Towards Feature Attachment[*]

Xiangping Chen Christoph M. Hoffmann
Department of Computer Science, Purdue University
West Lafayette, IN 47907-1398

First communicated as report CSD-TR-94-010[†]

Abstract

Feature attachment in generative, constraint-based CAD systems requires an unambiguous semantics that is easy to grasp by users and reasonable to implement in CAD systems. We propose a procedural semantics for attaching protrusions and cuts that addresses the problems found in generative, constraint based CAD systems. Our solution also pays attention to legacy considerations inherent in a number of CAD architectures that have evolved from the paradigm of creating geometry using regularized Boolean operations.

1 Introduction

Feature-based design is emerging as the basic design paradigm of CAD systems. In feature-based systems, the user designs with a vocabulary of design elements that are grouped into generated features, such as protrusions and cuts, modifying features such as chamfers and blends, and auxiliary features such as datum axes and planes. Commercial systems such as Pro/Engineer from Parametric Technologies provide evidence that the design process can be accelerated when organizing it by such feature operations, and basing it on a generative, constraint-based paradigm in which design instances are computed based on dimensional and geometric constraints.

Such feature operations are different from the familiar CSG style of constructing solids. In a CSG construction, a solid is built from standard primi-

[*] Supported in part by ONR contract N00014-90-J-1599, by NSF Grant CDA 92-23502, and by NSF Grant ECD 88-03017.

[†] Appeared in *CAD 27*, Sep 95, 695–702.

tives by regularized Boolean operations. A solid can then be abstracted as a tree structure in which the leaves are solid primitives and the interior nodes are Boolean operations and rigid-body transformations. In contrast, feature construction as done in Pro/Engineer is strictly sequential, and adds features whose shape and placement inseparably depend on the prior existing geometry. Feature operations such as the creation of protrusions and cuts seem to correspond to Boolean operations, and other operations such as rounds and chamfers to Brep modifications. However, there is a fundamental difference in the semantics of the operations. In the pure CSG construction, the semantics of a Boolean operation is well-defined; e.g., [5, 9, 10, 11]. But the semantics of feature operations, as pointed out in [6, 7, 12, 13], is loosely defined and exhibits unexpected anomalies and errors in current CAD systems.

In this paper, we define a sound semantics for the creation of generated features and discuss an implementation of it. There is a difference between the feature operations as defined here and the traditional approach of using regularized Boolean operations with solid operands that are defined independently of each other. Here, the Boolean operation is carried out on two solids where one depends on the other. This difference suggests that existing geometric core modeling systems such as ACIS might evolve away from the classical architecture that stresses well-defined operations with solids as operands, towards heavier reliance on surface-based operations. While surface-based operations, such as those found in Pro/Engineer, appear to have better performance, we find that a conceptual explanation of the exact semantics can well utilize the CSG vocabulary. That way, the results of an operation are never undefined.

Our work is part of a larger investigation of a suitable architecture for CAD systems. It is based on a neutral, high-level design representation, called Erep (editable representation), that allows design modifications based on a general design paradigm. In [8], we have described the general structure of the architecture and the editable representation on which it is based. We have argued in particular the potential for breaking down the traditional functional barriers that impede in current systems the integration of engineering design with engineering analysis, manufacturability analysis, process planning, and so on.

From a technical point of view, several research topics stand out as necessary prerequisites for our Erep-based architecture [6]. They include the neutral formalization of variational constraint solving, the semantics of feature attachment, and the generic identification of geometric elements that remains valid under regeneration of design variants, also called the persistent ID problem. This paper investigates the semantics of feature attachment. In [2] we are investigating the persistent ID problem, and in [1, 3, 4] the problem of variational constraint solving has been studied.

1.1 Feature Attachment in Commercial Systems

There are several commercial feature-based systems that have a method for feature attachment. Those methods are proprietary. To gain insight into how they might work, one needs to conduct extensive experimentation, forming hypotheses and confirming or disproving them. Since we have access only to Pro/Engineer, we cannot comment on the feature attachment rules of other systems such as I/EMS of Intergraph or I-DEAS of SDRC.

After extensive experimentation with Pro/Engineer, we believe that the feature-attachment rules of that system are based on surface operations [6, 7]. The following "semantics" is consistent with our experiments.

When generating a sketched feature, the user draws normally an open profile that is extruded into a surface in 3-space. By declaring "alignments," the user identifies graphically which other faces of the existing geometry might intersect this surface. The feature surface is trimmed by those faces and connected to the existing solid surface much in the way a blending surface is connected to the faces adjacent to a blended edge. Finally, the redundant surface is trimmed by a boundary traversal.

The commercial success of the feature-based design paradigm suggests that the process of feature attachment is intuitively clear to users. While this should be the case in most routine situations, special configurations can arise in which the intuition does not appear to be a clear guide to the intent of the operation. It is precisely the investigation of such borderline cases that is a prerequisite to a complete and successful implementation and is our motivation in this paper.

One might argue that engineering design does not generate strange borderline cases. While this may be true for finished detail designs, it is doubtful whether borderline situations can be avoided routinely during all intermediate stages of the design process. Moreover, since the generative design paradigm stresses *automatic* regeneration when design constraints have changed, it is mandatory to have explored all possible situations that can arise — unless one is to risk a failed design or, worse, abort of the design system. The comprehensiveness of our semantics is, of course, a consequence of grounding it in solid operations that have a mathematically well-defined semantics.

The semantics we define in this paper is not intended to be a final statement. Rather, we hope that the paper by Shapiro et al.[13], and this paper, facilitate a discussion as to what should be considered *natural* feature semantics. We believe that the ultimate arbiter of what constitutes expected meaning should be the end-user, and we hope that our work provokes others to articulate better interpretations of the design gestures in CAD systems today and in the future.

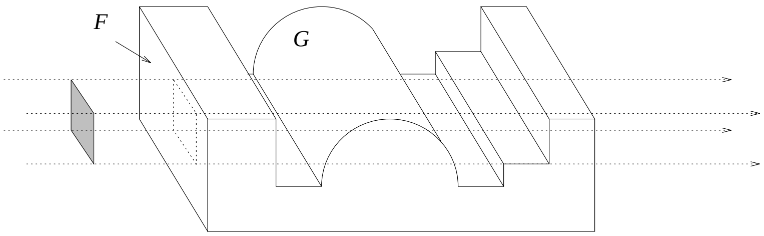


Figure 1: Ambiguity when cutting from face F to face G

2 Generated Features

We consider generated features that are based on a planar profile, swept into a three-dimensional shape. To simplify matters even further, we concentrate on extrusions and revolutions only. Such a sweep is to be modified by feature attributes that govern the exact interpretation of the sweep operation and determine how the existing geometry will be changed.

Conceptually, a proto feature is created that consists of a sweep of sufficient extent to accommodate the chosen attributes. In the case of blind extrusions, explicit dimensions determine the proto feature which is then used unchanged. Otherwise, the proto feature may be considered unbounded in the case of extrusions, or revolved by 360 degrees in the case of revolved features.[1] Although most situations are intuitively easy to grasp, special configurations can arise that make it difficult to define the feature unambiguously.

We address in this section the global process of generating features from extrusions and revolutions, without considering some of the finer points that depend on the feature being a protrusion, a cut, or a restriction. For an illustration of possible ambiguities consider Figure 1: A cut with a rectangular profile is to be made, from face F to face G. It is unclear whether the cut should go through the central half-cylinder or not. Our rules for resolving such ambiguities are explained in Section 3. Note that Pro/Engineer will always cut through the central half-cylinder when using the *thru-face* attribute.

2.1 Extrusions and Revolutions

A *profile* C is defined in a *sketching plane* P. The sketching plane can be the support of a planar face or a datum plane defined separately. The profile must be a set of closed curves defining interior and exterior. The profile interior is finite[2] and is used to define the interior and exterior of the sweep.

[1] We can restrict the proto feature to the intersection of the unbounded extrusion with the bounding box of the existing geometry.

[2] Infinite interior would be acceptable for cuts and restrictions but complicates the exposition unnecessarily.

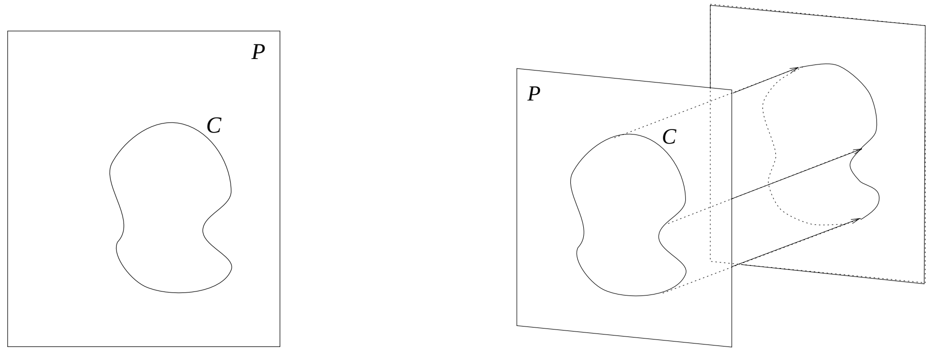


Figure 2: Extrusion of Profile C

The definition of C is based on variational constraints that are solved when the feature is created. The constraints define both the intrinsic shape of C as well as its position relative to the existing geometry. More precisely, the constraints position the profile C with respect to the projection of the existing geometry onto the plane P.

Let C be a closed profile in the plane P. The *extrusion* of C is defined to be a solid obtained as follows: The profile C is swept perpendicular to P, up to a parallel plane P', resulting in one or more tubular surfaces. The interior of the contour in P and in P' defines planar faces that together with the tubular surfaces bounds a solid volume. The solid volume so obtained is the extrusion of C. See also Figure 2.

Let C be a closed profile in the plane P, A be an oriented line in P that does not intersect C, except, possibly, in finitely many isolated points. Assume that no part of C is to the left of A. The *toroidal revolution*[3] of C is the solid obtained by revolving C, and its interior, about the axis A, by a positive angle not greater than 360 degrees. In case the revolution is by 360 degrees, the resulting surfaces are topologically tori. As before, planar end faces are required to complete a solid surface in the case of an incomplete revolution. See also Figure 3. If C intersects A, the intersection points in general become nonmanifold points on the surface of the resulting solid.

Let C be a closed profile in the plane P, A be an oriented line in P that intersects C in finitely many segments. Assume that no segment of C is to the left of A, and that every component of C intersects A in at least one segment. The *spherical revolution*[4] of C is the solid obtained by revolving C, and its interior, about the axis A, by a positive angle not greater than 360 degrees. See also Figure 4. In case the revolution is by 360 degrees, the segments on A

[3] A simple closed contour revolved around an axis it does not intersect generates a surface of revolution whose topology is a torus.

[4] The topology of such a revolution is that of a sphere with an interior segment on the axis of revolution. The interior segment can be eliminated by regularization.

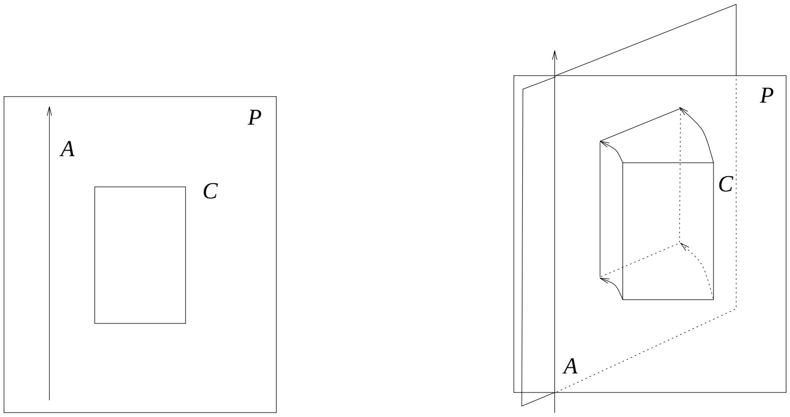


Figure 3: Toroidal Revolution of Profile C

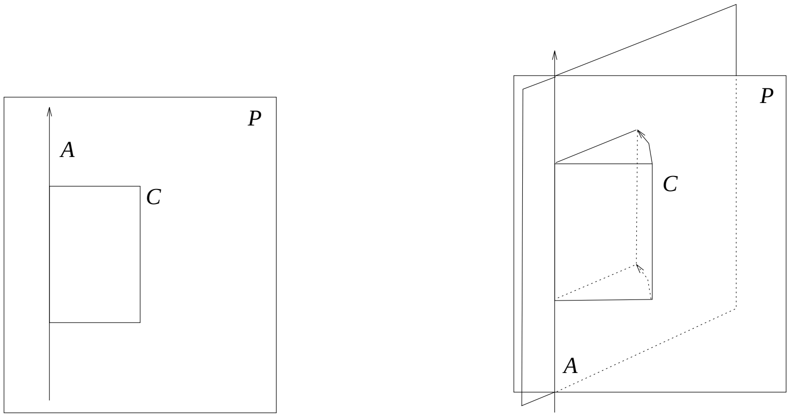


Figure 4: Spherical Revolution of Profile C

are interior to the solid and resulting topology is a collection of spheres. If C intersects A, the intersection points in general become nonmanifold points on the surface of the resulting solid.

In the following, we consider revolutions in which both spherical and toroidal topologies are generated from full revolutions. The formal specification is straightforward but tedious.

2.2 Shape Attributes

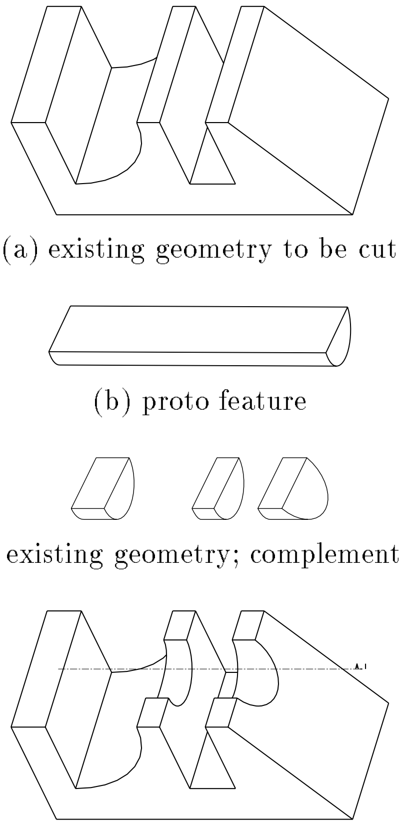
Extrusions and revolutions are generated based on shape attributes. The simplest case is a blind extrusion or revolution:

1. *One-Sided and Blind, or Bi-Sided and Blind*
   A blind extrusion is determined from an explicit dimension d specifying the depth of the extrusion. If the extrusion is one-sided, the solid is on one side of the plane P. For a positive value of d, this side is in the direction of the plane normal; for a negative value, the solid is on the opposite side. A bi-sided extrusion is one in which the extruded solid is bisected by the plane P, with the solid extending by $|d/2|$ to both sides of P. The sign of d is irrelevant in this case.

   A blind revolution is one in which the contour is revolved about an oriented line A by an angle α. For one-sided revolutions, a positive angle α is counterclockwise about A as seen from the direction in which A points. Thus, for positive angles less than 180 degrees, the revolved solid lies on the side away from the normal of P.

The other situations involve explicit or implicit face or plane identifications. Conceptually, these operations can be thought to have two phases. In the first phase, a blind extrusion of sufficient extent is computed, thereby obtaining a *proto feature*. The proto feature is intersected with the existing geometry. The result is a set of volumes S_i and their relative (regularized) complements S_c with respect to the proto feature. From these volumes the final operation is defined. For instance, if the feature is a *from-to* feature, then we select those volumes or their complements that include the ones bounded in part by these faces and those that lie "in-between." The semantics of "in-between" has to be defined with care and depends on the geometry of the selected faces and on the nature of the feature operation. In Figure 5(a), we specify the curved face to be the *from* face and the rightmost face to be the *to face*. Then the Figure 5(d) is the geometry after the cut. We consider the following attribute combinations:

2. *From-To*
   Intuitively, the *from-to* operation is a sweep that begins at a face or face plane designated as *from*, and ends at a face or face plane designated *to*.



(a) existing geometry to be cut

(b) proto feature

(c) intersection with existing geometry; complement volumes not shown.

(d) subtraction from the existing geometry with the selected volumes

Figure 5: Two phases of feature attachment.

3. *From-Next, Previous-To*
   The *from* or *to* face or face plane is explicitly designated and is called the *explicit* face. The *previous* face is the face preceding the explicit face in the direction of the sweep, the *next* face is the one following the explicit face in the direction of the sweep.[5] The operations are now like the *from-to* operations using a combination of explicit and implicit faces.

4. *FromAll-To, From-ThroughAll*
   Here, *fromall* means that all volumes preceding the *to* face or face plane are used, and *throughall* means that all volumes following the *from* face or face plane are used, in the direction of the sweep. These operations make sense only for extrusions.

[5] Strictly speaking, there need not be a single *previous* or *next* face, as discussed later.

The details of interpreting these attributes depend on the feature being a cut, a protrusion, or a restriction. The distinction between using a face vs. a face plane is made so that we can take advantage of the fact that a planar face has the supporting plane as its natural extension. In this case we can avoid some of the possible ambiguities that arise with curved faces.

3 Semantics of Cuts, Protrusions and Restrictions

We define the feature operations of cut, protrusion and restriction, paying close attention to the possibility that the conceptual view of the designer, formed by a visual design interface, does not necessarily match the technical view a system implementor has of them.

It is convenient to think of the three feature operations as being synonymous with regularized Boolean operations, and we explain their semantics using this vocabulary. Roughly speaking, a cut is a regularized volume subtraction from existing geometry. A protrusion is a regularized union, and a restriction is a regularized intersection.[6] However, we note that the operations need not be so implemented, and that the semantic properties to be defined encourage a mix of partial Booleans and boundary-based operations instead.

In a number of cases we use a volume decomposition of the proto feature by the existing geometry to define attachment rules. In this case, we consider a set S_i of volumes obtained conceptually by intersecting the proto feature with the existing geometry. The set S_c of volumes is the (regularized) relative complement of the volumes S_i with respect to the proto feature. The volume sets make the semantic definitions unambiguous in those cases where an intuitive interpretation of the attachment attributes is not immediately obvious. This point is taken up again in the last section.

3.1 Explicitly Bounded Features

Blind Features

Blind features are semantically straightforward. In essence, they are not different from the customary CSG design vocabulary. Blind cuts, protrusions and restrictions are semantically the corresponding Boolean operations using the explicitly dimensioned extrusions or revolutions as defined before.

From-To Features

The extent of the sweep is implied by the designated *from* and *to* faces or face planes. In either case, the direction of an extrusion must be known explicitly and determines how the *from* and the *to* faces or planes are used.

[6] A restriction could be used to organize widely separated faces into a single feature.

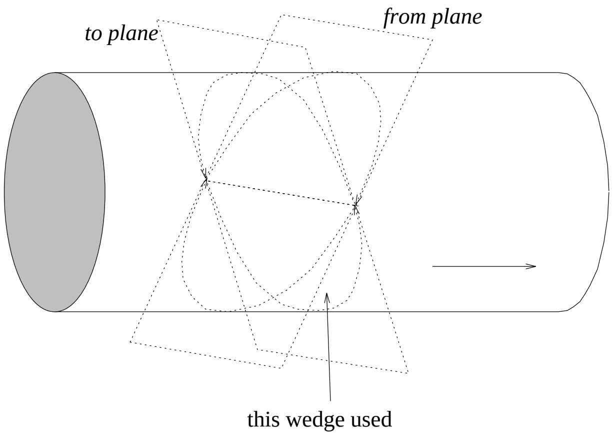


Figure 6: Wedge determination for proto feature definition.

Plane Delimiters

In the case of extrusions bounded by datum or face planes, we require that

> Neither the *from*-plane nor the *to*-plane is orthogonal to the sketching plane.

If the two planes are not parallel, then they bound four wedges of space. One of the wedges is candidate for defining the precise feature extent, and is determined by the following rules. See also Figure 6.

1. Consider the two-sided infinite extrusion of the profile. The two wedges whose intersections with this extrusion are infinite are not used.
2. Of the remaining wedges, use the one whose *from*-plane boundary precedes its *to*-plane boundary in the extrusion direction.

In Figure 6, the wedge used is the lower one because the direction of the extrusion is left-to-right.

The feature semantics for the case of *from-to* planes is defined as follows: The proto feature is the intersection of the wedge so identified with the infinite, two-sided extrusion of the profile. A cut is the regularized subtraction, a protrusion is the regularized union, and a restriction the regularized intersection, of the proto feature with the existing geometry.

In the case of a revolved feature with *from* and *to* planes we require that neither the *from*-plane nor the *to*-plane is orthogonal to the sketching plane. Consider the revolution of any point not on the axis of rotation about A in the designated orientation. Then the trajectory is a circle that is divided, in general, into four arcs by the two planes. The arcs are oriented, and two of them start at an intersection with the *from*-plane and end at an intersection with the

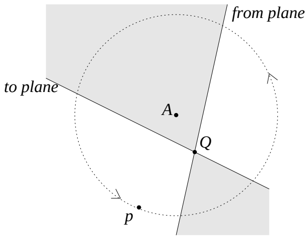


Figure 7: The two wedges used for revolved features delimited by *from* and *to* planes are shown shaded. Revolution seen in a plane perpendicular to axis A and from the direction in which A points. Q is the intersection of the *from* and *to* planes, p a generic point revolved about A.

to-plane. See also Figure 7. The two wedges in which these arcs lie, intersected with the full revolution of the profile define the proto feature.

A cut is now the regularized subtraction, a protrusion is the regularized union, and a restriction the regularized intersection, of the proto feature with the existing geometry.

From-To Face

In contrast to *from-to* plane feature definitions, face-based feature delimiters are defined based on the volumes in S_i and S_c, where the intersection volumes in S_i are used for cuts and restrictions, and the complement volumes in S_c are used for protrusions. The different conceptualization becomes necessary in view of the difficulties to define how to extend curved faces when the *from* or the *to* face do not completely intersect the proto feature. This will be further discussed later.

We explain the semantics of the feature operations for extrusions and revolutions assuming that the profile C has a connected interior. If C bounds an interior that has several components then every component is considered separately using these rules.

Let $S_f = S_i$ in the case of cuts and restrictions, and let $S_f = S_c$ in the case of protrusions. We define the semantics of the feature operation using the set S_f of volumes. The direction of sweeping must be explicitly designated by the user.

Let S_{from} be the set of volumes in S_f whose boundary contains a nonzero area of the *from* face.[7] We consider volumes as separate if their interior is not connected, and require that the set S_{from} be a singleton. Similarly, let S_{to} be the set of volumes in S_f bounded in part by a nonzero area of the *to* face.[8] This set also must be a singleton. We partition the set S_f into the following:

[7] In the case of protrusions, we require that such volumes be finite.

[8] In the case of protrusions, such volumes also must be finite.

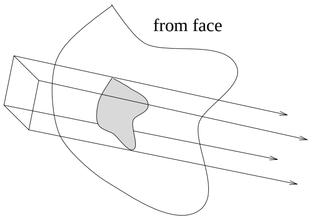


Figure 8: From face completely intersects swept volume.

1. S_{from}, containing the *from* volume.
2. S_{to}, containing only the *to* volume.
3. S_{in}, containing volumes that are "in-between' the *from* and the *to* volumes.
4. S_{out}, containing all remaining volumes.

The set S_{in} is defined differently depending on whether the bounding faces completely intersect the proto feature or not. Figure 8 illustrates the intuition of a complete intersection: More precisely, if the trajectory of every point of the contour C, and its interior, intersects a bounding face, then the bounding face completely intersects the proto feature. For complete intersecting faces it is straightforward to define the volumes S_{in}. Let V be a volume in $S_f - S_{from} - S_{to}$, p a point in V. If p is inside the volume bounded by *from* face, *to* face and their trimmed swept boundary faces, then V is in S_{in}; otherwise V is in S_{out}. Note this is the typical point classification with respect to a volume. A simple solution is to fire a ray, count the number of intersections and give the result based on whether the number is odd or even, duly considering degenerate intersections. See also Figure 9 left. We note that if one point of p satisfies the condition, then all must because the bounding faces intersect the proto feature completely.

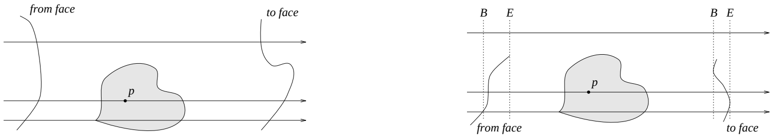


Figure 9: Definition of a volume in set S_{in}. Left: completely intersecting bounding faces. Right: partially intersecting bounding faces.

In the case of extrusions where one or both of the bounding faces have partial intersections with the proto feature, the semantic definitions are more technical. As before, we require that the sets S_{from} and S_{to} be singletons. Let B and E be two planes perpendicular to the direction of extrusion that box the area a bounding face. That is, that part of a bounding face is boxed that is on one of the bounding volumes. See also Figure 9 right. Then a volume is in S_{in} if all its interior points p are preceded by the E bound of the *from* face and precede, in turn, the B bound of the *to* face.

The semantics of extrusions is now as follows: Let the *feature volume set* V_F be $V_F = S_{from} \cup S_{in} \cup S_{to}$. Then a cut is the regularized difference, a protrusion the regularized union, and a restriction the regularized intersection of the existing geometry with V_F. For instance, the cut defined as shown in Figure 1 will extend through the central half cylinder.

In the case of revolutions, the semantics requires replacing the notion of preceding and succeeding by the corresponding ordering along a circular trajectory in the orientation of revolution. Furthermore, the boxing planes B and E are half planes that are bounded by the axis of revolution A.

3.2 Implicitly Bounded Features

The attributes *from all*, *through all*, *previous*, and *next* are implicit ways to define the extent of a sweep. Their exact meaning depends on the existing geometry, on the type of the feature operation, and on the direction/orientation of the sweep. Implicit bound designations must be paired with explicitly named faces or planes. We impose a number of restrictions to limit degeneracies.

As before, we explain the semantics of the feature operations in terms of the volumes in the sets S_i and S_c. Again, the set S_f is either S_i, for cuts and restrictions, or is the set S_c in the case of protrusions.

Previous and Next

Previous implicitly defines the *from* face of a feature extent and must be paired with an explicit *to* face. We require that the *to* face intersects the proto feature completely, and, as before, that the set S_{to} is a singleton. The implicit *from* face need not intersect the proto feature completely. Possible ambiguities are resolved by the requirement that S_{to} is a singleton. The set V_F is then defined by

$$S_{from} = S_{to} \qquad S_{in} = \emptyset$$

Thus, in the *previous-to* combination only one volume in S_f defines the feature.[9] The semantics is now as in the explicit *from to* case.

Next is symmetric to *previous* and designates implicitly the *to* face. Here, the *from* face must be designated explicitly, and must intersect the proto feature completely. Again, the implicit *to* face need not intersect the proto feature completely, and the feature volume set V_F is defined by

$$S_{from} = S_{to} \qquad S_{in} = \emptyset$$

[9] Recall that each component of the proto feature is considered separately.

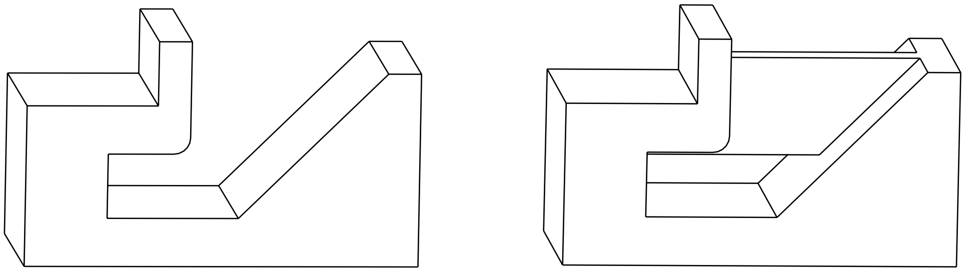


Figure 10: Creating a protrusion via *from face* to *next*. The protrusion is from right to left.

For an example of the operation see Figure 10. Note that the protrusion, proceeding from right to left, terminates at a combination of different faces.

FromAll or ThroughAll

Due to the circular topology, the interpretation of the *from all* and *through all* designations is not meaningful for revolved features. For extrusions, *from all* must be paired with an explicit *to* face, and *through all* must be paired with an explicit *from* face. The explicit faces must intersect the proto feature completely.

In contrast to *previous*, *from all* requires that all volumes preceding the *to* volume in S_f be in the set S_{in}, in addition to the *to* volume. Moreover, the set S_{from} is empty. This defines the feature volume set V_F, and with it the semantics of the operations. Similarly, *through all* requires that all volumes following the *from* face are in V_F, and that S_{to} is empty.

4 Implementation

Features can be attached using a suitable combination of Boolean operations, and we have done this in our implementation because of the architecture of ACIS. Our implementation compiles an Erep description of the design to the ACIS geometric modeling library. We discuss now the particulars of our implementation.

4.1 Feature Placement

In the case of extrusion or revolution, a feature is drawn on an plane that is chosen interactively by the user. In the graphical user interface (GUI) a sketching plane is visually identified, either by a datum plane or a planar face. The user sketches a contour on the plane as a 2D drawing with dimensional and

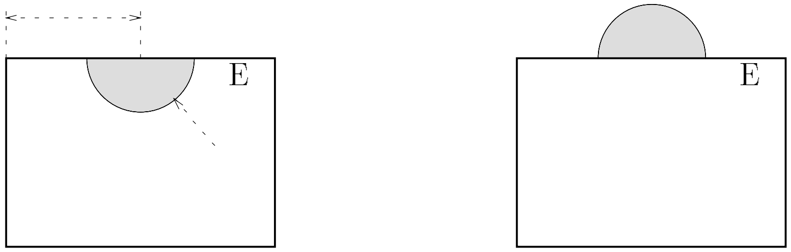


Figure 11: Two possible interpretations if the line orientation of the projected edge E is not considered.

geometric constraints including those that determine the position with respect to the (projected) existing geometry. The contour initially is a 2D structure, but is then transformed into a 3D structure on the plane initially selected. The transformation preserves all relations and constraints designated or implied by the sketch.

In the Erep, the projected geometry is recorded in an encapsulated section inside the declaration of the contour. In order to maintain the geometry consistently as intended by the user, the orientation of segments and lines must be preserved that were projected from edges, datums, and faces. If the orientation is not kept, different interpretations of the sketch would be possible, as illustrated in Figure 11. Datum axes have an intrinsic orientation in 3-space which is maintained in the projection. The orientation of projected edges is recorded explicitly by the adjacent vertices. Planar faces and planes projecting to lines are oriented by the projected face normal. This requires a persistent naming scheme as explained in [2].

4.2 Interface to Constraint-Solver

Consider Figure 11 left. The user has drawn a semicircle (light lines) and related it to the projected geometry (heavy lines). The constraint solver must keep the projected geometry fixed while computing the proper size and position of the drawn geometry. Recalling the techniques explained in [1, 4], the solver essentially treats the fixed geometry as a cluster that has been positioned already. Thus, only the sketched geometries are computed by the solver.

Although all existing geometry must be shown to the user, only some of the projected elements are actually referenced when dimensioning and constraining the sketch. Only the referenced geometry is recorded in the Erep and passed to the constraint solver. When compiling Erep, the unrecorded geometry can be reconstructed from the prior features, and if the dimensioning schema is changed, newly referenced geometry will be recorded in the changed design. This is easy with our persistent naming schema.

5 Discussion

The use of double wedges when delimiting revolved features with planes seems counter-intuitive. It would appear that the user has only one wedge in mind, particularly when the intersection of the two planes is the axis of revolution. If the planes are considered oriented, or if we work with half planes, then it is possible to define a single wedge in space whose interior limits the revolved feature. We did not do so because neither the definition of half planes, nor the orientation of datum planes, appears to be natural. While very familiar to implementors, it is not clear to us that users would think in such terms. One could find a middle ground: The user determines graphically which wedge is meant, the system internally orients planes and records design intent in terms of this internal orientation.

The rules for features with face boundaries negotiate several difficulties. The central problem is that for most curved faces there is no clear-cut mathematical definition of how to extend the face. If there were, one could extend a partially intersecting face to a completely intersecting surface and so determine what is "enclosed" between *from* and *to* face. To illustrate this point, consider Figure 12 in which we assume that F is a curved face that happens to be part of a larger surface that extends as shown in (b). Intuitively, the designated *from* face in an extrusion should eliminate all parts of the proto feature that "precede" the *from* face. In the situation shown as (a) in the upper left of the figure, we see a volume V whose classification is unclear. If the surface F is extended as mathematical surface, then V would precede F, as shown in (b). If the surface F is extended by its tangents at the planar edge of F, then V is partially intersected, (c), and no clear decision can be made. If the surface F is extended by a ruled surface whose generators are perpendicular to the extrusion direction and connects at the boundary of F, then V might follow F, (d). Moreover, in each case additional conventions are needed to define the extension mechanism unambiguously, and the conventions would not be very intuitive. Note that the ruled surface extension generalizes the bounding box idea we have used.

The concept of boxing planes in the case of partially intersecting bounding faces reduces the test whether a volume of S_f is in S_{in} to a bounding box computation. Variants of our definition could be considered. For example, our definition excludes volume V in Figure 13(a). Here, the concave *from* face has a bounding plane E that partially intersects V. Intuitively, V should be in S_{in}. If we use the plane B instead, then the interpretation of Figure 13(a) is as expected. However, in that case the volume V in Figure 13(b) would also be considered to be in the set S_{in} which is counter intuitive.

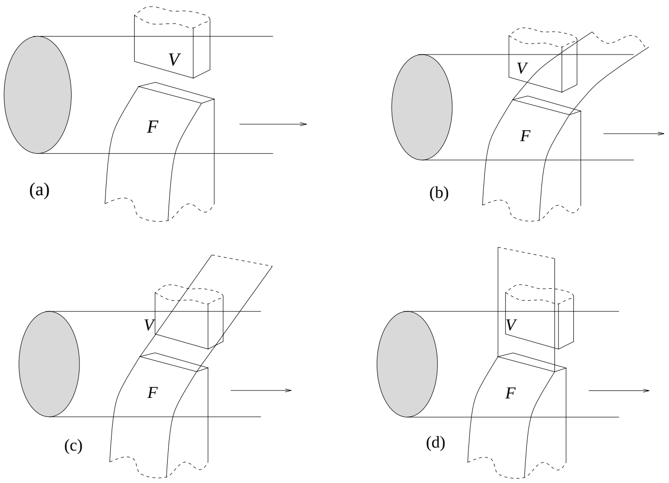


Figure 12: Alternatives for incompletely intersecting *from/to* faces: (a) *from* face F incompletely intersects circular proto feature; (b) F is extended as mathematical surface, (c) F is extended by tangent directions at the boundary, (d) F is extended by a ruling of the boundary perpendicular to the direction of the sweep.

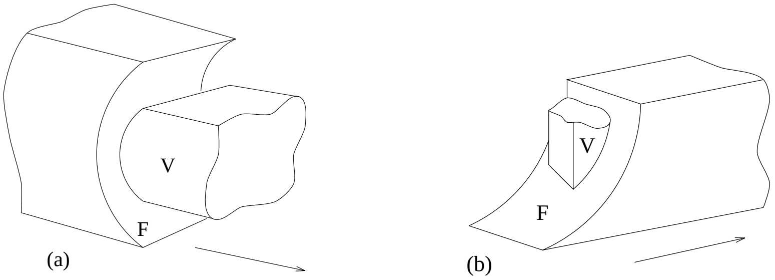


Figure 13: Determining in-between volumes from intervals: (a) use the minimum value of the bounding box to include V, (b) use the maximum value of the bounding box to exclude V.

Ultimately, the notion of "in-between" rests on a concept of separation that is unambiguous only for completely intersecting bounding faces. Partially intersecting boundaries are a necessity unless we allow open profiles. But open profiles have more difficult semantic problems; [6, 7]. A useful device, therefore, might be to allow users to define datum surfaces for the purpose of separating volumes in ambiguous positions.

Our semantic definitions have been given in terms of regularized Boolean operations. This was done so as to define unambiguously what each feature operation means. It also implies that the feature operations can be implemented literally using Booleans. This could be attractive in legacy systems in which Boolean operations are a prominent aspect of the system architecture. However, the manner in which the features have been defined implies a locality that should be exploited in any implementation. For example, it is clear that the definition of a contour for extrusion already reduces face-intersection candidates: Faces whose projections do not intersect the contour clearly could not intersect the proto feature in 3-space.

References

[1] W. Bouma, I. Fudos, C. Hoffmann, J. Cai, and R. Paige. A geometric constraint solver. *Computer Aided Design*, page to appear, 1994.

[2] V. Capoyleas, X. Chen, and C. M. Hoffmann. Generic naming in generative, constraint-based design. Technical Report 94-011, Purdue University, Computer Science, 1994.

[3] I. Fudos. Editable representations for 2d geometric design. Master's thesis, Purdue University, Dept. of Comp. Sci., 1993.

[4] I. Fudos and C. M. Hoffmann. Correctness proof of a geometric constraint solver. Technical Report 93-076, Purdue University, Computer Science, 1993.

[5] C. M. Hoffmann. *Geometric and Solid Modeling.* Morgan Kaufmann, San Mateo, Cal., 1989.

[6] C. M. Hoffmann. On the semantics of generative geometry representations. In *Proc. 19th ASME Design Automation Conference*, pages 411–420, 1993. Vol. 2.

[7] C. M. Hoffmann. Semantic problems in generative, constraint-based design. In *Practice of Computer Aided Geometric Design — What CAD Systems are Really Capable of.* Teubner Verlag, 1993.

[8] C. M. Hoffmann and R. Juan. Erep, an editable, high-level representation for geometric design and analysis. In P. Wilson, M. Wozny, and M. Pratt, editors, *Geometric and Product Modeling*, pages 129–164. North Holland, 1993.

[9] M. Mäntylä. *An Introduction to Solid Modeling.* Computer Science Press, 1988.

[10] A. Requicha. Mathematical models of rigid solids. Technical Report Memo 28, University of Rochester, Production Automation Project, 1977.

[11] A. Requicha and H. Voelcker. Constructive solid geometry. Technical Report Tech. Memo 25, University of Rochester, Production Automation Project, 1977.

[12] J.R. Rossignac. Issues on feature-based editing and interrogation of solid models. *Computers and Graphics*, 14:149–172, 1990.

[13] V. Shapiro, H. Voelcker, and D. Vossler. Boundary to CSG conversion: current status and open issues. IFIP WG5.2 Conference on Geometric Modeling for Product Realization, Rensselaerville, 1992.