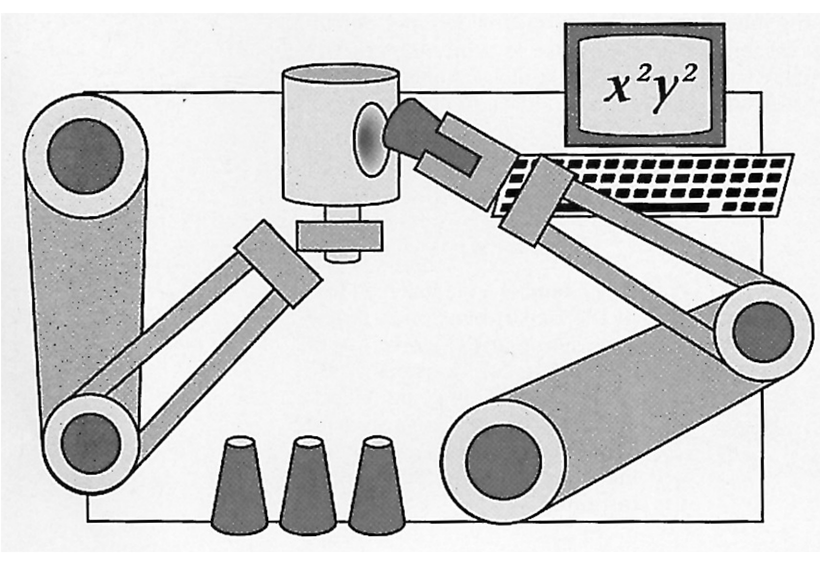
A Group Theoretic Approach to Assembly Planning

Robin J. Popplestone, Yanxi Liu, and Rich Weiss

High-level robotic assembly planning is concerned with how bodies fit together and how spatial relationships among bodies are established over time. To generate an assembly task specification for robots, it is necessary to represent the geometric shapes of the assembly components in a computational form. One of the principal aspects of shape representation that is relevant for assembly tasks is the symmetry of the shape. Group theory is the standard mathematical tool for describing symmetry. The interaction between algebra and geometry within a group theoretic framework has provided us with a unified computational treatment of reasoning about how parts with multiple contacting features fit together.

We treat robotic assembly planning on two distinct conceptual levels. Planning at the higher level involves deriving nominal trajectories along which the bodies to be assembled are to be moved. These trajectories are nominal in the sense that they would accomplish the assembly were we to have a perfect robot manipulating bodies whose shapes were perfectly accurate. Planning at the lower level transforms such a high-level specification into an assembly plan that takes account of uncertainty. In this article, we are primarily concerned with high-level robotic assembly planning.



An *assembly* is a collection of bodies that are spatially related. When two bodies in an assembly are related, they do not make contact over their whole surface; rather, features of each body are in contact. For the moment, let us think of a *feature* as some part of the surface of a body. Thus, for example, a journal on a shaft fits a bearing, the teeth of gears mesh, the threaded portion of the shank of a bolt fits a hole tapped in some body. All these relationships between bodies can be described as liaisons (Bourtjault 1984, De Fazio, and Whitney 1987). For the purpose of automatic generation of assembly plans, it is necessary to represent the shapes of assembly components, locations of components, and the spatial relations between bodies in a systematic computational form.

One of the important aspects of shape representation that is relevant to assembly planning and mechanical design is the symmetry of shapes and their features. A *symmetry* of a feature is simply a rotation or translation that maps this feature to itself.[1] In assembly planning, the symmetry of features is usually more important than the symmetry of the bodies to which the features belong because bodies are mated through their features. For example, consider a cylindrical hole in a block. This feature is mapped into itself by any rotation about the axis of the hole: Such rotations are, therefore, symmetries of the hole but not in general symmetries of the block. Suppose now that we wish to fit a cylindrical peg in the hole. Because the hole is symmetric, if the peg is in place, and we rotate it by any one of the symmetries of the hole, the fitting relationship is preserved. We see that the spatial relationships possible between two features strongly depend on the symmetries of the related features.

Symmetry has been formally studied by mathematicians since the work of Galois on the symmetries of the roots of equations in the early nineteenth century; his work is seminal to modern algebra. Group theory provides the modern treatment of symmetry and is an essential tool in physics and theoretical chemistry. Its use in the theory of mechanisms was pioneered by Hervé (1978). The basis of our work on spatial relations is in Popplestone (1984). Thomas and Torras (1988) make use of the product and intersection tables of Hervé (1978) in an implementation of a system for reasoning about spatial relations.

We can regard a *group* simply as a set of rigid transformations, that is, combinations of translations and rotations, that satisfy certain additional closure properties, as described in Group Theory and Spatial Reasoning. We see that the set of symmetries of a feature has the necessary closure properties to form a group. In our work (Liu and Popplestone 1989; Popplestone, Weiss, and Liu 1988), we made the symmetry group of a feature to be an important descriptor that provides us with a basis for systematizing the reasoning about spatial relations required in assembly planning.

In this article, we show how to compute the symmetry group of features of a body given a representation of its shape, determine whether two bodies can be assembled, and determine the freedom remaining in the relative position of two assembled bodies. In Locations as Rigid Transformations, we define rigid transformations and discuss their computational representation. This information is used in the next section, Spatial Relations, as the basis for expressing how bodies are spatially related in terms of possible relative locations. In Shapes, we discuss formalisms for representing the shape of bodies and how to express tolerances on shapes that provide the basis for reasoning about uncertainty. We conclude the exposition in Group Theory and Spatial Reasoning, where we develop the group theoretic approach to assembly planning. We start with a few important definitions of group theory, provide a formal definition of the symmetry group of a feature, and develop group theoretic constructs for sets of possible relative locations of bodies and show how to simultaneously satisfy multiple relations in this formalism. We conclude the article by stating some important results of our work and offering two examples in high-level assembly planning where the use of symmetries can be beneficial.

Locations as Rigid Transformations

In performing an assembly, it is necessary for a robot to move bodies (including itself) about in 3-space. This requires us to have some way of representing their locations. When we move a rigid body, the distance between any two points on the body remains the same. Thus, we are led to consider distance-preserving mappings of R^3 onto R^3. A mapping $g : R^3 \to R^3$ for which

$$||g(x) - g(y)|| = ||x - y||$$

is called an *isometry.* We can multiply two isometries using the operation of functional composition (which we write as juxtaposition), adopting the more traditional convention that $(fg)(x) = f(g(x))$ rather than the converse convention used by algebraists. The identity isometry, which we write as 1, is defined by $1(x) = x$. It is easy to show that any isometry g has an inverse g^{-1}, with the property that $g^{-1}g = gg^{-1} = 1$.

Isometries can be represented by 4*4 matrices, using homogeneous coordinates, as described in textbooks on robotics such as Fu, Gonzalez, and Lee (1987). With such a representation, function composition is represented by matrix product, and the application of an isometry to a member of R^3 is represented by premultiplying a column vector by the matrix representing the isometry.

Isometries, as defined, include *reflections*, which transform a right-handed axis system into a left-handed system. By *rigid transformations,* we mean those isometries that preserve the handedness of axes. The corresponding matrix representations will have determinant $+1$. For reasons discussed in Group Theory and Spatial Reasoning, we refer to the set of rigid transformations as the Proper Euclidean group (of 3-space) and denote it by E^+. Rigid transformations can be expressed as a product of two basic types of rigid transformation, namely *rotations,* in which points of an axis in 3-space remain fixed, and *translations*, which leave directions in 3-space fixed. No points in R^3 remain fixed under translation. If L is a rigid transformation, then $trans(x,y,z)$ denotes a translation by the vector (x,y,z),

High-level robotic assembly planning is concerned with how bodies fit together and how spatial relationships among bodies are established over time.

Rewrite rule for rigid transformation	Condition
$rot(\vec{v},\theta_1)rot(\vec{v},\theta_2) = rot(\vec{v},\theta_1+\theta_2)$	
$rot(\vec{v},0) = 1$	
$rot(\vec{v_1},\theta)rot(\vec{v_2},\pi) = rot(\vec{v_2},\pi)rot(\vec{v_1},-\theta)$	$\vec{v_1} \perp \vec{v_2}$
$trans(0,0,0) = 1$	
$trans(x_1,y_1,z_1)trans(x_2,y_2,z_2) = trans(x_1+x_2,y_1+y_2,z_1+z_2)$	
$trans(x,y,z)rot((x,y,z),\theta) = rot((x,y,z),\theta)trans(x,y,z)$	
$l1 = l$	
$1l = l$	
$(l_1 l_2)^{-1} = l_2^{-1} l_1^{-1}$	
$rot(\vec{v},\theta)^{-1} = rot(\vec{v},-\theta)$	
$trans(x,y,z)^{-1} = trans(-x,-y,-z)$	
Rewrite rule for rigid transformation applied to vector	
$trans(x_1,y_1,z_1)(x_2,y_2,z_2) = (x_1+x_2,y_1+y_2,z_1+z_2)$	
$rot(\mathrm{i},\theta)(x,y,z) = (x, y*cos(\theta) - z*sin(\theta), y*sin(\theta) + z*cos(\theta))$	
$rot(\mathrm{k},\theta)(x,y,z) = (x*cos(\theta) - y*sin(\theta), x*sin(\theta) + y*cos(\theta), z)$	

Table 1. Rewrite Rules.

$trans(\vec{V})$ denotes a translation by the vector $\vec{V}$, and $rot(\vec{V},\theta)$ denotes a rotation by an amount θ about a vector $\vec{V}$.

The representation of rigid transformations as 4*4 matrixes is highly redundant—a rigid transformation can be specified using six numbers (three to define cartesian position, together with three angles such as roll, pitch, and yaw). The six-number representation of rigid transformations has the disadvantage that performing the product operation requires the expensive computation of transcendental functions, for example, sin and cos and their inverses. However, a *via media*, namely, the use of quaternions, allows a rotation about an arbitrary axis to be represented by four numbers, with product and inverse being performed using only algebraic operations. Thus, a rigid transformation can be represented by a pair consisting of a vector, specifying the translation part, and a quaternion, specifying the rotation part. A discussion of the quaternion representation is in Hamilton (1969).

Spatial Relations

If a robot's world is known to consist of a set B of rigid bodies, then a state of this world can be treated as a mapping $S : B \rightarrow E^+$; that is, if $S(B) = g$, then g is the rigid transformation that specifies where the body B is in the state S. A state description corresponds to a constraint on these rigid transformations. Sense data can also be interpreted as state descriptions.

The Rapt language (Popplestone, Ambler, and Bellos 1980) represented an approach to the interpretation of state descriptions couched in terms of spatial relations between features of bodies—descriptions that might be paraphrased in English, for example, "the bottom of the block is against the top of the table." Likewise, action descriptions can usefully refer to body features, for example, "move the bolt along the axis of its shank." The Rapt system takes a task description couched in these terms and infers rigid transformations for each body in each stable state of the robot's world referred to in the task description. Because the bodies referred to include the end effector of the robot itself, the process of interpreting the Rapt program provides the basic data needed to construct a manipulator-level program in a language such as Val.

The implementation described in Popplestone, Ambler, and Bellos (1980) makes use of computer algebra technology (Buchberger and Loos 1982) to make inferences about rigid transformations. Relations between bodies were characterized by algebraic identities; for example, if a plane face on one body B_1 is against a plane face on another body B_2 in a state S, then the relative location of the bodies is given by

$$S(B_1)^{-1}S(B_2) = f_1\,\mathbf{trans}(x,y,0)\,\mathbf{rot}(\mathbf{k},\theta)\,\mathbf{rot}(\mathbf{i},\pi)\,f_2^{-1},$$

where x, y, and θ are free variables corresponding to the three degrees of freedom left unspecified by the plane-against-plane relationship, and f_1 and f_2 are rigid transformations specifying the location of the plane faces in body coordinates. The translation **trans**(*x,y,0*) and the rotation **rot**(*k,θ*) are general forms of symmetries of the plane.[2]

An assembly consists of a complex graph structure of spatial relations. A cycle in such a graph allows us to infer two expressions for the relative location of two bodies: These can be equated, giving rise to equations involving rigid transformations. Table 1 gives some identities that were used in the spatial relation inference system of Rapt. Similar identities were used by Brooks in Acronym (Brooks 1981). Such identities, with a left-to-right order imposed, can be interpreted by a term-rewriting system (Knuth and Bendix 1967).

The most recent formulation of this approach is outlined in Group Theory and Spatial Reasoning and offers a useful generalization to the older Rapt formulation. It is not entirely independent of this formulation because problems that cannot be resolved at the group theoretic level can be expanded in terms of rigid transformations. Our current work at the University of Massachusetts uses an algebraic simplifier running under Poplog (Hardy 1984; Sloman and Hardy 1983) that compiles such rules expressed as Prolog terms into Pop-11 procedures (Barrett, Ramsay, and Sloman 1985) that are indexed by principal functor and autoloaded on demand. Extensive memoisation (Michie 1968; Popplestone 1967) of the simplification function is used to achieve acceptable performance.[3]

Shapes

Any body occupies space, and this occupancy of space provides the most important constraints on assembly. The shape of a body *B* in a state of the world *S* is the subset *shape*(*B,S*) of R^3 that it occupies in this state.[4] Robots will have to manipulate the following kinds of material entity:

First, many bodies can be regarded as rigid from the point of view of assembly planning. Any rigid body *B* has a reference shape, denoted by $shape_0(B)$, that is independent of the world state. In any world state *S*, the body *B* has an associated rigid transformation *S*(*B*), as described in Locations as Rigid Transformations, so that $shape(B,S) = S(B)(shape_0(B))$.

Second, a *rigid subassembly* is a set of bodies whose relative location remains constant in a set of world states in which the subassembly is said to exist. While it exists, a rigid subassembly behaves like a rigid body.

Third, an *articulation* is a set of bodies, certain of whose features bear specified spatial relationships to each other. Like a subassembly, it can come into existence and go out of existence. Its shape can be characterized by the base shapes of the bodies, together with a rigid transformation specifying the location of some designated base body of the articulation, and a set of parameters defining instances of the spatial relations. For example, a robot arm is an articulation, as is a hinge.

Fourth, the shape of an elastic body changes reversibly in response to applied forces, reverting to a shape congruent to its base shape when no forces are applied. Locally, the change in shape is small.

Fifth, the shape of a flexible body (like a shirt or a gasket) changes radically in response to small applied forces and does not revert to being congruent to the base shape when the forces are removed. However, distances measured along the surface remain constant or approximately so.

Sixth, plastic matter reversibly deforms under small applied forces but irreversibly deforms under larger applied forces.

Eighth, liquid and gaseous matter have no intrinsic shape but take their shape from the container they occupy and, in the case of liquid, also in response to applied forces (usually gravity).

To date, almost all the work on robot plan formation that has made a serious attempt to treat shape has concentrated on the first three categories. How are we to represent the shape of bodies in a computationally tractable manner? The first problem is that seldom do we actually know the shape of a body. Body shape can be known either by knowing the process (for example, the manufacturing process) by which it was produced or making use of sensors. Both these methods are subject to error, so that we can only have approximate knowledge of shape. For some purposes, for example, gross motion planning, this problem might not matter because it will be possible to use representations that are known to be upper bounds on body shape. However, fine-motion planning will often require a more complete specification of the range of shapes within which the actual shape of a body is believed to lie. One approach to a more complete specification is treated in Tolerancing Shapes.

Let us now turn to the representation of the nominal base shape of rigid bodies. By *nominal,* we mean the ideal shape that the designer would give the body if he had perfect control of its manufacture. By *base shape,* we

. . . this occupancy of space provides the most important constraints on assembly.

. . . tolerances in manufactured parts are usually specified in terms of tolerances on features.

mean the space it occupies when it is in its reference location. Shape representations for assembly must support a number of capabilities:

First, shapes need to be translated and rotated. This requirement puts many popular shape representations at a disadvantage. For example, the voxel representation, where R^3 is quantized, and bounded subsets are represented by a bit array, is expensive to relocate in space. Likewise the octree representation, which represents shape by a recursive subdivision parallel to the coordinate axes, loses its advantages if rotated (although a dynamic octree, as used in divide-and-conquer algorithms does have a place; see Cameron [1984]).

Second, a shape representation should support the recognition of form features, such as faces, holes, oilways, bearing housings, and keyways. This requirement strongly suggests that a boundary representation of shape in terms of face, edges, and vertexes is necessary, together with the possibility of attaching attributes associated with form features.

Third, a shape representation should support the efficient implementation of a variety of algorithms, including set membership, intersection and null intersection, hidden line and surface removal, and trajectory planning.

Nominal Shapes

When we consider the assembly of manmade bodies that have been designed by an engineer, it is useful to employ the concept of the nominal shape of a body, which is the ideal shape the engineer would wish it to have if she had perfect machines available to make it.

Our primary concern is with representing the shapes of artifacts, which typically have some geometric regularity of shape, especially in the shape of their mating features. Mathematically, these can be characterized as semialgebraic and semianalytic sets.

In the volumetric or constructive solid geometry approach to defining shapes, nominal shapes are defined as subsets of R^3 by taking a collection of primitive shapes; relocating them with rigid transformations (Locations as Rigid Transformations); and combining them with the boolean operations of union, intersection, and set difference.

Implementation of Shapes Using PADL2

A *geometric solid modeler* is a software package providing informationally complete representations of solids such that any well-defined geometric property of any represented solid can be automatically calculated (Requicha and Voelcker 1983). PADL2 (Requicha and Voelcker 1982) is such a solid modeler. We made use of it to provide us with a boundary representation of solids in terms of faces, edges, and vertexes. Solid shapes, specified as Prolog terms in the formalism used in the Edinburgh designer system (Popplestone 1988), are converted to the PADL2 input formalism and processed by this system into boundary models.

The bounded shapes that can be symbolically described for PADL2 have the following form as Prolog terms:

block(X,Y,Z)	A block with dimensions X,Y,Z
cyl(H,R)	A cylinder of height H and radius R
sph(R)	A sphere of radius R
wed(X,Y,Z)	A wedge with dimensions X,Y,Z
con(H,R)	A cone of height H and radius R
tor(Min,Maj)	A torus with minor and major radii Min and Maj

Primitive or composite shapes can be combined through the following boolean operations:

Shape1 \/ Shape2	Union of Shape1 and Shape2
Shape1 /\ Shape2	Intersection of Shape1 and Shape2
Shape1 \ Shape2	Difference of Shape1 and Shape2

Rigid transformations are specified using the following Prolog rendering of the forms specified in Locations as Rigid Transformations:

trans(X,Y,Z)	A pure translation along the x, y, and z axes
trans(vec(X,Y,Z))	A pure translation along the x, y, and z axes
rot(ii,T)	A rotation of T radians about the x axis
rot(jj,T)	A rotation of T radians about the y axis
rot(kk,T)	A rotation of T radians about the z axis
rot(vec(X,Y,Z),T)	A rotation of T radians about a directional vector

If Shape is a shape, and Loc is a rigid transformation, then Shape @ Loc is Shape relocated by Loc; for example:

Solid	Defining Constraint	Description
H	$\{(x,y,z) \| z \leq 0\}$	Infinite half-space
Cyl(r)	$\{(x,y,z) \| x^2+y^2 \leq r^2\}$	Infinite cylinder, radius r
Sph(r)	$\{(x,y,z) \| x^2+y^2+z^2 \leq r^2\}$	Sphere, radius r
Cone(α)	$\{(x,y,z) \| x^2+y^2 \leq z \tan\alpha, z \geq 0\}$	A cone, angle α
Screw(p,.f)	$x^2+y^2 \leq f(\frac{z+p \arg(x+iy)}{2\pi} mod\ p)$	Screw, pitch p and profile f
Gear(n,r,f)	$x^2+y^2 \leq r^2 + f(\arg(x+iy) mod\ \frac{2\pi}{n})$	Gear, n-teeth, radius r profile f

Table 2. Infinite Solids and Their Descriptions.

cyl(4,1) \/ (cyl(4,1)@trans(4,0,0)) \/
(block(1,2,6)@trans(-1,-1,1)@rot
(jj,1.570796)))

denotes an object consisting of two cylinders stuck onto a block with the union operation \/.

A draw predicate is provided that prints the Prolog term as a string in PADL2 syntax and then calls PADL2 to form and display a boundary model of the object. The PADL2 internal representation of the boundary model is extracted using Fortran subroutines linked into Poplog (Hardy 1984; Sloman and Hardy 1983) as external procedures; Pop-11 objects expressing the face-edge-vertex structure of a body are built from the information thus extracted. These objects closely correspond to the structures that PADL2 uses internally.

Table 2 gives the definitions of some infinite solids. The boundaries of these solids correspond with the surfaces used by PADL2 except for the Screw and Gear forms. We treat these in our system by “lying” to PADL2—they exist in our input formalism, they are approximated by cylinders before being input to PADL2, and they are relabeled in the resultant boundary representation.

Tolerancing Shapes

In designing a rigid body, an engineer will also specify the *tolerance* on its shape, which defines a set of subsets of R^3, each of which is a legal shape for the body. One possible approach to tolerancing is to imagine a region of uniform thickness surrounding the surface of the nominal shape, within which the surface of the toleranced body must lie. However,this approach ignores three important issues in tolerancing:

The first is what the functional requirements of shape are. Many bodies do not need to be equally precise all over. Typically, high precision is required for the features at which a body mates with other bodies, and lower precision is required for nonmating features.

The second is how the body is manufactured. High precision is often expensive to obtain. Many bodies start their life as castings or forgings, which can only be produced to a relatively low precision. Precise features are then cut in these by machine tools, which are expensive to operate. Only features that are needed to be precise will be machined.

The third is that some process must allow a body to be inspected to ensure that it is within tolerance. The elementary steps in inspection are feature based.

Thus, tolerances in manufactured parts are usually specified in terms of tolerances on features. A theory of tolerancing intended for computational representation is developed in Requicha and Tilove (1978). They define three kinds of tolerance on features:

First is a *form tolerance,* which specifies the characteristics of some region of space within which the surface of the feature must lie. For example, if a cylindrical hole has form tolerance T_f, then the actual surface of the hole must lie within an annulus defined by two concentric cylinders of radius r_1 and r_2, where $r_2 - r_1 \leq T_f$. Thus, neither the position nor the size of the hole is specified by a form tolerance.

Second is a *size tolerance,* which specifies the dimensions of some region of space within which the surface of the feature must lie. For example, if a cylindrical hole of nominal radius r has size tolerance T_s, then the cylindrical surface of the hole must lie within an annulus defined by two concentric cylinders of radius $r - T_s/2$ and $r + T_s/2$. Thus, the position of the hole is not specified by a size tolerance.

Third is a *position tolerance,* which specifies how a feature is located within the body. The system of position tolerances used in defining a body depends on the inspection technology

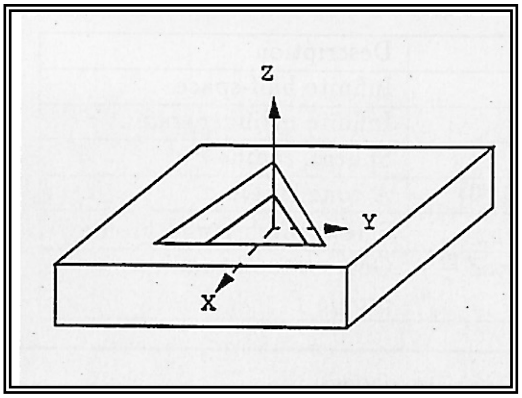


Figure 1. A Body Feature with C_3 *Symmetry.*

used. The modern tendency is to use a numerically controlled measuring machine in which features are located relative to a body-based frame of reference determined by a unique set of datum features.

Tolerance information is the data that characterize those uncertainties that arise from shape variations when assembling bodies designed by an engineer; these uncertainties can also reside in the robot itself and in its environment. Mathematically, tolerances can be ultimately expressed in terms of a system of inequalities. The computational complexity of the deductions required for inferring plans from such systems of inequalities is high (Canny 1987). In practice, much case law will be applied for dealing with standard assembly operations, for example, fitting bearings into bearing housings. The way in which tolerances affect the location of bodies in an assembly is discussed in Fleming (1985), Dakin and Popplestone (1989), and Ellis (1989).

Group Theory and Spatial Reasoning

As we said, bodies in an assembly mate through their features, and a feature can have symmetries. Such symmetries are essential to understanding how features mate and what the final assembly configurations can be. When two bodies are related through a pair of symmetric features, the isometry that specifies their relative location can not be exactly determined. Thus, a hexagonal socket wrench can fit a hexagonal bolt head in six ways because of the symmetry of the related features.

Group theory is normally developed in terms of abstract groups. For a detailed treatment, see Miller (1972). To help you understand the detail of our work, we provide some relevant definitions of group theory and examples of its use in assembly planning:

Definition 1: An *abstract group* G is a set of elements closed under an associative composition operation that we write as multiplication, with an identity element $1 \in G$, and an inverse for any $g \in G$ that obeys the laws:

$g1 = g = 1g, gg^{-1} = 1 = g^{-1}g$.

For example, because the product of isometries is an isometry, as is the inverse of an isometry, the whole set of isometries (Locations as Rigid Transformations) forms a group, called the Euclidean group, denoted by E, with the identity 1 provided by the identity isometry that maps every point to itself.

Consider figure 1, which consists of a block in which a shallow hole in the form of a triangular prism is cut, with the Z axis aligned with the axis of the prism. The symmetry group of the prism is a set of rotations

$C_3 = \{1, rot(\mathbf{k}, 2\pi/3), rot(\mathbf{k}, 4\pi/3)\}$

that we can write as $\{1,\omega,\omega^2\}$. We refer to this set as the cyclic group of order 3 and denote it by C_3. If $g \in C_3$, then g transforms the prism into itself and, thus, is a symmetry of the triangular hole (although not of the block). This group is finite because ω is a 120° rotation about the central axis of the triangle, $\omega^3 = 1$, and $\omega^{-1} = \omega^2$ so that products and inverses of its three members always lie within the set.

Note that group G is said to be finite of order n if it has a finite number n of elements and cyclic if it consists of multiples of a single element.

The group C_3 introduced previously is a subset of the larger group E.

Definition 2: A *subgroup* H of G is a subset of G that is closed under the multiplication and group inverse operations of G.

We restrict ourselves to those isometries that are rigid transformations, thus excluding reflections, which, as we remarked, form the proper Euclidean group, denoted by E^+.

Symmetry Groups

Certain features, which we call *set features*, can be treated as subsets of real 3-space R^3. In this subsection, we formally define the symmetries of a feature and then show that they form a group, called the *symmetry group* of the feature. This fact provides us with the basic justification for the use of group theory in assembly planning.

Definition 3: Let S Ã R^3. Then $g \in E^+$ is a *proper symmetry* of S if and only if $g(S)=S$.

Proposition 1: The proper symmetries of a set S Ã R^3 form a subgroup of E^+.

Let G denote the set of the symmetries of S Ã R^3. Obviously, $1(S) = S$, so $1 \in G$. If $g \in G$, then $g(S) = S$. Multiplying by g^{-1}, we have $g^{-1} g(S) = g^{-1}(S)$; therefore, $g^{-1}(S) = S$, and thus, g^{-1}

$\in G$. Finally, if $g_1, g_2 \in G$, then $(g_1g_2)(S) = g_1(g_2(S)) = g_1(S) = S$; therefore, $g_1g_2 \in G$. By the definition of a subgroup, G is a subgroup of E^+.

The group G associated with a feature S is called the symmetry group of S and denoted by $\Sigma(S)$. For example, the subgroup $C_3 \subset E^+$ is the symmetry group of the prismatic hole.

Extending Group Multiplication to Subsets

If we say that two bodies are spatially related, we must mean that the isometry that specifies their relative location lies in a subset of E^+. This fact leads us to consider the extension of the group composition to operate on subsets of a group. In Spatial Relations from Symmetry Groups, this extension provides a basis for characterizing spatial relations in terms of feature locations and feature symmetry groups.

Definition 4: Suppose S_1 and S_2 are subsets of G. Then, we define

$$S_1S_2 = \{s_1s_2 \mid s_1 \in S_1, s_2 \in S_2\} .$$

By convention, if $g \in G$, we write gS for $\{g\}S$ and Sg for $S\{g\}$. In particular, if $g_1g_2 \in G$, and $H \subset G$ is a subgroup, then g_1H is called a *left coset* of H, Hg_2 is called a *right coset* of H, and g_1Hg_2 is called a *two-sided coset* of H. In Spatial Relations from Symmetry Groups, we show that when two features are so closely related that they can be said to fit, the relative location of the bodies lies in a coset of the (common) symmetry group of the features. This extension of the group multiplication to subsets is associative; has the identity 1; but, in general, has no inverse operation.

Conjugate Subgroups

The particular group C_3 that we previously defined as the symmetry group of the prismatic hole depended on the fact that the axis of the hole happened to be aligned with the Z axis of the world coordinate system. Such a group is a *canonical subgroup* of the Euclidean group, as specified in The Canonical Subgroups of E^+. In general, bodies can have many features, not all of which have their symmetry axes coinciding with the Z axis. Thus, the symmetry group of such features will not be canonical subgroups of E^+. In this subsection, we develop the idea of *conjugate subgroups*, which allow us to make copies of canonical subgroups to be the symmetry groups of arbitrary features. We make these copies by using mappings (called *morphisms*) that preserve group structure.

Definition 5: If G_1 and G_2 are two groups, then a mapping $\theta : G_1 \rightarrow G_2$ is called a *homomorphism* if and only if $\theta(g_1g_2) = \theta(g_1)\theta(g_2)$ for any $g_1, g_2 \in G$. We say that θ projects G_1 into G_2.

Definition 6: If $\theta : G_1 \rightarrow G_2$ is a homomorphism, then the set $\{g \mid g \in G_1, \theta(g)=1\}$ is called the *kernel* of θ.

Definition 7: A 1 – 1 homomorphism from G_1 onto G_2 is called an *isomorphism*. Thus, isomorphic groups have identical group theoretic properties. An isomorphism from a group to itself $\theta : G \rightarrow G$ is called an *automorphism*.

In particular, if $a \in G$, the mapping $g \rightarrow aga^{-1}$ is an automorphism, called an *inner automorphism*. The two subgroups $H_1, H_2 \subset G$ are said to be conjugate if $H_2 = aH_1a^{-1}$ for some $a \in G$.

Note that if g_1Hg_2 is a two-sided coset of $H \subset G$ then

$$g_1Hg_1^{-1}(g_1g_2) = g_1Hg_2 \qquad (1)$$

so that a two-sided coset of a subgroup is a one-sided coset of a conjugate subgroup. Other combinations of group elements and subgroups are referred to as *generalized cosets*. In general, they are not the same as cosets.

Definition 8: If H is a subgroup of G such that for all g in G, $gHg^{-1} = H$, then H is called a normal *subgroup* of G. For example, the set of all translations T^3 is a normal subgroup of E^+.

The following proposition shows that if we move a set feature in space by applying a rigid transformation to it, then we transform its symmetry group into a conjugate group by using the associated inner automorphism.

Proposition 2: If G is the symmetry group of $S \subset R^3$ then for any rigid transformation a in E^+, aGa^{-1} is the symmetry group of $a(S)$.

Let $H = aGa^{-1}$, and let H_a be the symmetry group of $a(S) \subset R^3$. If $h \in H$, then a $g \in G$ exists such that $h = aga^{-1}$, and moreover, $g(S) = S$. Then, $h(a(S)) = aga^{-1}(a(S)) = ag(S) = a(S)$. Thus, h is a symmetry of $a(S)$ and, thus, $h \in H_a$. Thus, we can conclude $H \subseteq H_a$.

Conversely, if $h_a \in H_a$, then $h_a(a(S)) = a(S)$; thus, $a^{-1}h_aa(S) = S$; that is, it is a symmetry of S. Thus, $a^{-1}h_aa = g \in G$, and $h_a = aga^{-1} \in H$; therefore, $H_a \subseteq H$. Thus, we conclude $H = H_a$.

For example, remembering the C_3 symmetry of the triangular prism, suppose we have another prism centered on the X axis. Then, taking $g = rot(\mathbf{j}, \pi/2)$, the group $H = g\{1, \omega, \omega^2\}g^{-1} = \{1, g\omega g^{-1}, g\omega^2g^{-1}\} = \{1, \mathbf{rot}(\mathbf{i}, 2\pi/3), \mathbf{rot}(\mathbf{i}, 4\pi/3)\}$ is a conjugate group of the symmetry group for the earlier triangular prism and is the symmetry group for the original prism rotated to lie with its axis along the X axis.

The Canonical Subgroups of E^+

By proposition 2, when a feature is relocated by a transformation g, the symmetry group of the relocated feature will be the conjugation by g of the symmetry group of the original feature. One approach to representing any feature symmetry group is to make it a conjugate of a canonical symmetry group. These

Two bodies in an assembly are typically related to each other trhough multiple primitive features.

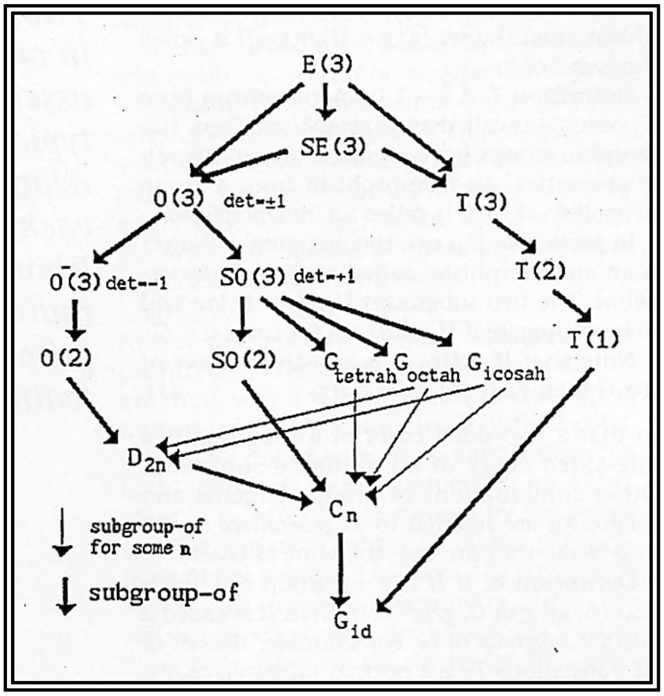


Figure 2. Relationships among Some Important Subgroups of E+.

Subset $S \subset \mathcal{R}^3$	Symmetry group
H	$\mathcal{G}_{plane}$
Cyl(r)	$\mathcal{G}_{cyl}$
Sph(r)	$SO(3)$
Screw(p, r)	$\mathcal{G}_{screw}(p)$
Gear(p_r, p_a, n)	D_{2n}
Cone(θ)	$SO(2)$

Table 3. Correspondence between Shape and Its Symmetry Group.

canonical groups are chosen in a systematic way: If they have a single axis of rotation, it is chosen as the *Z* axis; if they leave a single point of 3-space fixed, it is chosen as the origin; and so on.

A symmetry group of *S* can be represented by a pair consisting of a canonical symmetry group G_{canon} and an element *g* of E^+ that transforms *S* from the origin to its current location. Table 3 gives some of the correspondences between subsets of R^3 and their canonical symmetry groups.

A list of some important canonical subgroups of E^+ with their definitions is given in table 4. Figure 2 shows subgroup relationships between some important subgroups of E^+. The arrow $G_1 \rightarrow G_2$ in figure 2 means that G_2 is a subgroup of G_1.

To apply the theory to actual robotic reasoning, we make use of the boundary models provided within the Poplog system by the linked-in PADL2 modeler, as described in Shapes. Each face *F* of a model is labeled with its symmetry group, each group being considered as the image $f^{-1}G_{canon}f$ of a canonical subgroup of E^+ under an inner automorphism. A data structure denoting the canonical subgroup G_{canon} is obtained by table lookup from the surface type of the face, using a Pop-11 property procedure *gr_canon*. For example, if *F* is a canonical face, *gr_canon(F)* is a data structure denoting the group *SO*(2) of all rotations about the *Z* axis. The *f* for the inner automorphism is the rigid transformation defining the location of the face in body coordinates, as given by PADL2. The conjugation is performed by a procedure whose definition depends on which representation is used for G_{canon} (see Computing Group Intersections).

It is possible to use the feature location because the coordinate systems are embedded in features by PADL2 in such a way that permits a coherent and consistent choice of canonical groups, largely because the *Z* axis is chosen by PADL2 to be the axis of symmetry.

Spatial Relations from Symmetry Groups

In this subsection, we consider what we can infer about the relative location of two bodies that have two features in contact. We noted that such a relation must correspond to a set of isometries that specify the relative location of the two bodies.

Let B_1 and B_2 be two bodies, with primitive features F_1 and F_2 that have symmetry groups $\Sigma(F_1)$,$\Sigma(F_2)$ and that are located in their respective body coordinate systems by isometries f_1 and f_2. Suppose the two features are in contact. If they are in contact over a finite area, we say F_1 fits F_2. If the contact is a line or point contact, then we say F_1 against F_2. In either case, it is clear that if we move B_1 or B_2 by a member of the symmetry groups $\Sigma(F_1)$ or $\Sigma(F_2)$, respectively, the relationship between the features is preserved. We can generally express this relationship by a constraint between the isometries l_1 and l_2, specifying the locations of bodies B_1 and B_2 in the world coordinate system. Therefore, the location of B_1 relative to B_2 is $l_1^{-1}l_2$ and obeys

$$l_1^{-1}l_2 \in f_1\Sigma(F_1)\iota(\mathbf{v},\rho,F_1,F_2)\Sigma(F_2)f_2^{-1} \qquad (2)$$

Canonical Groups	Definition
$\mathcal{G}_{id}$	$\{1\}$
$\mathcal{T}^1$	$\mathbf{gp}\{\mathbf{trans}(0,0,z)\|z \in R\}$
$\mathcal{T}^2$	$\mathbf{gp}\{\mathbf{trans}(x,y,0)\|x,y \in R\}$
$\mathcal{T}^3$	$\mathbf{gp}\{\mathbf{trans}(x,y,z)\|x,y,z \in R\}$
$SO(3)$	$\mathbf{gp}\{\mathbf{rot}(\mathbf{i},\theta)\mathbf{rot}(\mathbf{j},\sigma)\mathbf{rot}(\mathbf{k},\phi)\|\theta,\sigma,\phi \in R\}$
$SO(2)$	$\mathbf{gp}\{\mathbf{rot}(\mathbf{k},\theta)\|\theta \in R\}$
$O(2)$	$\mathbf{gp}\{\mathbf{rot}(\mathbf{k},\theta)\mathbf{rot}(\mathbf{i},n\pi)\|\theta \in R, n \in \mathcal{N}\}$
$\mathcal{G}_{cyl}$	$\mathbf{gp}\{\mathbf{trans}(0,0,z)\mathbf{rot}(\mathbf{k},\theta)\mathbf{rot}(\mathbf{i},n\pi)\|n \in \mathcal{N},\theta,z \in R\}$
$\mathcal{G}_{dir\_cyl}$	$\mathbf{gp}\{\mathbf{trans}(0,0,z)\mathbf{rot}(\mathbf{k},\theta)\|z,\theta \in R\}$
$\mathcal{G}_{plane}$	$\mathbf{gp}\{\mathbf{trans}(x,y,0)\mathbf{rot}(\mathbf{k},\theta)\|x,y,\theta \in R\}$
$\mathcal{G}_{screw}(p)$	$\mathbf{gp}\{\mathbf{trans}(0,0,z)\mathbf{rot}(\mathbf{k},2z\pi/p)\|z \in R\}$
$\mathcal{G}_{T_1C_2}$	$\mathbf{gp}\{\mathbf{trans}(0,0,z)\mathbf{rot}(\mathbf{i},n\pi)\|n \in \mathcal{N}, z \in R\}$
D_{2n}	$\mathbf{gp}\{\mathbf{rot}(\mathbf{k},2\pi/n)\mathbf{rot}(\mathbf{i},m\pi)\|m,n \in \mathcal{N}\}$
C_n	$\mathbf{gp}\{\mathbf{rot}(\mathbf{k},2\pi/n)\|n \in \mathcal{N}\}$

Table 4. Some Important Subgroups of E^+.

Here, ρ is a token indicating the kind of relation that pertains, and f_1 and f_2 are the locations F_1 and F_2 in their respective body coordinates. The vector $\boldsymbol{v}$ provides variables that complement the variables implicit in the symmetry groups. For example, in the case of the relationship between a cam and its follower, one parameter is needed to specify the angle of the cam. In many important cases, there are no such complementary variables, for example, that of a cylinder against a plane surface.

Table 5 summarizes ι for the cases treated by the Rapt language (Ambler and Popplestone 1975; Popplestone, Ambler, and Bellos 1980) except for the fits relation, which is treated later.

The fits relation is particularly constraining. If the primitive features are those algebraic sets that are used in PADL2, then areal contact implies that the surfaces are identical so that the symmetry groups are identical. For two algebraic sets to fit, one must be the complement of the other, and we have

$$l_1^{-1}l_2 \in f_1\Sigma(F_1)f_2^{-1} \quad (3)$$

We can summarize equation 3 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 coset of the common symmetry group of the features.

Finally, let us note that we characterized a spatial relation between bodies B_1 and B_2 in equations 2 and 3 in terms of a generalized coset S_{12}:

$$l_1^{-1}l_2 \in S_{12} \quad (4)$$

Group Intersections

Two bodies in an assembly are typically related to each other through multiple primitive features. If bodies B_1 and B_2 are related by fitting two pairs of features, such as a peg in a blind hole (figure 3)—that is, f_{11} fits f_{21}, and f_{12} fits f_{22}, where f_{11} and f_{12} are features of B_1 and f_{21} and f_{22} are features of B_2—we can use equation 3 to obtain the relative location $l_1^{-1}l_2$ *of* B_1 *to* B_2 as

$$l_1^{-1}l_2 \in f_{11}\Sigma(F_1)f_{21}^{-1} \cap f_{21}\Sigma(F_2)f_{22}^{-1} \quad (5)$$

that is, the intersection of two two-sided cosets. Because equation 1 shows that each two-sided coset can be rewritten as a one-sided coset, we can compute equation 5 as the intersection of two one-sided cosets. We have the following proposition:

Proposition 3: If H_1 and H_2 are subgroups of G, and $g_1, g_2 \in G$, then the intersection of the two right cosets H_1g_1 and H_2g_2 is either a right coset or is null (Popplestone 1984).

If H_1 and H_2 are subgroups of G, and $g_1, g_2 \in G$, then $H_1g_1 \cap H_2g_2 = (H_1 \cap H_2g_2g_1^{-1})g_1$ by

F_1	F_2	relation ρ	interface element ι
H	H	fits	$rot(i, \pi)$
Cyl(r)	H	against	$trans(0,0,r)rot(i, -\pi/2)$
Sph(r)	H	against	$trans(0,0,r)$
Edge	H	against	$rot(i, -\pi/2)$
Vertex	H	against	I
Cyl(r)	Cyl(r)	fits	$rot(i, \pi)$

Table 5. Interface Element ι.

proposition 2 in Popplestone 1984:

$$H_1g_1 \cap H_2g_2 = ((H_1 \cap H_2)h_1)g_1 = (H_1 \cap H_2)h_1g_1 \quad (6)$$

where $h_1 \in H_1$.

Equation 6 implies that the intersection of two cosets can be obtained by intersecting the corresponding subgroups, finding h_1 (we can use the fact that $h_1 \in H_1 \cap H_2g_2g_1^{-1}$, and forming the final coset $(H_1 \cap H_2)h_1g_1$.

In effect, this shows that such multiple-fitting relationships can be regarded as a single relationship between a pair of compound features. A compound feature F_{comp} of body B is a set of primitive features F_i of B. Provided that the features F_i are all distinct, the symmetry group of F_{comp} is the intersection of the symmetry groups of those primitive features of which it is composed:

$$\Sigma(F_{comp}) = \cap_i \Sigma(F_i) \ .$$

In our discussion, we have assumed that primitive features (set features) are distinguished when a mating relationship is formed, as if each feature has a distinct color. If they are not, then a compound feature can have additional finite symmetries; for example, the head of a bolt formed by six planes has the symmetry group C_6. Some permutations of the primitive features of a compound feature can generate a symmetry group. We can say, in effect, that the feature fits a transform of its complement. By repeatedly applying the rule described in the last subsection, such a permutation will give either a coset of E^+ or the empty set. The union of these cosets, taken over all permutations, will generate the feature symmetry group. In the case of polyhedra, algorithms for finding whole-body symmetries are described in Waltzman (1987) and Wolter, Woo, and Volz (1985).

Cycles and Chains of Spatial Relationships. Let us begin by observing that the spatial relationships we have discussed allow us to define relations between the locations of pairs of bodies. Graphs of such binary relations are studied by Montanari and Rossi (1988). In particular, they discuss algorithms for reducing such graphs.

Because the relations under consideration for assembly are, in general, infinite, we have to compute using descriptions of the relations rather than sets of pairs. In the Rapt language, this computation was implemented in two ways: (1) using algebraic descriptions based on an algebra of locations and the reals, as described in Popplestone, Ambler, and Bellos (1980), and (2) using labels for different kinds of relations in a constraint network and simplifying the network using an extensive set of reduction rules (Ambler and Popplestone 1975). This latter implementation, in effect, used the reduction techniques described in Montanari and Rossi (1988).

In this subsection, we consider a third approach, namely, one in which generalized cosets are used. Group theory is not a magic bullet in this work—the apparatus of spatial relations is sufficiently powerful to describe any mechanism made of prismatic and revolute joints, and inherent in such problems are algebraic equations of high degree. Group theory can assist, however, in treating the simple cases that are important in assembly. It provides a generalization to finite and discrete symmetries, and where the solution of algebraic equations cannot be avoided, group theory can help us come up with more tractable forms of the equations.

In Group Intersections, we saw how to treat the simplest kind of relation cycle of length 2, in which two bodies are related by the fitting of two pairs of compound features. However, we also need to deal with nonfitting relationships and cycles of length > 2. Cases of these relationships and cycles, important for assembly, can be treated by using a kind of transitivity that holds among spatial relationships when certain alignments exist. For example, if a block B_3 is placed on a block B_2, which itself is placed on a block B_1, then B_3 can be regarded as being placed on an imaginary surface of B_1 placed at a height equal to the thickness of B_2 above the actual top surface of B_1.

We can relate this consideration to the idea of generalized cosets as follows: Suppose we have two bodies B_1 and B_2, each of which has features F_{11} and F_{21}, and these features are related. Suppose also that B_3 is related to B_2 because a feature F_{22} of B_2 is related to feature F_{31} of B_3. Let l_1, l_2, and l_3 be the positions B_1, B_2, and B_3, respectively. Then, from equation 4,

$$l_1^{-1}l_2 \in S_{12} \quad (7)$$

and

$$l_2^{-1}l_3 \in S_{23} \quad (8)$$

where S_{12} and S_{23} are generalized cosets, as specified in Spatial Relations from Symmetry Groups. Hence, by the definition of set multiplication,

$$l_1^{-1} l_3 \in S_{12}S_{23} \quad (9)$$

that is, a generalized coset $S_{12}S_{23} = S_{13}$, say, can be used to characterize the relationship between B_3 and B_1. Now, it is common to find alignments of body features that occur in assemblies, for example, the top and bottom faces of a block or a washer or the inner and outer cylindrical faces of a bush. These alignments give rise to possible simplifications. The strategy for achieving simplifications is, typically, to use commutation conditions between groups and elements to bring together groups whose product is known. Suppose the term $S = G_1 g G_2$ occurs in S_{13}. Then, it might happen that g commutes with G_1 so that we can rewrite our term as gG_1G_2. It might also happen that G_1G_2 is known to be a group—for example, when G_1 0 G_2, so that $G_1G_2 = G_2$—or that G_1 might be a translation group and G_2 a rotation group, with the right alignment to make their product a *TR* group (Computing Group Intersections). In this case, our original term can be rewritten in the form gG, where $G = G_1G_2$. This process allows us to provide an exact equivalent for our original subterm *S*. It is also possible to provide a weaker form that might still prove useful; namely, we can use the fact that G_1G_2 Ã G_1 » G_2, the group generated by the product.

Let us consider the example shown in figure 4. Here, B_1 is a block with a pillar on top and a triangular hole, B_2 is a cylinder with a triangular prism on top, and B_3 is a block with a bigger cylindrical blind hole and a smaller cylindrical through hole. Then, in the configuration, if they are about to be assembled as shown in figure 4, the relative positions of B_1 to B_2 and B_2 to B_3 are

$$l_1^{-1} l_2 \in trans(0,0,3)C_3 trans(0,0,-1)$$

and

$$l_2^{-1} l_3 \in trans(0,0,4)SO(2)trans(0,0,-2)$$

so, the relative position of B_1 to B_3 can be obtained by the product of the left hand from the previous two expressions:

$$l_1^{-1} l_3 \in trans(0,0,3)C_3 trans(0,0,-1)trans(0,0,4) SO(2)trans(0,0,-2) = trans(0,0,3)C_3 SO(2)trans(0,0,1)$$

because translations along the Z axis commute with $SO(2)$, and C_3 Ã $SO(2)$:

$$l_1^{-1} l_3 = SO(2)\, trans(0,0,4).$$

We can also use the fact that B_3 fits the pillar on B_1, giving

$$l_1^{-1} l_3 \in trans(0,3,0)G_{cyl} trans(0,-3,0)$$

Intersecting these cosets, we obtain

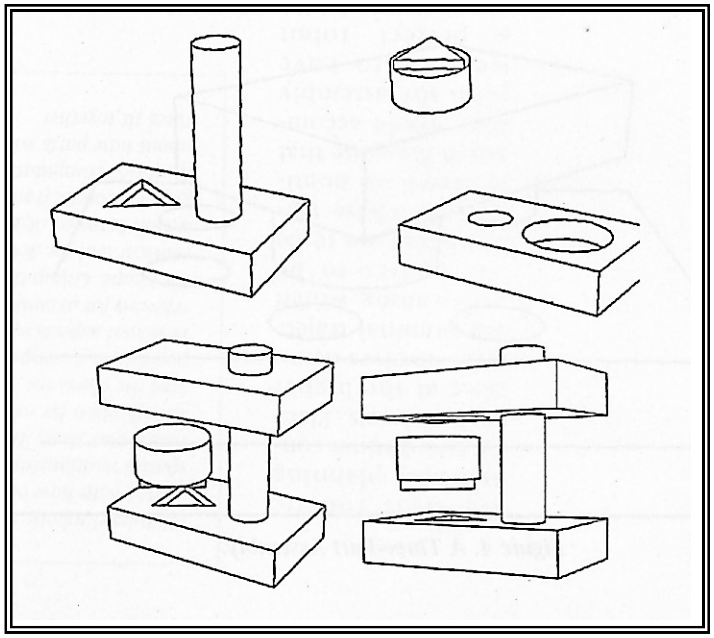


Figure 3. Two Bodies Are Related by Multiple Mating Features.

$$l_1^{-1} l_3 \in trans(0,0,4)\{1\}$$

that is, we know the relative location of these two bodies.

Group theory is a high level of abstraction for spatial reasoning and cannot resolve all our problems. We can translate from generalized cosets into location expressions by repeated application of the rules of definition 4 and the group membership definitions given in table 4, thus bringing us into the technology of Rapt, discussed in Locations as Rigid Transformations. An alternative approach could be to look for standard kinematic mechanisms represented in the group theoretic form.

Planning What Body Features to Relate. In specifying assemblies, it is often desirable that the actual mating features of bodies are not specified because of the tediousness caused by the symmetries of assembly components, for example, instructing a robot to fit a spline into a splined hole instead of specifying all the possible mating surfaces. When the only fact given is that bodies themselves are to be mated, an assembly planner is expected to find the possible mating feature pairs and assembly configurations from the geometric models of assembly components. Therefore, we have studied how to infer sets of possible mating features of two bodies. For a pair of bodies, each having *m* and *n* primitive features, respectively, with *m*

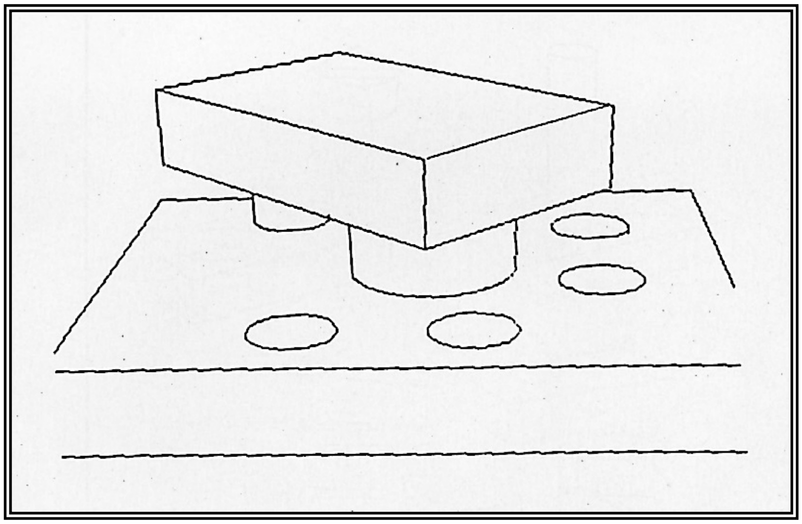


Figure 4. A Three-Part Assembly.

$\leq n$, the number of possible mating feature pairs between the two bodies would be $\sum_{i=1}^{n} C_n^i C_m^i$, of which only a few will be found to be compatible. An alternative approach to mitigating the combinatorics is the identification of compound features of a body that are instances of a salient feature library (Liu and Popplestone 1989). Some of the library features are quite specific, such as countersink, counterbore, keyway, and certain cases of spline. More generic assembly-relevant features are inserters, containers, multiinserters, and multicontainers, which are, in effect, general protrusions, concavities, and combinations of these. Feature definitions refer to the faces of the features of a single body and relationships between them, such as being adjacent, perpendicular, or parallel.

In the case of the fit assembly operation, the mating features have the same symmetry group at the area of contact. Therefore, one important necessary condition for candidate mating features is that they must have the same symmetry group. If this condition is checked first, it saves the planner from examining the detailed dimensions of every pair of compound features when they appear to be noncompatible from a glance at their symmetries.

The importance of using the symmetry group as the main descriptor for features is that necessary conditions for spatial relationships to hold between body features can be expressed in terms of the symmetry groups of the features. Necessary and sufficient conditions will, of course, depend on additional descriptors—a gear and a spline might have the same symmetry group, but the gear will not fit the spline. However, the main geometric aspects of the spatial relationship can be encompassed in the group theory, leaving the sufficiency to be checked by applying rules for assessing the consistency of scalar and discrete parameters.

Dimensional consistency of candidate mating features is also required. There are two kinds of dimensions involved: First, the parameters of each PADL2 surface that is a component of one compound feature should be consistent with the parameters of the corresponding surface component of the other compound feature. Second, sets of characteristic invariants (Computing Group Intersections) used in calculating the intersection groups have intrinsic dimensions (for example, the length of the common perpendicular between line invariants and the angle between them); these dimensions should be consistent between corresponding compound features. A detailed description of this work can be found in Liu and Popplestone (1989).

Computing Group Intersections

As discussed earlier, the symmetry group of a compound feature is the intersection of the symmetry groups of its components. Two methods were developed and implemented for computing intersections: characteristic invariants and tractable groups.

Characteristic invariants are geometric entities associated with a group that have the property that they are invariant under the group actions and the property that they characterize the group. The fact that E^+ is the semidirect product of T^3 and $SO(3)$—that is, $E^+ = T^3\ SO(3) = \{tr \mid t \in T^3, r \in SO(3)\}$—led us to examine a family of subgroups of E^+ called *TR* groups. These are the groups $G = TR \subseteq E^+$, where T is a subgroup of T^3, and R is a conjugation of a subgroup of $SO(3)$. Because T is the kernel of a homomorphism from G onto R, T is normal in G. Therefore, G is a semidirect product (MacLane and Birkhoff 1979) of T and R, and the quotient group G/T is isomorphic to R.

There are two types of invariant for a *TR* subgroup of E^+, namely, translational invariants T_G and rotational invariants R_G. They characterize the maximal translational subgroup of G and the maximal rotational subgroup of G, respectively. The translational invariant is the T-orbit of the origin s_0; that is, $\{t(s_0) \mid \text{for all } t \in T\}$. The rotational invariant is a pair composed of a fixed-point set F together with a set of poles, $F = T(\{x \mid x \in R^3, r(x) = x, r \in R\})$. A pole of a rotation group is obtained by conjugating the group by a translation so that the conjugation is centered at the origin. Each pole is then an invariant point on the unit sphere, together with an integer indicating the order of the stabilizer, that is, the number of different rotations that leave the point fixed, or *0* if it is *SO*(*2*). For example, the translational invariant of the canonical plane group $G_{plane} = T^2SO(2)$ happens to be the subvector space coincident with the *X*-*Y* plane. The fixed-point set *F* is all of 3-space, and the poles are {((*0*,*0*,*1*),*0*), ((*0*,*0*,-*1*),*0*)}. We proved that a one-to-one correspondence exists between *TR* groups and the set of characteristic invariants (Liu and Popplestone 1990). The method of intersecting two groups G_1 and G_2 maps each group to its invariants, $G_1 \rightarrow (T_{G1}, R_{G1})$, $G_2 \rightarrow (T_{G2}, R_{G2})$. Then,some simple geometric computations are performed on the invariants to get a new pair of invariants $(T_{G1 \cap G2}, R_{G1 \cap G2})$. Finally, this pair is uniquely mapped back to the intersected group $G_1 \cap G_2$. In essence, this pair of characteristic invariants sufficiently represents the intersected group itself.

The representation by characteristic invariants of *TR* groups *G* = *TR*, where *T* and *R* can be finite or infinite, discrete or continuous, has an efficient implementation algorithm and has been applied to compute symmetry groups of the boundary models from the solid modeler PADL2. If the translational group *T* is restricted to being a vector space, then the group *TR* is called *tractable* (Zahnd, Nair, and Popplestone 1989). The method of tractable groups also simplifies the computation of group intersections by separately analyzing translations and rotations. In contrast to the characteristic invariant approach, in the tractable group approach, translational groups, required to be subvector spaces, are represented by a basis for this vector space. *TR* groups form a super set of tractable groups.

As an example of how one computes intersections of tractable groups, consider a cylinder on a plane, with the axis of the cylinder parallel to the normal of the plane. The group of the plane and the group of the cylinder can each be written as a product of a translation group with a rotation group. Because the translation parts are just vector subspaces of R_3, they can easily be intersected; in this case, the intersection has dimension 0. The rotation group in each case is a conjugate of *SO*(2), although in the case of the plane the choice is not unique. As for characteristic invariants, the fixed-point set for the plane can be made to coincide with that of the cylinder. The intersection of the two symmetry groups is just *SO*(2).

Conclusion

There are several approaches to robotic assembly planning. One important aspect that has received limited attention is the use of the symmetries existing in assembly components. Our work has shown the potential of exploiting such symmetries of components in planning their assembly. The following results of our work are relevant to the issues raised in the beginning of this article: First, the symmetry group of a compound feature can be obtained by the intersection of the symmetry groups of each primitive feature of which the compound feature is composed, provided each of these primitive features is distinct. Second, when two features fit, they have the same symmetry group; therefore, having the same symmetry group is a necessary condition for features to mate. Third, because two mating features have the same symmetry group—the relative positions of the two bodies, to which the features belong is a coset of this group—and, in particular, if the symmetry group is the identity group, the relative position of the two bodies is uniquely determined.

An implementation of the approach described in this article is under development. Aspects of the work based on group intersection have been implemented, as has the interface to PADL2. Complete nominal assembly plans, making use of group intersection only, have been created (Liu 1990). The term-rewriting system referred to has also been implemented but has not been integrated with the planner, and work on simplifying group products is in the preliminary stages.

Beyond our own work, it is possible to relate the potential use of symmetries to other existing work in high-level assembly planning. We would like to offer the following two examples:

First, Homem de Mello and Sanderson (1986) present a representation for assembly plans based on AND/OR graphs, or *hypergraphs*. Each node in such a graph corresponds to an assembly. Those nodes containing only one part are the leaves of the graph. A set of directed arcs, which are related by AND, represents a disassembly operation. Each arc points from the original assembly to one of the subassemblies. If symmetries are present in the assembly, then the AND/OR graph to describe the possible disassemblies can be bushy. A treatment of symmetries could provide a more compact and efficient representation for assembly plans.

Second, De Fazio and Whitney (1987) extended Bourjault's work on generating all the assembly sequences from a liaison diagram. Although the algorithm for generating all possible assembly sequences was successfully implemented (Whitney et al. 1989), it is still unclear how the liaison diagram can be automatically generated and how difficult it is to answer those questions asked prior to the generation of assembly sequences. Liu and Popplestone's (1989) work on finding mating features from boundary models of assembly components could be extended to establish liaison diagrams and answer the questions based on the geometric, spatial, and kinematic constraints and, thus, be complementary to De Fazio and Whitney's work.

We are currently developing the connection between planning with nominal shapes, described in previous sections, with an analysis of uncertainty and the exploitation of compliance to reduce uncertainty (Popplestone et al. 1989).

Acknowledgments

The preparation of this article was supported by National Science Foundation grants IRI-8709949 and CCR-8500332, Office of Naval Research grants N00014-84-K-0564 and N00014-86-K-0764, Defense Advanced Research Projects Agency grant F30602-87-C-0140, and the National Swiss Research Foundation. We would like to express our thanks for the helpful and constructive suggestions of the reviewers of this article.

References

Ambler, A. P., and Popplestone, R. J. 1975. Inferring the Positions of Bodies from Specified Spatial Relationships. *Artificial Intelligence* 6:157–174.

Barrett, R. ; Ramsay, A.; and Sloman, A. 1985. *POP-11: A Practical Language for Artificial Intelligence*. New York: Wiley.

Bourtjault, A. 1984. Contribution a Une Approche Méthodologique de L'Assemblage Automatisé: Elaboration Automatique des Séquences Opératoires ("A Contribution to a Methodological Approach to Automatic Assembly: Automatic Generation of Sequences of Operations"). Thése d'Etat, Université de Franche-Comté.

Brooks, R. A. 1981. Symbolic Reasoning among 3-D Models and 2-D Images. *Artificial Intelligence* 17:285–348.

Buchberger, B., and Loos, R. 1982. Algebraic Simplification. In *Computer Algebra—Symbolic and Algebraic Computation*, eds. B. Buchberger, G. E. Collins, and R. Loos., 11–43. New York: Springer-Verlag.

Cameron, S. 1984. Modelling Solids in Motion. Ph.D. diss., University of Edinburgh.

Canny, J. 1987. The Complexity of Robot Motion Planning. Ph.D. diss., Massachusetts Institute of Technology.

Dakin, G., and Popplestone, R. J. 1989. Calculation of Object Pose Constraints from Sparse, Erroneous Sensory Data, COINS Technical Report, 89-64, Department of Computer and Information Science, University of Massachusetts.

Dakin, G.; Liu, Y.; Nair, S.; Popplestone, R.; and Weiss, R. 1989. Symmetry Inference in Planning Assembly. In Proceedings of the 1989 IEEE International Conference on Robotics and Automation, 1865–1868. Washington, D.C.: IEEE Computer Society Press.

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

Ellis, R. E. 1989. Uncertainty Estimates for Polyhedral Object Recognition. In Proceedings of the IEEE International Conference on Robotics and Automation, 348–353. Washington, D.C.: IEEE Computer Society Press.

Fleming A. 1985. Analysis of Uncertainties in a Structure of Parts. In Proceedings of the Ninth International Joint Conference on Artificial Intelligence, 1113–1115. Menlo Park, Calif.: International Joint Conferences on Artificial Intelligence.

Fu, K. S.; Gonzalez, R. C.; and Lee, C. S. G. 1987. *ROBOTICS: Control, Sensing, Vision, and Intelligence.* New York: McGraw-Hill .

Hamilton, W. R. 1969. *Elements of Quaternions*. New York: Chelsea.

Hardy S. 1984. A New Software Environment for List-Processing and Logic Programming. In *Artificial Intelligence, Tools, Techniques, and Applications*, eds. T. O'Shea and M. Eisenstadt, 41–63. New York: Harper and Row.

Hervé, J. M. 1978. Analyse Structurelle des Mécanismes par Groupe des Déplacements ("Analyzing the Structure of Mechanisms Using Groups of Transformations"). *Mechanism and Machine Theory* 13(4): 437–450.

Homem-de-Mello, L. S., and Sanderson, A. C. 1986. AND/OR Graph Representation of Assembly Plans. In the Proceedings of the Fifth National Conference on Artificial Intelligence, 1113–1119. Menlo Park, Calif.: American Association for Artificial Intelligence.

Knuth, D. E., and Bendix, P. B. Simple Word Problems in Universal Algebras. *OXFORD* 67: 263–298.

Liu, Y. 1990. High-Level Assembly Planning from Solid Models Using Symmetry Groups. Ph.D. diss., Department of Computer and Information Science, University of Massachusetts. Forthcoming.

Liu, Y., and Popplestone, R. J. 1990. A Justification for the Characteristic Invariant Representation of TR Subgroups of the Proper Euclidean Group. Submitted to the 1990 IEEE International Conference on Robotics and Automation.

Liu, Y., and Popplestone, R. J. 1989. Planning for Assembly from Solid Models. In Proceedings of the 1989 IEEE International Conference on Robotics and Automation, 222–227. Washington, D.C.: IEEE Computer Society Press.

MacLane, S., and Birkhoff, G. 1979. *Algebra*. New York: MacMilla n.

Michie, D. Memo Functions and Machine Learning. *Nature* 218:19–22.

Miller,W., Jr. 1972. Symmetry Groups and Their

Applications. New York: Academic.

Montanari, U., and Rossi, F. 1988. Fundamental Properties of Networks of Constraints: A New Formulation. In *Search in Artificial Intelligence,* eds. L. Kanal and V. Kumar, 426–449. New York: Springer-Verlag.

Popplestone, R. J. 1988. The Edinburgh Designer System as a Framework for Robotics: The Design of Behavior. *Artificial Intelligence for Engineering Design Analysis and Manufacturing*1(1): 25–35.

Popplestone, R. J. 1984. Group Theory and Robotics. In R*obotics Research: The First International Symposium*, eds. M. Brady and R. Paul, 55–64. Cambridge, Mass.: MIT Press.

Popplestone, R. J. 1967. Memo Functions and the POP-2 Language. Research Memorandum, MIP-R-30, Department of Machine Intelligence and Perception, University of Edinburgh.

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

Popplestone, R. J.; Grupen, R. A.; Liu, Y.; Dakin, G.; Oskard, D.; and Nair, S. 1989. Planning for Assembly with Robot Hands. In Proceedings of the Society of Photo-Optical Instrumentation Engineers Conference on Intelligent Robotic Systems. Philadelphia, Penn.

Popplestone R. J.; Weiss R.; and Liu,Y. 1988. Using Characteristic Invariants to Infer New Spatial Relationships from Old. In Proceedings of the 1988 IEEE International Conference on Robotics and Automation, 1107–1112. Washington, D.C.: IEEE Computer Society Press.

Requicha A. A. G., and Tilove R. B. 1978. Mathematical Foundations of Constructive Solid Geometry: General Topology of Closed Regular Sets, TM27A, Production Automation Project, University of Rochester.

Requicha, A. A. G., and Voelcker, H. B. 1983. Solid Modelling: Current Status and Research Directions. *IEEE Computer Graphics Applications* 3(7): 25–37.

Requicha, A. A. G., and Voelcker, H. B. 1982. Solid Modeling: A Historical Summary and Contemporary Assessment. *IEEE Computer Graphics Applications* 2(2): 9–24.

Sloman, A., and Hardy, S. 1983. POPLOG: A Multi-Purpose Multi-Language Program Development Environment. *AISB Quarterly* 47.

Thomas, F., and Torras, C. 1988. A Group-Theoretic Approach to the Computation of Symbolic Part Relations. *IEEE Journal of Robotics and Automation* 4(6): 622–634.

Waltzman, R. 1987. Finding Symmetries of Polyhedra, CS-TR-1937, Center for Automation Research, University of Maryland.

Whitney, D. E.; DeFazio, T. L.; Gustavson, R. E.; Graves, S. C.; Abell, T.; Cooprider, C.; and Pappu, S. 1989. Tools for Strategic Product Design, Preprints of National Science Foundation Engineering Design Research Conference, College of Engineering, University of Massachusetts, 11–14 June.

Wolter, J. D.; Woo, T. C.; and Volz, R. A. 1985. Optimal Algorithms for Symmetry Detection in Two and Three Dimensions. In *The Visual Computer,* 1–12. New York: Springer-Verlag.

Zahnd, A.; Nair, S.; and Popplestone, R. J. 1989. Tractable Subgroups of the Euclidean Group, COINS Technical Report, 89-51, Department of Computer and Information Science, University of Massachusetts.

Robin J. Popplestone is a professor of computer and information science at the University of Massachusetts at Amherst. In the 1960s, with R. M. Burstall at Edinburgh University, he developed the Pop-2 AI language, combining a human- oriented syntax with extended functional capabilities. In the 1970s, he worked on robotics and vision, including the demonstration of robotic assembly using two-dimensional vision and touch, the use of structured light for three-dimensional sensing, and the specification of assembly in terms of spatial relations. In the early 1980s, he was the architect of the Edinburgh designer system (EDS), which applies AI techniques to the support of engineering design at multiple conceptual levels. He is currently concerned with developing a general and complete mathematical basis for robotics.

Yanxi Liu is a doctoral candidate in the Computer and Information Science Department at the University of Massachusetts at Amherst. She received her B.S. in radio and electronics from Beijing Normal University in 1982 and her M.S. in computer science from the University of Massachusetts at Amherst in 1986. Her Ph.D. dissertation addresses the theory of symmetry groups and its application to robotics assembly planning.

Richard S. Weiss received the B.A. degree from Brandeis University in 1969. He received a Ph.D. in mathematics from Harvard University in 1976. He has been a research associate in the Department of Computer and Information Science at the University of Massachusetts at Amherst since 1983.

Notes

1. Symmetry in ordinary parlance includes mirror symmetries, which cannot in general be realized by any physical movement and, thus, have little relevance to reasoning in a three-dimensional assembly.

2. We modified the original Rapt conventions about embedding axes in features, and so on, to be consistent with engineering practice.

3. To memoise a function, some kind of associative memory is attached to it, so that repeated evaluations are avoided. The concept, owed to D. Michie, is the software equivalent of caching.

4. This concept of shape is only appropriate for a macroscopic world and, of course, breaks down at the atomic level.