Symmetry Constraint Inference in Assembly Planning — *Automatic Assembly Configuration Specification**

Yanxi Liu and **Robin J. Popplestone**
Laboratory for Perceptual Robotics
Department of Computer and Information Science
University of Massachusetts at Amherst, 01003 USA
liu@cs.umass.edu, pop@cs.umass.edu

Abstract

In this paper we shall discuss how to treat the automatic generation of assembly task specifications as a constraint satisfaction problem (CSP) over finite and infinite domains. Conceptually it is straightforward to formulate assembly planning in terms of CSP, however the choice of constraint representation and of the order in which the constraints are applied is non-trivial if a computationally tractable system design is to be achieved. This work investigates a subtle interaction between a pair of interleaving constraints, namely the kinematic and the spatial occupancy constraints. While finding one consistent solution to a general CSP is NP-complete, our work shows how to reduce the combinatorics in problems arising in assembly using the symmetries of assembly components. Group theory, being the standard mathematical theory of *symmetry*, is used extensively in this work since both robots and assembly components are three-dimensional rigid bodies whose features have certain symmetries. This forms part of our high-level robot assembly task planner in which geometric solid modelling, group theory and CSP are combined into one computationally effective framework.

Introduction

A robot task level assembly planner requires, as part of the input, the final assembly configuration to be described (Fu 1987, Hutchinson & Kak 1990, Lozano-Pérez 1982). Even at the abstract level, planning the sequence of assembly also requires the relationships among assembly components to be specified (Homem De Mello 1989, De Fazio & Whitney et al 1989). However from a mechanical design it is not always trivial to derive an assembly configuration specification that is complete and unambiguous. One element of this problem arises from the symmetries of assembly components. For a component of cubic shape, for example, there are six equivalent single surfaces, eight equivalent vertices and twelve equivalent edges; without a proper representation of symmetry a complete specification of any type of contact with such a component will be tedious. There is an essential relationship between symmetry and function in mechanical engineering, which needs to be addressed in any system that 'reasons' about mechanically engineered entities.

In this paper we consider the process of determining a specification for assembly configurations from a design in a way that exploits the symmetries present in a 3-d body, that is computationally tractable, and that places a minimal load on the human user. Our goal is to incorporate this planning work into a integrated robotic assembly system such as Handey (Lozano-Pérez et al 1987), but one that has an intimate comprehension of symmetries not restricted to polyhedral bodies.

A constraint satisfaction problem (*CSP*) is defined over finite discrete domains in (Mackworth 1985) as:

> Given a set of n variables each with an associated domain and a set of constraining relations each involving a subset of the variables, find all possible n-tuples such that each n-tuple is an instantiation of the n variables satisfying the relations. Each of such n-tuples is a *solution.*

Assembly planning is a problem of *the assignment of values to variables subject to a set of constraints* (Dechter & Pearl 1987), therefore can be formulated as a CSP. Such constraints may include:

Kinematic constraints: A kinematic constraint specifies the contact between a pair of bodies, albeit one that may not pertain to every situation occurring during a particular assembly. Contacts between surfaces of simple shape can be treated as kinematic pairs (Angeles 1982, Herve 1978).

Spatial constraints: no two bodies occupy the same volume of Euclidean space at the same time; such spatial constraints are bounded (in C-space (Lozano-Pérez 1983)) by kinematic constraints.

Static and dynamic constraints: These relate to the forces acting on bodies during assembly. Subassemblies must be stable under gravity, friction must

*Preparation of this paper was supported in part by NSF grant number IRI-8709949, in part by ONR grant number N00014-84-K-0564, and University Research Initiative Grant N00014-86-K-0764.

be overcome when necessary, and material must not be overstressed.

Temporal constraints: These constrain the order in which actions may occur. They depend primarily on the previous constraints.

One of the difficulties in robotics research is the interleaving of the application of different constraints. It is standard practice in the analysis of mechanical systems to treat the kinematics before the dynamics of the system — indeed the conceptual dependencies and the possibility that the assembly may be performed either in space or on earth demand this. The work described here is concerned with establishing this kinematic basis. The problem we are facing is this:

Given : a set of 3-d rigid bodies and a set of relationship requirements upon bodies (implicit kinematic constraints), such as *body A fits body B* or *body C meshes body D*;

Find: all the possible assembly configurations in terms of mating features between bodies and relative positions of bodies, thus making kinematic constraints explicit such as *surface* 1 *of A fits surface* 3 *of B*;

Satisfy: the kinematic and spatial constraints.

We shall call this the *assembly constraint satisfaction problem* (ACSP) to distinguish it from the general CSP. Each given relationship on bodies in ACSP is a non-instantiated kinematic constraint that corresponds to a variable of CSP. The domain associated with each variable is composed of those candidate mating feature pairs that satisfy the necessary mating feature conditions described in (Liu & Popplestone 1989). In (Liu & Popplestone 1989) we reported in detail how to find candidate mating features from a set of geometric boundary models of bodies by matching against a prominent-feature library. In this paper we assume that such mating feature pairs have been found, as have the relationships between each pair of mating features. These relationships are described in terms of symmetry groups and are called the *symmetry constraints.* The possible mating features form domains of finite sizes, thus ACSP appears to be a CSP over finite sets. However each solution of ACSP, a set of mating feature pairs defining an assembly configuration, also has to satisfy the spatial constraint, whereby we have to know the relative positions of bodies subject to the symmetry constraints, and this is an embedded CSP over possibly infinite domains.

Besides the interleaving of different constraints, our work differs from that of the SPAR system (Hutchinson & Kak 1990) in that a constraint between bodies is expressed as the relative location of the bodies belonging to a (possibly infinite) generalized coset of feature symmetry groups. Our work differs from that of RAPT (Ambler & Popplestone 1975, Popplestone, Ambler & Bellos 1980) in two ways, one is the incomplete kinematic constraint input and the other is that instead of mapping symbolic representation for spatial relationships to a set of algebraic equations, symmetry groups of the mating feature pairs are generated first (Liu & Popplestone 1990,Popplestone, Liu & Weiss 1990, Popplestone 1984). This provides greater computational tractability and a more uniform treatment of spatial relationships, plus the ability to treat multiple feature relationships through symmetry group intersections (Liu 1990, Popplestone, Weiss and Liu 1988).

Group theory, being the standard mathematical theory of *symmetry*, is used extensively in this work, since both robots and assembly components are three-dimensional rigid bodies whose features have certain symmetries. Some group theory related concepts and how to express relative positions of bodies using symmetry groups of features are discussed in the next section. It is known that in a general *CSP* finding one consistent solution is an NP-complete problem. In ACSP clues deduced from the symmetries of the assembly are used to minimize the variable domain, thus mitigating the combinatorics, as described in Section "Symmetry constraint and constraint satisfaction graph". In the final section we present two examples and discuss the pros and cons of this work.

Symmetry groups and Relative positions

A *group* G can be regarded simply as a set of mappings that are closed under an associative composition operation, where $1 \in G$ is an identity element such that for all $g \in G, 1g = g1 = g$, and where each element g has an inverse g^{-1} such that $gg^{-1} = g^{-1}g = 1$. A *subgroup* H of a group G is a subset of G such that H is itself a group under the composition operation of G. If H is a subgroup of G and $g \in G$ then a left *coset* of H is the set $gH = \{gh | h \in H\}$. If $g_1, g_2 \in G$, a *two-sided coset* of H is the set $g_1 H g_2 = \{g_1 h g_2 | h \in H\}$. For more details on groups and their applications refer to (Miller 1972, Popplestone, Liu & Weiss 1990). One important *group* to which researchers in robotics can easily relate is the *proper Euclidean group* $\mathcal{E}^+$, which is composed of all 3-d rotations, translations and their combinations. Every element of $\mathcal{E}^+$ can *act* on a subset of the Euclidean space $\Re^3$, thereby *relocating* the subset. For each point x in $\Re^3$ an *orbit* of x under a subgroup G of $\mathcal{E}^+$ is defined to be the set $G(x) = \{g(x) | g \in G\}$. This is an important and useful concept.

Bodies in an assembly mate through their features. A *primitive feature* F of a solid is a surface which is a subset of $\Re^3$ associated with a boundary model of the solid. Examples are infinite planes, cylinders and spheres. A *compound feature* is a set of primitive features. A feature, primitive or compound, may have *symmetries.* A symmetry of a feature is an element of $\mathcal{E}^+$ that leaves the feature set-wise invariant. The set of all the symmetries of a feature has a group structure and is called the *symmetry group* of the feature. When two features from different bodies *mate* the relationship of the two bodies can be expressed in terms of the symmetry groups of the mating features. In the case

Table 1: Continuous Group and Degree of Freedom

Dimension (d.o.f.)	Symmetry Group (constraint)	Associated Lower pair
1	T^1	*Prismatic*
1	$SO2$	*Revolute*
1	$\mathcal{G}_{screw}$	*Screw*
2	$\mathcal{G}_{cylinder}$	*Cylindrical*
3	$\mathcal{G}_{plane}$	*Planar*
3	$SO3$	*Spherical*

when they have an areal contact, i.e. a *fitting* relationship, the symmetry groups of contacting features are identical. In this paper such fitting relationships are our primary concern. We call such a symmetry group a *symmetry constraint* of the mating feature pair. By associating with each feature its symmetry group, it is possible for the planner to represent and reason about symmetry in an efficient way. This is of essential importance in understanding how features mate and what the final assembly configurations can be.

Symmetry groups can be either finite or infinite[1]. An example of a finite group is the symmetry group of a hexagonal bolt-head, which is called the cyclic group of order 6. Some examples of infinite groups are listed in Table 1. Each of these symmetry groups corresponds to a *lower pair*, which is a kinematic coupling with areal contact (fitting). Our work treats both infinite and finite cases in a uniform way.

Consider what we can infer about the *relative location* of two bodies that have two features in contact. Such a contacting relationship must correspond to a set of displacements which specify the relative location of the two bodies. Note that a displacement as a mapping over $\Re^3$ is a member of $\mathcal{E}^+$. Now let B_1 and B_2 be two bodies, with primitive features F_1 and F_2 which are in contact and have symmetry groups $sym(F_1), sym(F_2)$ respectively. Suppose l_1, l_2 specify the locations of bodies B_1, B_2 in the world coordinate system and f_1 and f_2 specify the locations of features F_1, F_2 in their respective body coordinates. By the definition of symmetry group, it is clear that if we move B_1 or B_2 by a member of the symmetry group $sym(F_1)$ or $sym(F_2)$ respectively, the relationship between the features is preserved. The *fits* relation is particularly constraining. If F_1 *fits* F_2 then their surfaces coincide and thus have the same symmetry group. The relative location of body B_1 to body B_2 can be expressed as:

$$l_1^{-1}l_2 \in f_1 sym(F_1) f_2^{-1}. \qquad (1)$$

We can summarize this by saying that *if a primitive feature of one body fits a primitive feature of another body then the relative location of the two bodies is a two-sided coset of the common symmetry group of the features.* This coset is an infinite set when the symmetry group is of infinite order.

In the case where $sym(F_1)$ is the identity group $\{1\}$, from formula (1), we know that the relative position of bodies B_1 and B_2 is uniquely determined:

$$l_1^{-1}l_2 = f_1 f_2^{-1}. \qquad (2)$$

This shows that the most asymmetrical case appears in the simplest form under this formulation. Two bodies in an assembly are typically related to each other through multiple primitive features. If the above two bodies are related by fitting two pairs of features, i.e. F_{11} fits F_{21} and F_{12} fits F_{22} with feature locations in their body coordinate systems $f_{11}, f_{21}, f_{12}, f_{22}$, then the relative location of body B_1 to body B_2 should be constrained by both relations expressed in the form (1) *simultaneously*. Equivalently, it should be in the intersection of the two sets:

$$l_1^{-1}l_2 \in f_{11} sym(F_{11}) f_{21}^{-1} \bigcap f_{12} sym(F_{12}) f_{22}^{-1}. \qquad (3)$$

This illustrates the use of kinematic constraints on mating features to find possible relative body locations, and provides a framework for determining relative positions when two bodies are mated through multiple features.

Constraint Satisfaction Process

In this section we describe in principle how ACSP is treated and in the next section we shall reinforce this principle through examples. As pointed out in the introduction, the given requirements among bodies of an assembly are treated as variables in *ACSP* and the possible mating features form the domain for each variable. If the number of variables is N, the domain size is D (average), and the number of constraints is C, then the search space would be on the order of $O(CD^N)$. This shows the importance of reducing the domain size D. Methods employing local arc/path-consistency and heuristics are used in searching for a solution to mitigate the combinatorics.

Input: n relationships among bodies (variables in CSP), each of which is associated with a set of candidate mating features (the domain of the variable).

Output: All the assembly configurations that the relative positions of bodies can be determined and that satisfy the kinematic and the spatial constraints.

In the following we describe a three-step-algorithm for finding a solution in ACSP:

Step one is setting up the constraint satisfaction graph. Given the non-instantiated kinematic constraints (input) and all the possible mating features, the system establishes a two-layered graph. The top layer has bodies as nodes and required relationships among bodies as arcs (this is a dual graph of the actual constraint satisfaction graph). The bottom layer

[1] For a list of important symmetry group definitions refer to (Popplestone, Liu & Weiss 1990, Liu & Popplestone 1990)

has candidate mating features as nodes and the common symmetry groups between mating features as arcs. Whenever the domain size of one of the variables is zero the process terminates. Some primitive features such as cylindrical surface, surfaces with texture *threads*, or *gears* have higher priority to be matched first. Because such surfaces with given parameters are relatively rare, so the probability of being correctly matched is higher. Whenever two candidate mating features have the *identity* group as their common symmetry group, or the common symmetry group is the same as the symmetry group of one of the bodies to which the features reside, the relative position of the two related bodies can be uniquely determined and the *intersect* predicate of PADL2[2] (Brown 1982) can then be called for spatial occupancy checking. In this case, node-consistency is achieved locally.

Step two is reducing the domain size of each variable. The following set of rules are applied in turn:

- **Mandatory request law**: if there is *only* one correspondence at the bottom level to a upper layer input constraint this mating feature pair has the highest priority. Therefore if one of the features is a non-**sharable** feature[3], then any other arcs pointing to this feature should be deleted. The precondition of this rule corresponds to the case in *CSP* where the domain size of a variable is one. The effect of applying this rule can be propagated until the network is quiescent.

- **Equivalent feature law**. A pair of features of a body is said to be *equivalent* if they have the same symmetry group and same dimensions and belong to the same *orbit*.

  A set of features $F_1, F_2, ...F_n$ of body B belong to the same *orbit* if and only if there exists $g \in G$ (the symmetry group of the body) such that $g(F_i) = F_{i+1}, g(F_n) = F_1$. Since $g(B) = B$, the features permute among themselves under g while the body sits still set-wise. Being able to represent and reason about these characteristics the system will choose only one out of a set of equivalent features. The reasoning for the chosen one can be applied to the rest of the features in its orbit. For example, consider a finite cylinder which has two identical ends: all the reasoning about one end of the cylinder applies to the other end of the cylinder. This rule shall be applied unless it causes the termination of the process.

- **Arc/Path consistency checking** (Mohr & Henderson 1986, Mackworth 1985) is a local consistency checking method for CSPs which can be used to eliminate those arcs which will not lead to a conclusion.

Step three is checking the spatial constraint globally. Firstly, we need to find all the subgraphs in the bottom level (relating mating features) which are isomorphic to the top level graph (relating bodies). Each of these subgraphs is an instantiation of the input requirements on bodies. Secondly, the no-pairwise-interference constraint is applied. When the relative position of a pair of related bodies cannot be determined locally in step one, this is the time to evaluate the situation globally i.e. to see whether constraint from other bodies can be used to decide the position. we have experimented with a simple and quick algorithm due to (Kramer 1990) that solves certain algebraic equations geometrically. This will be explained further in the examples.

Complexity analysis of the above algorithm:

Step one takes $O(nd)$ time where n is the number of relationships among bodies and d is the domain size.

Step two uses the arc/path consistency checking algorithms described in [21] which has complexity $O(nd^2)$ where d is the bound for domain sizes.

Step three needs to check $s_1 \times s_2 ... \times s_n$ combinations where n is still the number of kinematic constraints and s_i is the *reduced* domain size.

In summary, the planner starts by establishing the two layered constraint satisfaction graph. Whenever possible, node consistency checking is carried out. The feature-level graph is pruned by applying heuristics, arc/path consistency checks. Finally, isomorphic subgroups are mapped from the feature-level to the body-level where the spatial occupancy constraint is applied to each. The original network is thus transformed into smaller and smaller networks such that the final isomorphic graph matching can be performed on a much slimmer graph.

Examples and Discussion

In this section we describe and compare three examples where different approaches are taken to determine relative positions so that the spatial occupancy constraint can be applied.

Example one

Each component of this assembly is shown in Figure 1. The input goals are:

```
[goal 1 [fit b1 b4]]
[goal 2 [fit b2 b4]]
[goal 3 [fit b3 b4]]
```

The associated compound features for each body:

```
b1 : comp1 = [1 3 4], ...
b2 : comp1 = [1 3 4], ...
b3 : comp1 = [1 3 2], ...
b4 : comp1 = [7 8 9], ...
     comp12 = [7 10 13]
     comp13 = [7 11 12]
```

[2] A solid geometric modeller PADL2 has been interfaced with our planning system. Its *intersect* predicate returns *true* if no interference of two specified bodies is found, *false* otherwise.

[3] A predicate called **sharable** is applicable to a pair of features from distinct bodies. It returns *true* if a third feature can also be mated simultaneously, *false* otherwise.

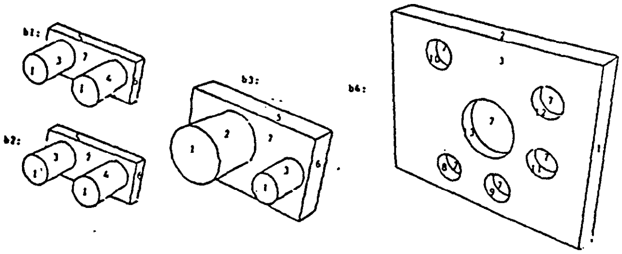

Figure 1: First assembly example

All of the mating feature pairs found from these sets are non-sharable. At step one, the symmetry group computed for each of these compound features by the system is the identity group, thus the relative position of the two bodies on which the mating features reside can be determined uniquely. The planner applies the spatial occupancy constraint on all pairs of mating bodies by using the PADL2 *intersect* predicate. Then a node consistent network is achieved. At step two, intersection is found by arc consistency checking (Figure 2). The process terminated since at least one domain is empty. After replacing to another set of assembly components, two solutions are found at the end of step three i.e. the spatial constraint is satisfied globally.

```
** [Or  [And [fit [b1 comp1] [b4 comp13]]
             [fit [b2 comp1] [b4 comp1]]
             [fit [b3 comp1] [b4 comp12]]]
        [And [fit [b1 comp1] [b4 comp1]]
             [fit [b2 comp1] [b4 comp13]]
             [fit [b3 comp1] [b4 comp12]]]]
```

See Figure 2. Since $b1$ and $b2$ each has two compound features belonging to the same feature orbit, there are actually eight different assembly configurations found.

Example two

The assembly components of this assembly are shown in Figure 3. The input goals are:

```
[goal 1 [fit b3 b1]]
[goal 2 [fit b3 b2]]
[goal 3 [fit b1 b2]]
```

The compound features of each body are composed of these primitive features:

```
b1 : comp1 = [7 8]
     comp2 = [9 10]
b2 : comp1 = [7 8]
     comp2 = [9 10]
b3 : comp1 = [7 9]
     comp2 = [7 8]
```

All the potential mating feature pairs are non-sharable. All the common symmetry groups of mating features are $SO(2)$, which corresponds to a revolute joint. Since $SO(2)$ is an infinite group, node/arc consistency checking in *step two* cannot be applied immediately. Although *comp1* and *comp2* of $b3$ are equivalent features but dismissing one

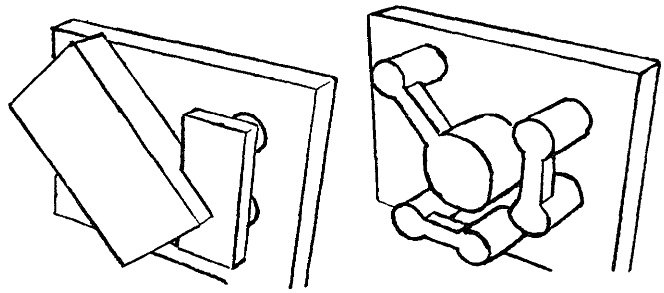

Figure 2: A rejected solution and a confirmed solution of Example one

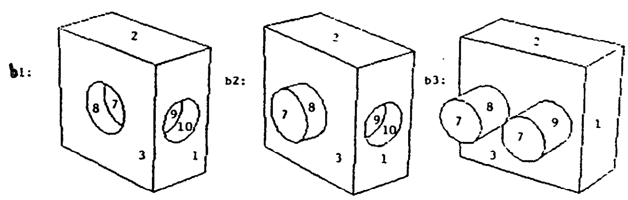

Figure 3: Second assembly example

would cause the process to terminate so they are preserved. At the step three, the following configurations are proposed after subgraph mapping:

```
** [Or  [And [fit [b3 comp1] [b1 comp2]]
             [fit [b3 comp2] [b2 comp2]]
             [fit [b1 comp1] [b2 comp1]]]
        [And [fit [b3 comp2] [b1 comp2]]
             [fit [b3 comp1] [b2 comp2]]
             [fit [b1 comp1] [b2 comp1]]]]
```

Although the exact relative position of $b3$ to $b1$ is unknown from their mating features, the symmetry constraint $SO(2)$ restricts $b1$ to having only one degree of freedom when it is fitted to $b3$: the rotation about the central axis of their mating features. As a matter of fact, every body in this assembly is constrained by two such revolute joints. Any pair of points coinciding on the mating feature surfaces should have the same orbit under the action of the common symmetry group of the feature pair. We picked the center point at the bottom of a cylinder, concave or convex, as the representative point for a cylindrical mating feature. This is a good choice for cylinders since its orbit under the symmetry group of the feature is itself (it

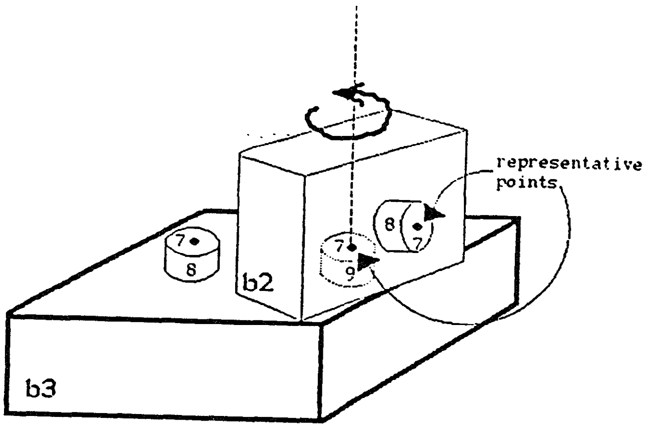

Figure 4: When $b2$ is fitted to $b3$, the representative points of the fitted features coincide, the orbit of the other representative point forms a locus under this revolute motion

is invariant under actions by any member of the symmetry group). Therefore the two representative points of a pair of mating features should be coincident all the time when the bodies fit (Figure 4). Figure 5 shows that the orbit of the representative point on feature *comp*1 of $b2$ forms a locus under the symmetry group of feature *comp*2 of the same body, and same is illustrated for the representative point of feature *comp*1 on body $b1$. The intersection of these two loci is the only possible position for the representative points of *comp*1 of $b1$ and *comp*1 of $b2$ to be coincident. Thus it uniquely determines the relative positions of bodies $b1$ and $b2$ such that both mating relationships can be satisfied simultaneously. Finally, the spatial constraint is applied and both sets of possible mating feature pairs are confirmed to be solutions.

Example Three

Figure 6 shows an assembly containing a mating feature pair of triangle shape. This is the case where the symmetry group of the mating feature is the same symmetry group as the symmetry group of the body thus the relative position can be uniquely determined. Due to length limit, we omit the detail here.

Discussion

In conclusion, this paper describes the use of constraint satisfaction networks to reify assembly configurations which are composed of possible mating feature pairs related by symmetry constraints, i.e. the symmetry groups shared by the mating feature pairs. This work has been implemented in POPLOG on the SUN workstation under OS 4.0. Through this work one can see that the performance of a special CSP depends strongly on how well the domain dependent subproblems have been solved. The domain dependent constraint application: finding possible mating

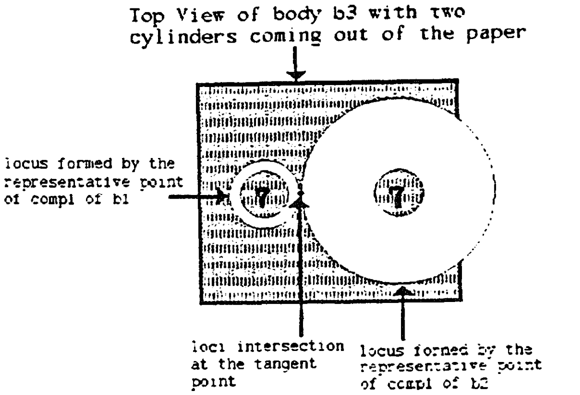

Figure 5: Intersecting loci from *comp*1 of $b1$ and *comp*1 of $b2$

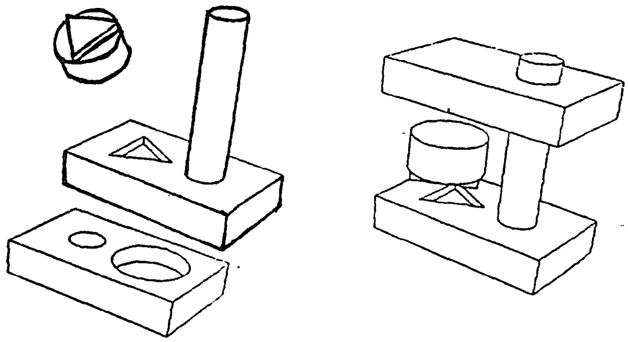

Figure 6: An example containing non-revolute joint

features, finding positions, finding intersections etc., dominates the implementation as well. This seems to suggest that an effective combination of artificial intelligence and robotics must be woven together at each problem solving step.

The bottleneck of ACSP is the determination of the relative position of bodies so that the spatial constraint can be applied. The usefulness of symmetry constraints is reflected in the fact that when the identity group is present, the relative position can be computed uniquely. When no identity groups are present, we can apply the representative-point-intersection method to find possible locations or to claim that no solution exists. Things become more complicated when a symmetry constraint network is underconstrained, i.e. relative motions are allowed in an assembly for certain functional purposes, such as a pair of scissors. One plausible way to deal with this is simply to sweep the body concerned under the symmetry group of its constantly contacting features, then intersect this swept volume with the rest of the assembly. If no intersection is detected, the kinematic constraint is correctly instantiated, otherwise the intersected volumes set up boundaries (or intervals, label set [5]) on the kinematic constraint. PADL2 does not have the ability to sweep a body therefore we have not yet experimented with these.

Our method cannot be complete even for assemblies with no residual degrees of freedom since it is possible to define assemblies that require the solution of a polynomial equation of arbitrarily high degree (Hopcroft, Joseph & Whitesides 1987), and the geometric methods we use cannot generate solutions of such equations. However in any practical assembly, only a small portion of the assembly planning problem will require the solution of such equations; the majority of the relationships should be amenable to analysis by our methods. Our system could well be complemented by harnessing it to a system capable of treating a full range of polynomial equations as envisaged by Canny [4].

Acknowledgment

Thanks Gordon Dakin for implementing the POPLOG to PADL2 interface. Thanks King Shaw and Seth Malitz for some helpful discussions.

References

[1] Ambler, A.P. and Popplestone, R.J. 1975. Inferring the positions of Bodies from specified Spatial Relationships. *Artificial Intelligence* 6(1975): 157–174.

[2] Angeles, J. 1982. Spatial Kinematic Chains: Analysis-Synthesis-Optimization. Springer-Verlag, New York.

[3] Brown, C.M. 1982 PADL2: A Technical summary. *IEEE Computer Graphics Appl.* 2(2):69–84.

[4] Canny, J. 1987. The complexity of Robot Motion Planning. Ph.D. diss. Dept. of Electrical Engineering and Computer Science. MIT.

[5] Davis, E. 1987. Constraint Propagation with Interval Labels. *Artificial Intelligence* 32(3):281-331.

[6] De Fazio, R.L. and Whitney, D.E. 1987. Simplified Generation of All Mechanical Assembly Sequences. *IEEE Journal of Robotics and Automation* RA-3(6):640–658.

[7] Dechter R. and Pearl J 1987. Network-based heuristics for constraint-satisfaction problems. *Artificial Intelligence* 34(1): 1-38.

[8] Fu,K.S.; Gonzalez,R.C. and Lee, C.S.G. 1987. *ROBOTICS: Control, Sensing, Vision, and Intelligence*, McGraw-Hill Book Company.

[9] Hervé,J.M. 1978. Analyse Structurelle des Mécanismes par Groupe des Déplacements. *Mechanism and Machine Theory* 13(4):437–450.

[10] Homem de Mello, L.S. 1989. Task Sequence Planning for Robotic Assembly. Ph.D. diss., Dept. of Electrical and Computer Engineering, Carnegie Mellon University.

[11] Hopcroft J., Joseph, D. and Whitesides, S. 1987. Movement Problems for 2-Dimensional Linkages (chapter 12). Schwartz J.T., Sharir, M. and Hopcroft, J. (Eds.). Planning, geometry and complixity of robot motions. Norwood, N.J.: Ablex Pub. Corp.

[12] Hutchinson, S.A. and Kak, A.C. 1990. SPAR: A Planner That Satisfies Operational and Geometric Goals in Uncertain Environments. *Artificial Intelligence Magazine* 11(1):30–61.

[13] Kramer, G.A. 1990. Solving geometric constraint systems. In Proceedings of AAAI-90. Boston, MA.

[14] Liu, Y. 1990. A Justification for the Characteristic Invariant Representation of TR Subgroups of the Proper Euclidean Group. to be presented at the fifth SIAM Conference on Discrete Mathematics, June 11 -14, Atlanta, Georgia.

[15] Liu, Y. and Popplestone, R.J. 1989. Assembly Planning from Solid Models. In Proceedings of *IEEE International Conference on Robotics and Automation*, 222-227. Washington, D.C.: IEEE Computer Society Press. Also: Assembly Planning from Solid Models using Symmetry Groups. *Progress in Robotics and Intelligent Systems.* Ho, C.Y. and Zobrist, G. (Eds.), Ablex Publishing Corporation, New Jersey. Forthcoming.

[16] Lozano-Pérez, T. 1983. Spatial planning: a configuration space approach. *IEEE Transactions on Computers* C-32(2):108–120.

[17] Lozano-Pérez, T., Jones, J.L., Mazer, E., O'Donnell, P.A., Grimson, W.E.L., Tournassoud, P. and Lanusse, A. 1987. Handey: A Robot System that Recognizes, Plans, and Manipulates. In Proceedings of *IEEE International Conference on Robotics and Automation*, 843 – 849. Washington, D.C.: IEEE Computer Society Press.

[18] Lozano-Pérez, T. 1982. Task Planning. *Robot Motion: Planning and Control*, Ed. Michael Brady, et al. The MIT Press Series in Artificial Intelligence.

[19] Mackworth, A.K. and Freuder, E.C. 1985. The Complexity of Some Polynomial Network Consistency Algorithms for Constraint Satisfaction Problems. *Artificial Intelligence* 25(1985): 65-74.

[20] Miller,W. Jr. 1972. Symmetry Groups and Their Applications. *Academic Press.* New York.

[21] Mohr, R. and Henderson, T.C. 1986. Arc and Path Consistency Revisited. *Artificial Intelligence* 28(2):225–233.

[22] Popplestone, R.J., Liu, Y. and Weiss, R. 1990. A Group Theoretic Approach to Assembly Planning. *Artificial Intelligence Magazine* 11(1):82–97.

[23] Popplestone R.J. , Weiss R., and Liu Y. *Using Characteristic Invariants to Infer New Spatial relationships from Old.* In Proceedings of the 1988 IEEE International Conference on Robotics and Automation. Washington, D.C.: IEEE Computer Society Press.

[24] Popplestone, R.J. 1984. Group theory and Robotics. *Robotics Research, The First Int. Symp.*:55–64. M. Brady and R. Paul(Eds.). Cambridge, Mass.: MIT Press.

[25] Popplestone,R.J.; Ambler A.P., and Bellos I. 1980. An Interpreter for a Language for Describing Assemblies. *Artificial Intelligence* 14(1): 79–107.

[26] Whitney D. E. et al 1989. Tools for Strategic product Design. In Preprints of NSF Engineering Design Research Conference.