union, intersection and difference) and primitives solids (block, cylinder, sphere, etc.) adopted in traditional solid modelling systems do not provide designers with approaches which are natural in a sense of engineering practice. Designers have difficulty in positioning and dimensioning a large number of primitives. Moreover, the designer has to remember all of the interrelationships between various parts of the design when making design modifications.

Constraint based modelling techniques have proved to be the first viable solution to the above mentioned problems. They employ sets of rules, or constraints, that help define how changes to a group of geometric elements should be handled so as to best protect the integrity of the design. For example, if the diameter of a hole changed, it would not be the designer's responsibility to remember to change the diameter of the bolt that passed through the hole, rather it is the system that should catch up with the change.

The impact of constraint based solid modelling has been so profound that great attention has been drawn to this area. However, due to the complexity of constraint processing and the design of user interfaces to geometric modellers, most modelling systems reported so far can only confidently deal with 2D constraints. In these 2D cases, the geometry is manipulated by controlling individual entities, e.g. points and curves, directly[1,2]. Only a little work has been reported in 3D constraint based solid modelling, although progress is being made in parametric modelling, where constraints are captured and solved in a strictly predefined order.

It has been realised that constraint based techniques offer substantial potential in assisting the design process. But they would be of little interest to engineers, if only individual geometric entities are manipulated. In fact, a designer normally devises his/her design in terms of a design hierarchy. Two steps are usually conducted in engineering design practice. The whole design or the general assembly is first conceptually laid out. It is then subdivided into sub-assemblies fulfilling significant requirements at this level. Sub-assemblies are further detailed into parts. A part is also made up of several levels of features, such as bosses, slots, a group of holes and possibly an individual hole. This is the first step, which is performed from top down. The second step is carried out in the reverse order. A part is designed in terms of features. Parts are then assembled, so are the subassemblies.

Constraints more or less reflect the engineering requirements. But the constraints are practically useful only when they are correctly embedded into design hierarchies. It would otherwise be difficult to answer some engineering design questions. For example, if there is a distance constraint between an end face of a shaft and a fixed face, how would the shaft react to the distance change; would it stretch or move? If there is a key way on the shaft; how would the key way be changed? In fact, although constraints connect only low level entities (points, lines and faces, etc.), the real bearing on a design is imposed through design hierarchies.

To advance 3D constraint based modelling, in addition to appropriate constraint processing algorithms[3,4], techniques which enable engineering rules to be easily mapped into constraints that assist design are needed. This paper introduces a design paradigm which combines *geometric features* (explained later) with *constraints*. Rather than controlling individual geometric entities, such as points and lines, it aims to provide designers with tools to work at a high semantic level appropriate to their needs. A design can thus be implemented in conformity with engineering design practice.

2. A BRIEF REVIEW OF CONSTRAINT BASED MODELLING TECHNIQUES

In Sohrt and Bruderlin's system[5], implicit constraints are generated when modelling operations are carried out. These implicit constraints eliminate some degrees of freedom. All the constraints are based on the so called constraint coordinate system. Initially, this coordinate system is aligned with the local coordinate system. During the course of modelling, if a point is fixed, it becomes the new origin of the system. So the system is changing according to the operation. Since the implicit constraints are usually simple, they are solved immediately after being defined and drag-handles display the state of current degrees of freedom.

Using 3D input devices, Fa et al developed a solid modelling system[6] which integrates implicit constraints specification and direct manipulation. With 3D direct manipulation, the system figures out the implicit constraints. The allowable motion is then determined and used to automatically constrain subsequent geometric manipulations without invalidating the associated constraints. Automatic model assembly is based on the extracted aligning information.

De Kraker et al[7] and Emmerik[8] have developed GeoNode, a constraint-based CSG modelling system, which is also based on direct manipulation. In this system, a geometric model is constructed by building a hierarchy of coordinate systems, called geometric tree. This tree is used to specify the position, orientation and dimensions of all primitives in the model. For a primitive, the position and orientation are defined by the base coordinate system of the primitive. The dimensions are determined by their dimensioning coordinate systems, the distances between the base and dimensioning coordinate systems determine the dimensions of the primitive. They categorise constraints into high-level and low-level ones. The former are those defined between primitives and the later are defined on characteristic points of primitives. Constraints solving has been implemented with a pre-processing step which translates high-level constraints to low-level ones to facilitate the resolution process. The proposed scheme is not suited to more complicated design models, because the high level constraints are manipulating primitive solids individually.

In general, most constraint based modelling systems work at a low semantic level in terms of the design hierarchy. Research work in constraint based modelling is mostly devoted to algorithms for constraints processing; there is little work on the interpretation of constraints in engineering design practice. It is important, especially for 3D modelling, to advance constraint based modelling to a high semantic level.

3. GEOMETRIC FEATURES AND DESIGN HIERARCHY

As stated previously, engineers normally follow a design hierarchy. In order to be compatible with engineering practice for the constraint based solid modelling techniques, geometric features are proposed.

3.1 Geometric features

A *geometric feature* is a hierarchical representation of point sets. Geometric entities, such as points, curves, surfaces and solids, are instances of point sets.

3.1.1 Definition of geometric features

By using the Backus-Naur form (BNF) language[9], *geometric features* are recursively defined below, where feature denotes a geometric feature:

feature = primitive_feature / set(feature)

primitive_feature = set(point_set)

point_set = set(point) = set(R^3)

Here *a*/*b* denotes either *a* or *b*. Therefore, a geometric feature is either a single primitive feature (i.e. a set containing a point set) or a combination of features which may or may not be primitive features. For example, if $P_1, P_2, \ldots, P_6$ are point sets and $f_1, f_2, \ldots, f_6$ are the corresponding primitive features, F_1, F_2, F_3 are non-primitive features, they can have following relationships:

$$f_1 = \{P_1\}, f_2 = \{P_2\}, \ldots, f_6 = \{P_6\}$$

$$F_1 = \{f_1, f_2, f_3\}$$

$$F_2 = \{f_4, f_5\}$$

$$F_3 = \{F_1, F_2, f_6\}$$

Here { } denotes an instance of a set.

A geometric feature is mapped to a point set by the mapping function:

$$\mathrm{PointSet}(\,) \equiv \mathrm{feature} \rightarrow \{R^3\}$$

$$\mathrm{PointSet}(F) = \begin{cases} F, & F \in \{R^3\} \\ \bigcup_{i=1}^{n} \mathrm{PointSet}(F_i), & \text{otherwise} \end{cases}$$

where F_i is a feature and $F = \{F_1, F_2, \ldots, F_n\}$

This mapping function from feature to point sets defines the geometry of combinations of geometric features.

Thus a *geometric feature* is not exactly the same as the *form feature* used in the references[10,11], despite some similarities. Although, nothing should prevent a geometric feature performing the duties of form features when necessary, the use of feature for CAD and CAM integration is not of main interest of this paper. In the context of the paper, features are used as shorthand for geometric features wherever no confusion could occur.

3.1.2 Properties of geometric features

- Features are constructed hierarchically. Higher level features are created from lower level ones. Features are thus represented as a (not necessarily binary) tree structure, called a *feature tree* (figure 1a). Each node of the tree represents a feature. Usually, a feature has an explicit engineering meaning.
- Geometric features at the leaves of the tree are primitive features. These features represent the basic geometry of engineering features such as slots, flanks, bosses or primitive solids such as blocks, cones and spheres.
- A geometric feature in a design can be manipulated independently.
- A feature hierarchy will be easily extended to the level of assembly. Branches of a feature tree can represent distinct parts. Therefore a part can be regarded as a feature and so can a group of parts.
- To be compatible with engineering convention, features in different parts do not overlap.
- All the geometry is represented in a unified structure from primitive features (bottom level) through middle level features, parts, assemblies to the whole design (top level). Clear design concept and simple manipulation of a design process are easy to achieve.
- Recall the design hierarchy adopted in engineering, each feature represents a link in the hierarchy. Thus the design practice is readily mapped to the computer aided design process with or without constraints.

3.1.3 Representation of geometric features

As mentioned above, features are represented as a tree structure with arbitrary number of branches. Each feature is associated with a piece of geometry and a local co-ordinate system, which is called a *frame*. All the geometric properties (position, orientation, etc.) are defined with respect to the frame. A transformation associated with each feature transforms the geometry of the child feature to the frame of the parent feature.

3.2 Manipulation of geometric features

Geometric manipulation is achieved through geometric features. Here follows a list of feature operations, some of which manage features and others manipulate geometric entities.

- **Primitive feature creation**. When primitive geometry is created, e.g. a cuboid, it is automatically associated with a feature. Such a feature is the *primitive feature*.
- **Feature creation**. *Non-primitive features* are created by grouping lower level features (refer to the definition in section 3.1.1). Since these features may already have a common parent feature, it is the system's responsibility to include the newly created feature within that parent. For example, new feature $\{F_1, F_2\}$ can be created as a child of feature $\{F_1, F_2, f_3\}$ which then becomes $\{\{F_1, F_2\}, f_6\}$. A new node is then inserted into the feature tree. It is not possible to combine features with different parents. The designer can make as few or as many levels of features as required.
- **Feature removal**. This is the inverse of *feature creation*. It does not delete the associated geometry, it merely deletes the node in the feature tree. The system detaches the feature from its parent, recognises the hierarchy for the child and parent features and corrects the child frame transformations. *Feature rearrangement* is achieved by deleting and creating features.
- **Feature transformation**. Features can be transformed, such as translated, rotated and differentially scaled. Such transformations operate on the associated geometry.
- **Feature deletion**. This deletes the feature and all its child features including all the associated geometry.

With the functions listed above, a design can be manipulated by controlling the geometric features directly. This scheme provides a convenient way for the user to make any modification even without using constraints.

4. CONSTRAINTS WITH GEOMETRIC FEATURES

Geometric constraints are usually imposed on geometric entities such as points and lines. However, a constraint in a solid modelling system may affect either a primitive feature or a higher level feature. In terms of constraints management, geometric features serve two purposes: they control the effect of constraints and decompose a complicated geometric constraint problem into simpler ones.

4.1. Constraint based modelling with primitives

In solid modelling systems, primitive solids (block, cylinder, extruded and revolved volumes, etc.) are basic geometric elements, based on which more complicated designs are constructed[12]. To help explain how constraints work with geometric features, we first consider the case where only the primitives in a set theoretical expression are controlled and modified by constraints. These primitive solids are equivalent to the primitive features previously defined, but they are combined by set operators. Thus, a geometric model is not constructed by features as defined above.

4.1.1 Constrainable parameters

The following parameters of primitives are manipulated by constraints:

- Rigid body motion parameters. They are three position and three orientation parameters.
- Differential scaling parameters in three orthogonal directions x, y and z.
- Other shape control parameters, such as the apex angle of a cone and the radius of a cylinder.

Changes to these parameters cause the geometry of the model to be updated.

4.1.2 Representation

Suppose primitives are the only elements which can be controlled and modified by constraints. Their coordinate systems and relations between them are as follows:

- One local coordinate system (LCS) is associated with each primitive. The geometry of the primitive is relative to the LCS
- The global space (GS) where the geometric model stays is regarded as the top level of the design hierarchy. GS is associated with the global coordinate system. Therefore it is a two level system.
- The mapping between the global co-ordinate system and each LCS is achieved by a transformation, which includes rigid body motion and differential scaling. The change of the transformation leads to rigid body movement as well as the modification of the geometric shapes.

Constraints are of three types: dimensions, geometric relations and simple arithmetic relations. All constraints are applied to geometric entities, which are points, unbounded lines and planes, and linear and angular dimensions. Geometric entities are associated with their own local co-ordinate systems. Constraints belong to the global space. Constraints therefore cause the transformations of primitives to change, rather than entities themselves.

This scheme is better than the traditional solid modelling approach. However, since every primitive has to be fully and correctly constrained in the global context, it is tedious especially when there are a large number of primitives and constraints. In addition, basic primitives do not normally have engineering meaning thus are inconvenient for designers.

4.2. Constraint based modelling with geometric features

It is necessary to reduce the tedious design operations and make the design process more meaningful while keeping the merits of constrained set theoretical primitives. If each child feature plays the role of a primitive and each parent feature plays the role of the global space described in the last section, the techniques developed for primitives can be readily adopted.

4.2.1 Representation

- **Features**. Except for the transformation which maps to the coordinate system of the parent feature, a feature is independent of higher level features. A feature consists of a frame (LCS), within which geometric entities, child features and constraints are defined.
- **Frames**. Analogous to the relationship between features, the frame of a child feature is also the child frame of the frame in the parent feature. Relationships between frames are represented as a tree structure and the top of the tree is the global co-ordinate system. Transformations are links between child and parent frames.
- **Constrainable parameters**. The parameters controlled by constraints are the same as those of primitives in the last section. Those parameters are used to modify transformations and shape control dimensions.
- **The domain of effect of a constraint**. A constraint in a feature is used only to change the transformations of the immediate child features, although it could constrain geometric entities of possibly lower level features. It certainly has no direct impact on the grandchild features transformations.
- **Constraint placement**. A constraint should be placed in the lowest level *common parent feature*, which contains all the constrained entities. Figure 1 shows the hierarchy of features and placement of constraints. Figure 1a illustrates the feature tree. Fea1 is the top feature or the whole design. Fea2 and Fea3 are child features of Fea1, Fea4 and Fea5 are the child features of Fea3. Constraints Con1 and Con2 should belong to the feature Fea3, and Con3 to the lowest level common parent feature Fea1.

4.2.2 Properties of the proposed scheme

- It allows constraints and geometry in one feature to be treated as if they formed an independent design and thus solved independently.

- A feature is regarded as being encapsulated in its parent feature. The constraints in a feature have no impact on the parent and brother features. Hence a complicated problem of constraints can be decomposed using features and processed more efficiently.

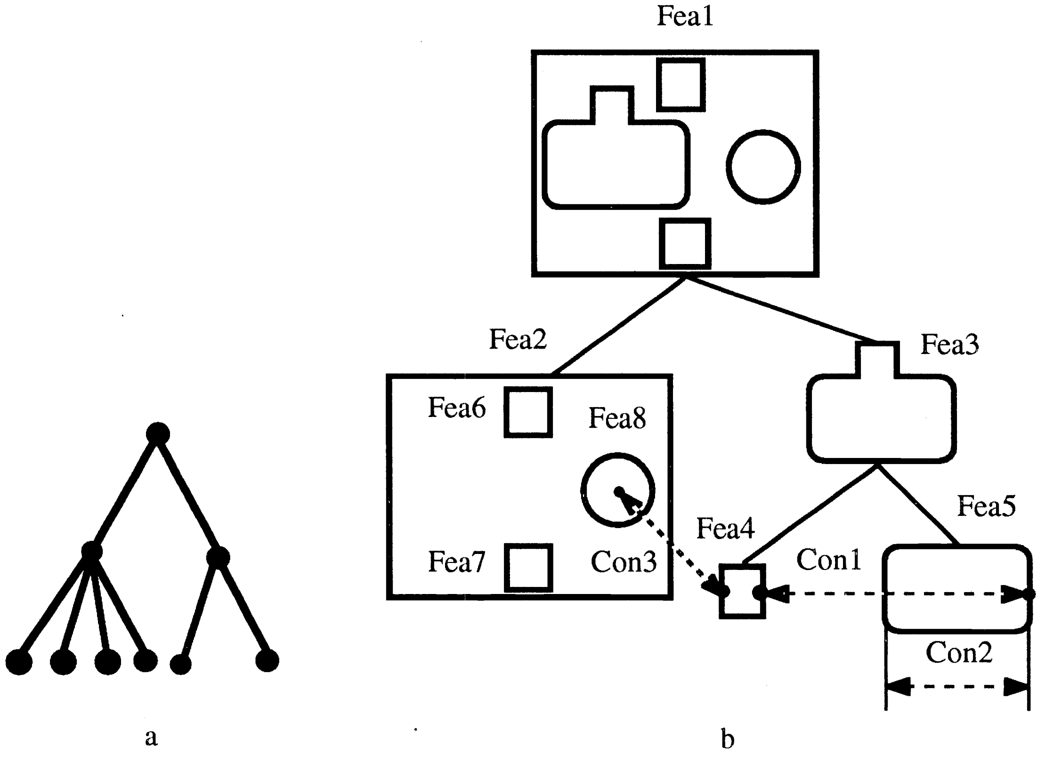

Fig. 1 Feature hierarchy and constraints placement

- Under-constrained child features are allowed. In fact, since a child feature is treated as a rigid body plus necessary dimensions, it need not have any constraints. As illustrated in figure 2, there are no constraints in the feature Mid-Fea. The constraint Con would actually move the slot as well as the hole relative to the base feature. This makes the designer's work easier.

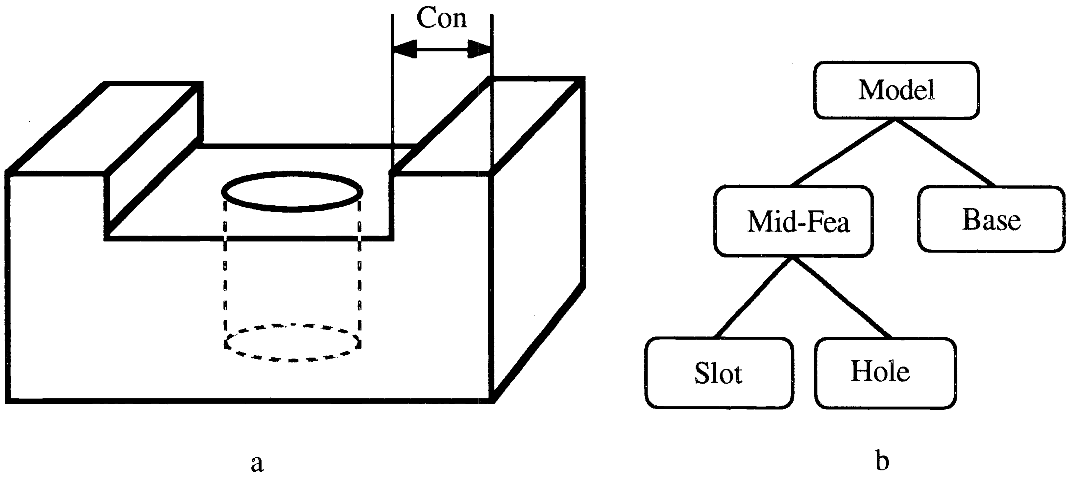

Fig. 2 Under constrained case is allowed in child features

- Constraints resolution. All the constraints in a design model are solved by solving the constraints from one feature to another, following the hierarchically from the bottom up.

4.2.3 Constraints satisfaction

Constraint processing is conducted in three steps: pre-processing, constraints resolution and post-processing. Constraint resolution is undertaken by an independent constraint solver, whose algorithm is beyond the scope of the paper.

- Pre-processing: The information of the current geometric parameters, feature hierarchy and constraint configuration is conveyed to the constraint solver. This information includes: position, orientation and scaling of the LCS associated with each feature; the constraints, the entities being constrained and the geometric configuration associated with the LCS.
- Post-processing: The results from the constraint solver, i.e. the transformations between each LCS and its parent LCS and necessary dimension values are retrieved and the geometry is updated.

5. DESIGN3D: AN EXPERIMENTAL CONSTRAINT BASED SOLID MODELLING SYSTEM

The proposed scheme of geometric feature and constraint based solid modelling has been implemented in our experimental interactive system DESIGN3D. Due to the length limit on the paper, only will the brief description be given.

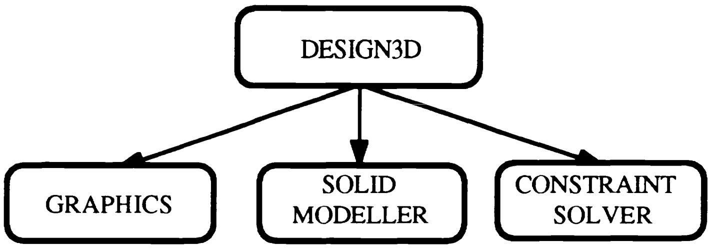

Fig. 3 The structure of DESIGN3D

DESIGN3D interfaces to three functional modules: a solid modeller, a graphics display system and a constraint solver (figure 3). Inside DESIGN3D, there are also three functional modules: the feature module which deals with the feature hierarchy; the constraints module which manages the constraints and their connections to features; and the user interaction module.

In general, the user has two options when using DESIGN3D to model a design: directly manipulating geometric features and using constraints with features. Figure 4 shows a simple example modelled and assembled by DESIGN3D with constraints and features. Each feature corresponds to a separate part. Lines between features denote constraints.

6. CONCLUSIONS AND FUTURE WORK

Flexibly defined *geometric features* are aids to designers. They can be transformed either by direct manipulation or by constraints. The feature hierarchy is an emulation of an *engineering design hierarchy*.

Constraints are applied to individual entities, such as points and lines. However, they do not normally effect the entities themselves directly. In the system proposed by this paper, the domain of effect of a constraint is clearly defined according to the feature hierarchy. Thus a constraint is in the sense mapped to engineering rules.

In the proposed approach, constraints are solved independently for each feature. A set of complicated constraints can hence be decomposed and simplified by the hierarchy of geometric features.

Although the feature hierarchy appears to be consistent with the design hierarchy, due to the complexity of engineering design, constraints do not always follow the strict hierarchy described in this paper. Apart from improving the user interface functionality in DESIGN3D, future work will concentrate mainly on tackling this problem.

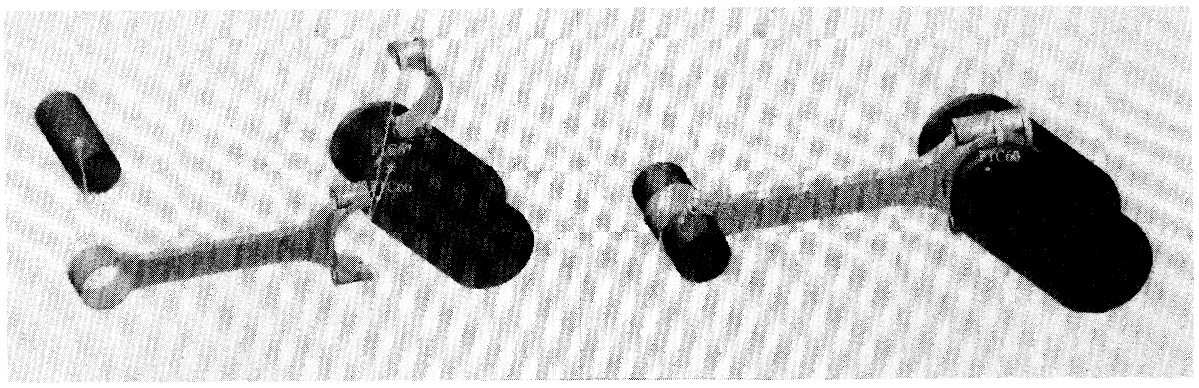

Fig. 4 An example modelled with DESIGN3D

7. ACKNOWLEDGEMENT

The authors wish to acknowledge the Engineering and Physical Sciences Research Council of Britain for their financial support through Grant GR/J/18712.

8. REFERENCES

1. G. A. Kramer, "A geometric constraint engine", Artificial Intelligence, Vol.58, pp 327-360, 1992.

2. J. C. Owen, "Algebraic Solutions for Geometry from Dimensional Constraints", Proc. of Symposium on Solid Modelling Foundations and CAD/CAM Applications, pp397-407, 1991.

3. G. A. Kramer, Solving Geometric Constraint Systems MIT Press, 1992

4. R. S. Latham and A. E. Middleditch, "Connectivity analysis: a tool for processing geometric constraints", submitted to Computer-aided Design.

5. W. Sohrt and B.D.Bruderlin, "Interaction with constraints in 3D modelling", Proceedings of the Symposium on Solid Modelling Foundations and CAD/CAM Applications, ACM Press, pp387-396, 1991

6. M. Fa, T. Fernando and P. M. Dew, "Interactive Constraint-based Solid Modelling Using Allowable Motion", Proceedings of the Second Symposium on Solid Modelling and Applications, ACM Press, pp243-252, 1993

7. K. J. de Kraker, M. Dohmen and W. F. Bronsvoort, "High-level Constraints in Interactive CSG Modelling", Proceedings of the CSG'94 Conference, Winchester, UK, 1994.

8. M.J.G.M. van Emmerik, "Interactive Design of 3D Models with Geometric Constraints", Visual Computer, Vol. 7, No. 5/6, pp309-325, 1991

9. P. J. Brown, Writing interactive compilers and interpreters, Chapter 4.3, John Wiley and Sons, 1979

10. L. C. Sheu, J. T. Lin, Representation scheme for defining and operating form features, Computer-aided Design, Vol.25, No. 6, pp333-347, 1991

11. K. Case, J. X. Gao, N. N. Z. Gindy, "The implementation of a feature based component representation for CAD/CAM integration", Proc. Instn. Mech. Engrs, Part B: Journal of Engineering Manufacture, Vol. 208. pp71-80, 1994

12. A. A. G. Requicha and H. B. Voelcker, "Solid Modelling: A Historical Summary and Contemporary Assessment", IEEE Computer Graphics and Applications, Vol.2, No. 2, pp 9-24, 1982