File No. UIUCDCS-F-86-957

Toward a Practical Planning System for Assembly Tasks

Heedong Ko
Artificial Intelligence Laboratory
Department of Computer Science

Kunwoo Lee
Department of Mechanical and Industrial Engineering
University of Illinois at Urbana–Champaign

ISG 86–7

September 1986

Toward a Practical Planning System for Assembly Tasks

by

Heedong Ko and Kunwoo Lee

ABSTRACT

A method is developed to automatically generate an assembling procedure for an assembly. In this work, an assembly is described by its components and their relationship in an assembly, especially the mating conditions. From the mating conditions of the components, an assembling procedure is generated in two steps. First, a hierarchical structure of the assembly is generated. Second, an assembly procedure is generated from the hierarchical structure with the help of interference checking.

To be published in *Computer Aided Design,* February 1987, under the title of "Automated Assembling Procedure Generation from Mating Conditions."

1. INTRODUCTION

A designer creates a physical object meeting some functional requirements, using computer aided design tools, and the manufacturing engineer has to create a procedure to manufacture the object, possibly a robot program. Such decentralization of expertise created a lack of communication and misunderstanding between them. A designer worries the manufacturability of the object he is designing while the manufacturing engineer worries the intended functional requirements of the object so that many iterative cross checks between them may be required before the object is manufactured. Computer integrated manufacturing (CIM) is to combine the design and manufacturing activity into one integrated working environment.

We view the key to CIM is the intelligent link from a designed object to its manufacturing process. Traditionally, this link is known as planning in AI. That is, given a description of the goal, the designed object, a planning system generates a sequence of actions to achieve the goal state from the working environment.

Previously, planning was mainly concerned with creating plans in simple assembly task domain, the block's world [Fikes 1971, Sussman 1975]. Their systems laid the background for most contemporary planners. One of the main limitations of classic planners is that their source of power came mostly from knowledge about actions. Little attention was paid to integrating planners with commonsense reasoning modules. Future planners should derive its power much more from knowledge about objects and processes of the world. Some contemporary planners are already being developed with more sophisticated temporal notions [Allen 1983, McDermott 1982] so that planning is possible under the influences of temporal processes. Here, we explored a way to integrate knowledge about objects with a planner for generating assembling procedure.

Representation of objects and their relationships in classic planners was too abstract and idealized. The description of an assembly suggested by Lieberman and Wesley [Lieberman 1977], and Wesley, et al. [Wesley 1980] shows more detailed symbolic relationship between components such as "part–of", "attachment", "constraint", and "assembly". However, it does not provide enough geometric information required for assembling procedure generation. Here, Lee's virtual link [Lee, Grossard 1985] was used to describe the assembly. A virtual link is an entity between any pair of mating components in an assembly and carries all the mating conditions between the two components. Thus a designer has to provide only the mating conditions between mating components to describe an assembly.

The reported work is implemented in a program called, APG (Assembling Procedure Generator). After all the mating conditions of an assembly are given, APG produces a hierarchical

This material is partially based upon work supported by National Science Foundation under grant DCR 84–06801, Office of Naval Research under grant N00014–82–K–0186, Defense Advanced Research Project Agency under grant N00014–K–85–0878, and the National Science Foundation under Grant DMC–8505941.

structure of the assembly. Then an ordering constraints between components of an assembly are generated with the aid of interference checking.

2. REPRESENTATION OF AN ASSEMBLY

An assembly can be represented by specifying its components and their relationships in the assembly. In this work, the components themselves are specified by their boundary representations [Mortenson 1985] and their relationships in an assembly are specified by the mating conditions between all the mating components. Currently, four types of mating conditions are considered. Other types of mating conditions may have to be added in the future by studying many other examples. The four types are against, fits, tight-fits, and contact conditions. These mating conditions are explained with our example, bell assembly, as shown in Fig. 1.

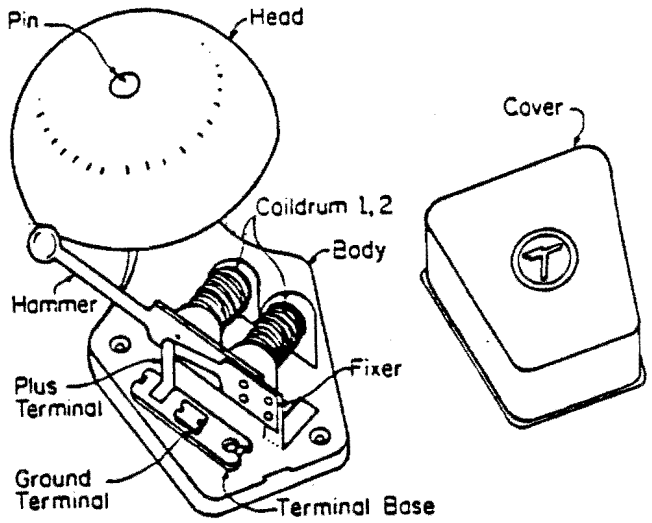

Fig. 1 Bell Assembly

Against condition holds between planar faces of a pair of components. In Fig. 2a, the face 5 of body (f_5,Body) is against face 2 of head (f_2,Head). This mating feature merely assures face 5 of body and face 2 of head be in contact at all time. That means head could move around as long as face 2 is in contact with face 5 of the body. Some of the possible movements would be that the head could rotate around the body or that the head could slide against the body.

Fits condition holds between a solid cylinder and a hole. In Fig. 2a, the centerline of a pin (C_1,Pin) is aligned with the centerlines of the head and body (C_1,Head and C_1,Body). Fits condition allows rotational freedom of movement between the involved components and also the translational movement along the centerline.

There are some components with fits condition that do not have rotational degree-of-freedom between each other. For example, the rod and the coildrum are related by fits condition

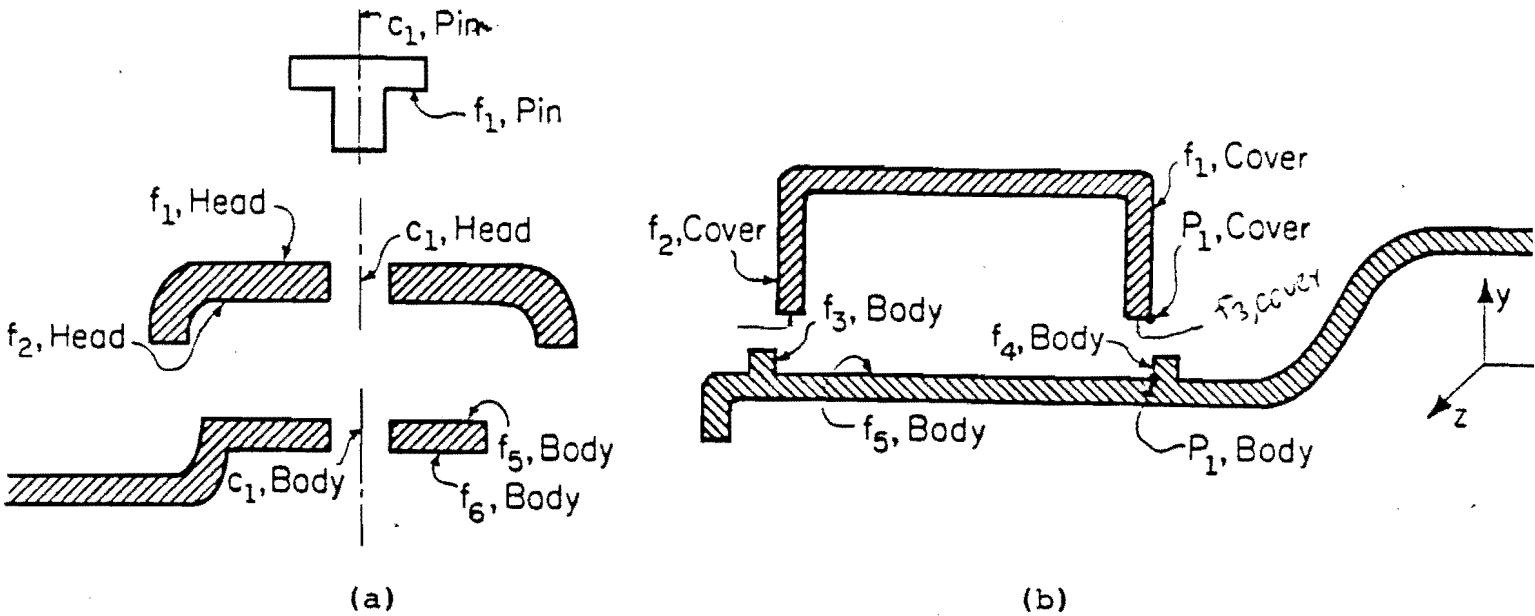

Fig. 2 Head and Cover Assemblies

but the force to rotate the rod was too great to be called a rotational degree-of-freedom. Such mating condition is defined as a tight-fits condition.

Body and cover are related by three against conditions: one between face 2 of the cover and face 3 of the body, the other between face 1 of the cover and face 4 of the body and between face 3 of the cover and face 5 of the body as shown in Fig. 2b. Although the geometry of the 3 against conditions allowed one degree of freedom along z-direction, the cover was fixed to the body. This was true between body and fixer, between body and terminal base, between terminal base, ground terminal and between terminal base and plus terminal and between hammer and fixer. We introduced an additional mating condition (a 'contact' condition) to prevent any movement. A contact condition holds between two points of the faces of the involved components. In Fig. 2b, point 1 of body is related to point 1 of cover by a contact condition.

All the mating conditions for bell assembly are represented in a mating graph and is shown in Fig. 3.

3. HIERARCHICAL STRUCTURING OF COMPONENTS

An assembly is naturally thought of as a hierarchical structure. That is, an assembly is composed of subassemblies, each subassembly is composed of groupings, and each grouping is composed of several components. Any two components are in different subassemblies if the components have relative motion with respect to each other. Any two components in a subassembly are in different groupings if they do not mate directly except the base component.

It is economical to conceptualize an assembly as a hierarchical structure because locality

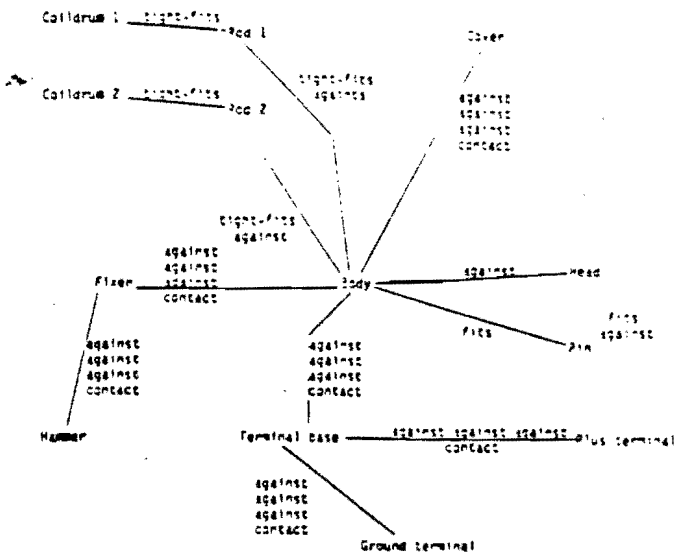

Fig. 3 Mating Graph of Components

between components for various analysis of the assembly is made explicit in the structure. For example, it is highly unlikely that the kinematic analysis of the wing of an airplane should be concerned with that of a passenger seat. Locality is especially important when the number of components of the assembly is large such as an airplane. We will make use of locality assumption in generating component orderings in the following section.

A procedure based on the mating graph in Fig. 3 works as follows. Identify a base component. A base component has a special status. During assembling the components in the grouping, the base component is to stay fixed. The criterion for choosing a base component can be updated. For bell assembly, the component which is connected to the greatest number of components by virtual links (body for the bell assembly) is chosen as a base component.

Gather all the components directly connected to the base component by virtual links of non-subassembly type. A virtual link is subassembly type if the components involved have relative movements to each other and non-subassembly type otherwise. The reason we include those components connected via non-subassembly type is that the shape of the grouping need to be computed for finding a collision free path while the robot carries the grouping object. It is easier to compute the shape if the components of the grouping do not have any movements.

In our bell example, cover, rod1, rod2, fixer and terminal base are gathered with body as a base component. Note that head and pin are not included even though they are connected to body via virtual links because the associated virtual links are of subassembly type because the head and body have a relative movement as discussed in the previous section. The gathering procedure is recursively called using each component gathered as a new base component. When this

process terminates with the body as a base component, all the components related to body via non-subassembly type virtual links directly or indirectly are formed.

All the components not yet visited are because they belong to different subassembly. Here, head and pin are not visited. Again identify the base component among the components left to be visited. Both head and pin have only one virtual link, ignoring virtual links that involve components already visited. Thus, either head or pin is a next base component.

Using head as a base component, there are no components related by non-subassembly type virtual link that had not been visited. Pin is connected to head via a virtual link but head can rotate around the pin because the fits condition allows the rotational motion and the against conditions do not prevent the motion. So head is the only component for this subassembly. Same is true for pin. In general, these subassemblies could form a grouping hierarchy of their own as for body. Fig. 4 is the final outcome of the procedure.

A general purpose kinematic analysis for identifying virtual links of subassembly type is required. Some sufficient mating conditions for subassembly virtual links are identified but they are by no means complete:

- only one fits condition,
- only one against condition,
- one fits and one against, if the centerline of fits is perpendicular to the planar face of against,
- arbitrary number of against conditions if all the normals of the planes of the against conditions are coplanar.

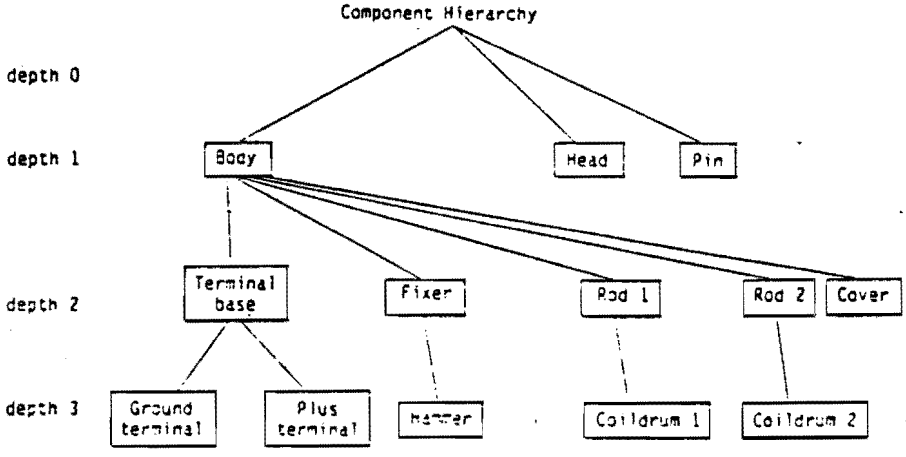

Fig. 4 Component Hierarchy of Bell Assembly

4. ASSEMBLING PROCEDURE GENERATION

The base component is assumed to be assembled after its descendants as mentioned in the previous section. In Fig. 4, terminal base should be assembled to body only after ground terminal and plus terminal are assembled to terminal base. This can be represented as follows:

Ground terminal < Terminal base

Plus terminal < Terminal base

Terminal base < Body

So, the orderings between siblings and their parent comes directly from the component hierarchy. Here, comp1 < comp2 means comp1 should be assembled before comp2. Note "<" relation is transitive: that is, comp1 < comp2 and comp2 < comp3 implies comp1 < comp3. This property is used to maintain dependencies between different ordering constraints and to derive new constraints.

The next task is to enforce ordering amongst siblings. Consider two siblings of the root of the component hierarchy, head and pin. The head should come before pin, head < pin, because pin will hinder head from being assembled. The reason pin hinders is because the space that will be occupied by the pin intersects with the space through which the head should traverse to be assembled to the pin. That is, the swept volume of the head whose axis vector is the path of the traversal intersects with the occupied volume of the pin as shown in Fig. 5.

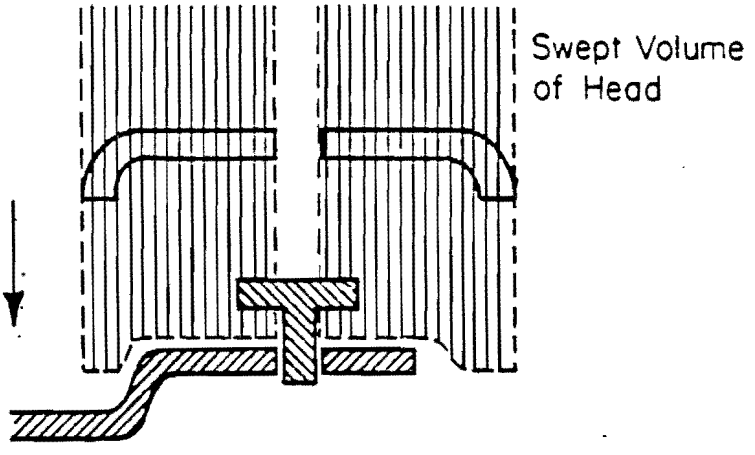

Fig. 5 Interference between Head and Pin

But between pin and body, there is no ordering since the swept volume of body does not intersect with pin in its assembled position as shown in Fig. 6a and the swept volume of pin does not intersect with body in Fig. 6b. In general, such interference checking involves two compound objects, groupings or subassemblies, so that a method of computing their swept volumes and occupied volumes is needed. An occupied volume of a compound object is a set union of its individual components, a standard operation of most solid modeling systems.

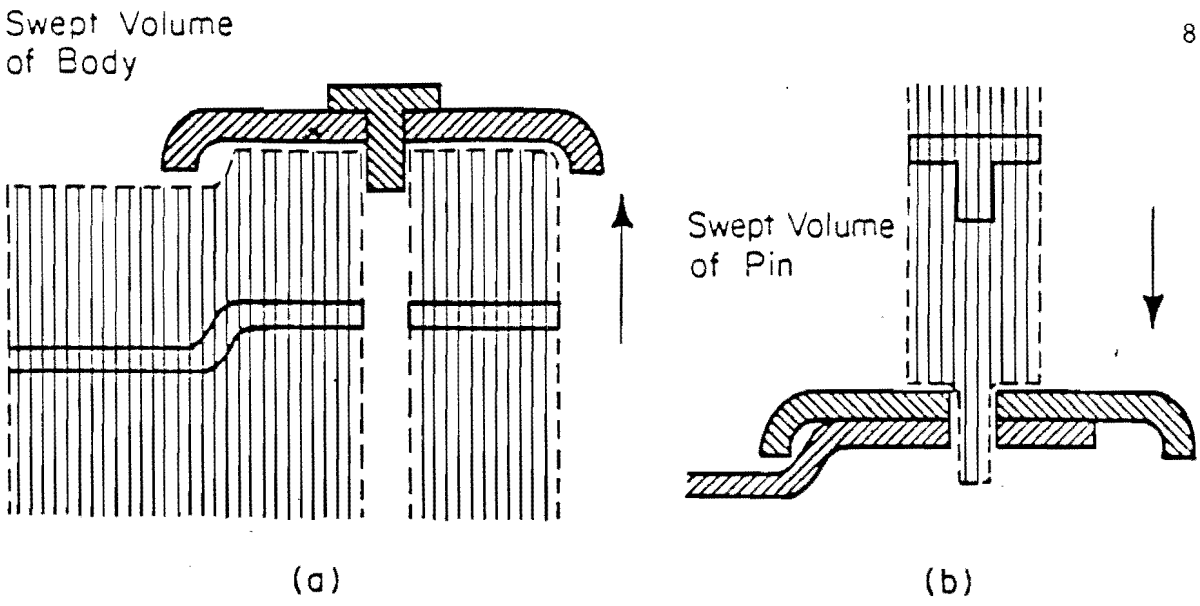

Fig. 6 Interference between Pin and Body

A swept volume of a compound object is computed from a union of two volumes: the space occupied by the object in its finally assembled configuration and the space through which the object traverses, a sweep volume. The first can be decomposed into two processes. One is the volume of the compound object which is a union of its individual components. The other is to determine the exact location of this volume to occupy in its finally assembled position. This is computed from mating conditions and shown in Lee [Lee, Andrew 1985].

The second volume is computed from a direction of the traversal of the compound object and its extent. The direction is specific to the mating conditions which the objects make with each other. For example, in Fig. 5 the head and body are related by against condition. The direction of the swept volume of the head to its assembled position is outward normal of the mating face of the head in against relation with the face of the body. But, when there are multiple against conditions such as in Fig. 2b, there are multiple normals to consider and computing the direction of the swept volume from them may yield an inconsistent result. For example, the direction of the swept volume of cover based on the against condition between face 1 of cover and face 4 of body is in positive x-axis but negative x-axis when considering face 2 of cover and face 3 of body and negative y-axis when considering face 3 of cover and face 5 of body. In this case, we can consider the area of the faces of the cover that are in against conditions with body and choose the normal of the face with the largest area as the direction of the swept volume Here face 3 of cover has the largest area and the direction of the swept volume of the cover is along the negative y-axis. We do not have a good theory of calculating the direction component of the swept volume in general.

The extent of the volume is from infinity to the assembled position of the grouping in the direction of the swept volume. This may be too strong when we actually integrate the

assembling procedure with robotic actions.

The cumulation of all the orderings for bell assembly with 29 ordering constraints is summarized in Fig. 7a. All the component names followed by "G" denote the grouping object, not the component. So terminal base G is composed of terminal base, ground terminal and plus terminal. When an ordering constraint is asserted, its transitive closure is computed with dependencies using truth maintenance system so that interference checking is done only when it is deducible from known ordering constraints. For example, rod1 G $<$ cover was derived from rod G $<$ terminal base G and terminal base G $<$ cover.

Using these orderings, we can derive a precedence graph as in Fig. 7b which uses the and-gate and buffer in analogy with the digital circuit. For and-gate, all the inputs to the gate have to be present to proceed to its output while buffer is a special case of and-gate where there are only one input.

Using swept volume for interference checking is similar to that of the findpath problem using configuration space by Brooks and Lozano-Perez [Brooks, 1983]. The difference is that they want to find a collision free paths while the environment stays invariant. We want to find the sequence of changes to the environment, assembling sequence, so that the shape of a collision free paths does not involve rotation. However, when the interference check between two objects is positive, they may not interfere although when the interference check is negative, they may not interfere with each other. Their findpath algorithm should be incorporated for sweeping volume to compute more sophisticated interference checking.

5. LOCALITY ASSUMPTION VIOLATION

Locality assumption is that components in one grouping are not likely to interfere with components in another grouping. There are two ways the assumption may fail. First, the component hierarchy is not a tree in general. This may result in two compound objects involved in interference checking share a common object. This is inherently contradictory since the shared object has to be stationary and swept at the same time. APG make the two compound objects disjoint by excluding the shared components from the compound object being swept. For example, consider a scale for measuring your height in Fig. 8, its mating graph is illustrated in Fig. 9a and its component hierarchy is given in Fig. 9b.

(< HEAD PIN)(< TERMINAL-BASEG BODY)(< TERMINAL-BASEG COVER)(< FIXERG BODY)
(< FIXERG COVER)(< FIXERG TERMINAL-BASEG)(< ROD2G BODY)(< ROD2G COVER)
(< ROD2G TERMINAL-BASEG)(< ROD2G FIXERG)(< ROD1G BODY)(< ROD1G COVER)
(< ROD1G TERMINAL-BASEG)(< ROD1G FIXERG)(< COVER BODY)(< COILDRUM2 BODY)
(< ROD2 BODY)(< COILDRUM2 ROD2)(< COILDRUM1 BODY)(< ROD1 BODY)(< COILDRUM1 ROD1)
(< HAMMER BODY)(< FIXER BODY)(< HAMMER FIXER)(< PLUS-TERMINAL BODY)
(< GROUND-TERMINAL BODY)(< TERMINAL-BASE BODY)(< PLUS-TERMINAL TERMINAL-BASE)
(< GROUND-TERMINAL TERMINAL-BASE)

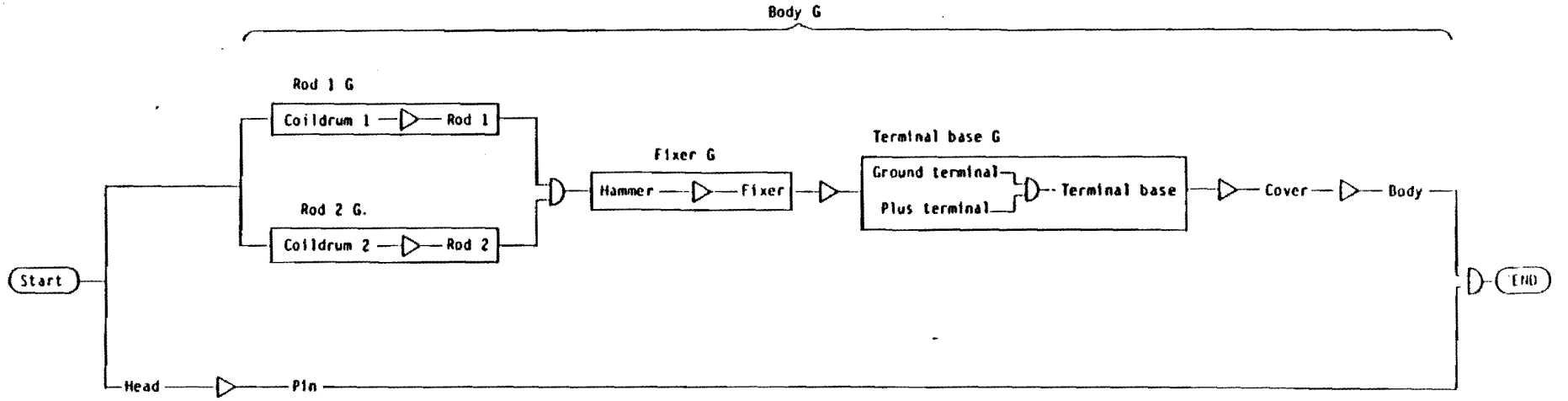

Fig. 7 Component Orderings and Precedence Graph of Bell Assembly

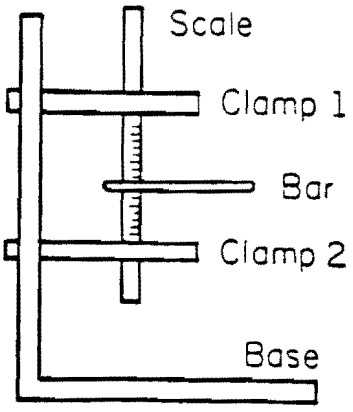

Fig. 8 Scale Assembly

Note both clamp 1 G (clamp1 and base) and clamp 2 G (clamp2 and base) contain the base. When clamp 1 G is to be swept for the interference checking against clamp 2 G, we sweep clamp 1 instead. Same for clamp 2 and discover clamp 2 < clamp 1 G.

The second type of failure occurs when there are simultaneous interferences between objects. We need to identify the culprit by systematically breaking down the object into parts by going down the hierarchy until an ordering can be found. Consider scale G with bar. Scale G is composed of clamp 1 G, clamp 2 G and scale. The culprit is sought by comparing clamp 1 G, clamp 2 G and bar. (Scale is left out so that when there are not culprits found, scale is to blame.) Compare clamp 1 G with bar and find the simultaneous interferences again. Clamp 1 G is broken down further so that base and bar is compared. There is no interferences between base and bar. Since clamp 1 G and bar are interfering and base is not the reason, clamp 1 is the

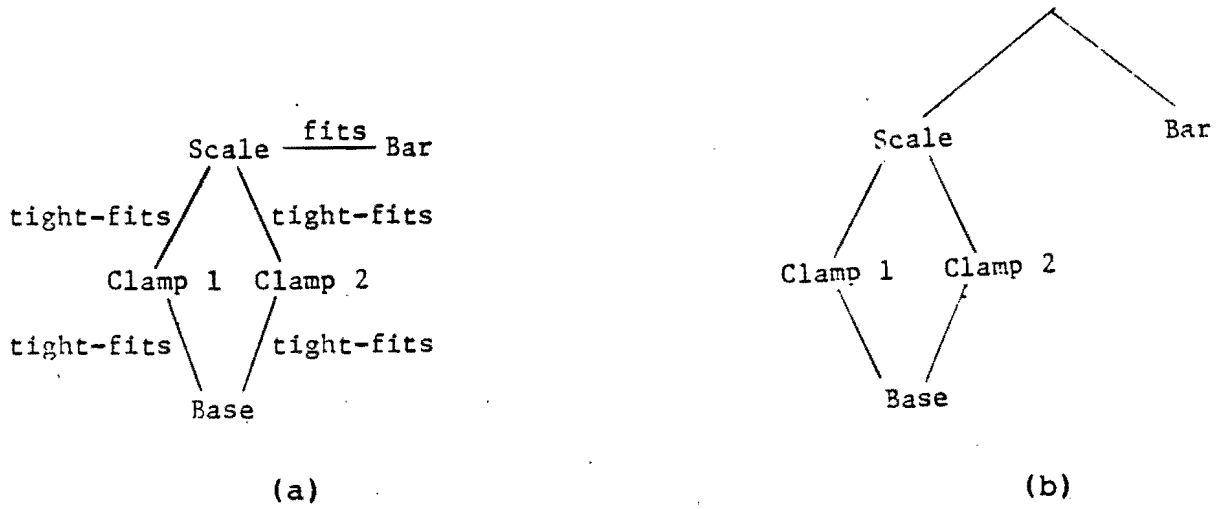

Fig. 9 Scale Assembly Structures

culprit: i.e., bar < clamp 1.

6. FURTHER RESEARCH

The reported work is an ongoing research. Many avenues of further research were identified from this work.

Each mating condition allows only a limited set of freedom of movements. Symbolic kinematic analysis is possible using constraint propagation technique since the mating conditions of a virtual link specify the set of possible movements and the virtual links specify mutual constraints between components. For example, in Fig. 2a both body and head have fits relation with the pin so there is an implied fits condition between the holes of body and head. But note the body and head can not move along the axes because of the against relation between them. Conversely, the implicit fits condition excludes the sliding motion between body and head.

A planning and execution monitoring system using component orderings generated by APG should be developed. The execution monitoring system might discover expectation failure of the component orderings. Since APG maintains the dependencies between orderings, it can process the conflicting candidates and "learn" from the failure so that the notion of "assemblibility" can be enriched iteratively.

7. CONCLUSIONS

A case study of the bell assembly with 13 components for generating component orderings was successfully performed.

Hierarchical structuring makes use of the locality principle of kinematic relationships so that the number of comparisons necessary are greatly reduced without any loss of completeness. That is, if there is an ordering discovered in exhaustive comparisons, the procedure using the hierarchical structure will discover it, too. This is because when the locality assumption fails so that components within the compound objects being compared interact, the procedure backtracks and searches for the culprit among only the relevant ones.

A crude analysis of complexity of two methods is done. Suppose the hierarchy is a balanced binary tree and there were no interactions mentioned in the section 5. If we have 2^n components, we have 2^n number of leaves in the tree. The number of interference checkings using the locality principle is twice as many as the number of internal nodes of the tree, $2 * (2^n-1)$. The factor of 2 comes from the fact that for every comparisons of two objects, two interference checks are needed: one object sweeps while the other is stationary and vice versa. On the other hand, it takes $2^n * (2^n - 1)$ comparisons in the exhaustive case. That is, with N components, the

number of comparisons in the exhaustive case takes $N/2$ times as much as in the hierarchical case. This is a significant saving in serious applications where an assembly has thousands of parts.

APG makes good use of various levels of representation, the mating conditions. Hierarchical structuring uses more abstract information such as fits, against, tight-fits and contact conditions and computing swept volume of an object makes use of more detailed information such as its surface normals.

Currently, APG produces ordering constraints between components from the specification of the final structure of the assembly. Although the implementation of the planner is not complete, these constraints should prove very useful for a planner especially in ordering conjunctive goals: the goal interactions in [Sussman 1975] , "A goal clobbers a brother goal.", can be avoided.

An meaningful integration of spatial reasoning with a planning system is suggested via component orderings. Previously, the knowledge of actions was the main source of power in most planning systems. Future planning systems should be more object oriented and make use of commonsense reasoning.

Our work is related to task-level programming. The task specification involves specifying the configuration of all the objects in the task. Our component ordering can be seen as a partial task specification where only interesting components and their configurations are specified so that the full configuration of all the objects can be changed dynamically at execution time.

The component orderings are partial. The task specification should also be partial so that a parallel operations are possible.

It is a common practice that a designer creates a model of a mechanical component on a CAD system and the commands for a numerical controlled machine are directly generated to manufacture the component. By analogy, the design activity will be greatly improved if an assembling procedure were generated directly as a designer creates a model of an assembly of components. When APG is integrated with a robot planning system, a robot will assemble the components as a designer models an assembly on the screen.

REFERENCES

1. Allen, J. F. *Maintaining Knowledge about Temporal Intervals.* **Communication of the ACM** (Nov. 1983) vol. 26.

2. Brooks, R. A. and Lozano-Perez, T. "A Subdivision Algorithm in Configuration Space for Findpath with Rotation", Proceedings of the Eighth Internation Joint Conference on Artificial Intelligence, 1983, pp. 799–806.

3. Fikes, R. E. and Nilsson, N. J. *STRIPS: A New Approach to the Application of Theorem Proving to Problem Solving.* **Artificial Intelligence** (1971) vol. 2, pp. 189–208.

4. Lee, K. and Andrew, G. *Inference of the Positions of Components in an Assembly: Part 2.* **Computer Aided Design** (Jan. 1985) vol. 17, pp. 20–24.

5. Lee, K. and Grossard, D. C. *A Hierarchical Data Structure for Representing Assemblies: Part 1.* **Computer Aided Design** (Jan. 1985) vol. 17, pp. 15–19.

6. Lieberman, L. I. and Wesley, M. A. *AUTOPASS:An Automatic Program System for Computer Controlled Mechanical Assembly.* **IBM J. Res. Dev** (July 1977) pp. 321–333.

7. McDermott, D. *A Temporal Logic for Reasoning About Processes and Plans.* **Cognitive Science** (1982) vol. 6, pp. 101–155.

8. Mortenson, M. E. *Geometric Modeling.* In: **Geometric Modeling.** John Wiley and Sons, New York, 1985, pp. 469–477.

9. Sussman, G. J. **A Computer Model of Skill Acquisition.** American Elsevier Publishing Company, Inc., New York, 1975.

10. Wesley, M. A., Lozano-Perez, T., Lieberman, L. I., Lavin, M. A. and Grossman, C. D. *A Geometric Modeling System for Automated Mechanical Assembly.* **IBM J. Res. Dev.** (Jan. 1980) vol. 24, pp. 64–74.