ELSEVIER

Computer-Aided Design 37 (2005) 1081–1093

COMPUTER-AIDED DESIGN

www.elsevier.com/locate/cad

Geometry-based semantic ID for persistent and interoperable reference in feature-based parametric modeling

Yan Wang*, Bartholomew O. Nnaji

Center for e-Design, University of Pittsburgh, 1048 Benedum Hall, Pittsburgh, PA 15261, USA

Received 28 April 2004; received in revised form 25 November 2004; accepted 30 November 2004

Abstract

In current feature-based parametric design systems, the reusability principle is not fully supported as it was expected. Unpredictability and ambiguity of models often happen during design modification within one system as well as among different systems. This reference deficiency significantly reduces the power of feature-based parametric modeling, where geometry re-evaluation generates unexpected shapes. In this paper, a sufficient condition of B-Rep variance based on geometry continuity in parametric complex Euclidean ($\mathbf{P}^p\mathbf{C}^3$) space is proposed. Shape and relation parameters are differentiated in $\mathbf{P}^p\mathbf{C}^3$, thus parametric family can be defined. A semantic id scheme based on continuity of geometry is developed to solve the problem of naming persistency and to improve interoperability of CAD feature modeling. Hierarchical namespaces localize entity creation and identification. All geometric and topological entities are referred uniformly based on surface ids, and topology semantics is retained in id itself.

Keywords: Parametric family; Continuity; Persistent naming; Topological ID; Feature-based parametric modeling

1. Introduction

Feature-based parametric modeling has advanced in the last 15 years and rapidly become the mainstream of mechanical design. Nevertheless, parametric modeling systems share a number of shortcomings related to design process [1]. For example, lack of common definition of features introduces extra barriers in model data exchange and sharing; chronological dependency of features in the modeling process reduces the flexibility of modeling sequences; sequential feature evaluation would easily generate unstable and unpredictable geometry; ambiguous and inconsistent models would be created within one system as well as among different systems. Difficulties of design data sharing and reuse still exist in current systems.

The above modeling procedure related problems are all connected to one basic modeling issue: the semantics of feature is not captured actively and maintained throughout the modeling process using current way of feature specification. Currently, features and associated parameters are defined based on evaluated boundary geometry. The design history and the reference structure are constructed based on geometric elements evaluated in previous steps. The chronological dependency between features is volatile during the design process. As a result, feature interaction affects the interpretation of features.

Lack of common standards of feature definition and representation prohibits extensive design exchange and data sharing. Features in different systems are created in proprietary formats. Interoperability during design collaboration and data exchange can only be achieved at the pure geometry level using neutral formats. Comprehensive information of design features and parameters is not transferable within a heterogeneous environment. Interoperability at the level of feature semantics is needed to enable the reuse of original models.

The primary symptom of lack of semantics in feature specification is persistent naming problem, which is common in current commercial parametric modeling systems. In these systems, evaluated topological entities

* Corresponding author. Tel.: +1 412 624 9830.
E-mail addresses: ywang@engr.pitt.edu (Y. Wang), nnaji@engr.pitt.edu (B.O. Nnaji).

0010-4485//$ - see front matter © 2004 Elsevier Ltd. All rights reserved.
doi:10.1016/j.cad.2004.11.009

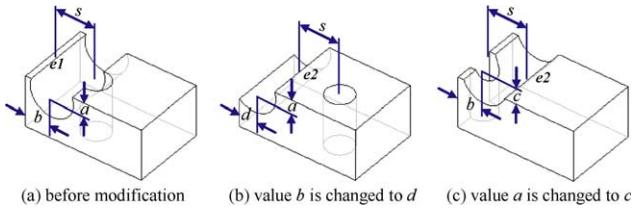


Fig. 1. An example of dramatic topology change known as naming persistency problem.

(e.g. faces, edges, and vertices) in a Boundary Representation (B-Rep) model could be references to new features. The change of a feature may directly affect the features that have reference dependency on it during the model re-evaluation. Some features at later steps may refer to a different entity unexpectedly, or even cannot find the reference. Thus, an unexpected geometry is generated.

A typical example (as in Pro/Engineer®) is shown in Fig. 1(a), where a part is constructed by a protrusion and a circular cut feature, followed by a hole feature. The position of the hole is partly determined by the distance s from the center of the hole to edge $e1$, which is generated by the cut. If the distances from the center of the cut to its references are changed, by either from b to d horizontally or from a to c vertically, as shown in Fig. 1(b) and (c), respectively, the distance reference of the hole to $e1$ will jump to edge $e2$. This is because the id of the edge $e1$ was assigned to edge $e2$ after the Boolean operation of the cut, and the orientation information of edges is also used in the re-evaluation process. This naming problem produces unstable and unpredictable parametric models in variational design.

Feature evaluation inconsistency also exists among different systems. Lack of feature semantics forces current systems to use heuristic approaches to handle naming issues. This creates ambiguity and unpredictability between CAD systems in feature re-evaluation. For example, Fig. 2(a) shows a part that is constructed by two protrusions and one circular cut feature, where the location of the second protrusion is constrained by distances a and b. At the same time, the position of the cut is defined by distances c and d. If the values of a and b are set to zeros, the desired design should be the one in Fig. 2(b). However, in current systems, the cut would jump unexpectedly because of the face mergence. The updated part in Solidworks® is shown as in Fig. 2(c), and the new evaluation in Autodesk Inventor® is shown as Fig. 2(d). Both are very different from the intended one.

The direct impact of naming persistency problem is that geometry re-evaluation generates unexpected shapes. The original principle of knowledge reuse and ease of modification in parametric design is not followed adequately by current naming and reference mechanisms. The cause behind this is that design intent of feature definition is not captured actively and thoroughly. The geometric meaning of features is interpreted totally based on the references of evaluated topological entities. The semantics of feature is degenerated into references of topological entities and parameter values, as illustrated in Fig. 3. It causes inconsistency and unpredictability of geometric models. This inconsistency significantly degrades the efficiency of feature-based parametric modeling method.

One fundamental issue associated with feature semantics representation is parametric family. A parametric solid model corresponds to a class of solids, but there is no formal definition or standard for what this class is [2]. Without general understanding of this problem, heuristic and incompatible methods of feature definitions are used in CAD industry. While CSG models are globally parameterized, B-Rep models need extra boundary evaluation steps to apply parametric modeling, which causes the complexity of parametric family. It is not clear how to generate and differentiate members of a family, how to describe a family generally, and how to represent and retrieve the common semantics of family members.

Fig. 4 shows a group of parts within one parametric family yet with very different topological constructs. Common descriptions at the semantic level instead of the topological level are needed to categorize models. Until the parametric family is understood systematically, the efforts to allow exchange of parametric representation are likely to remain ad hoc. Exploring the nature of parameterization is an important portion of understanding the semantics involved in feature modeling.

In order to eliminate the flaws of current systems and improve the robustness and efficiency of the parametric modeling method, maintaining feature construction semantics is necessary. In this paper, a geometry-based semantic approach for design feature representation and reference is

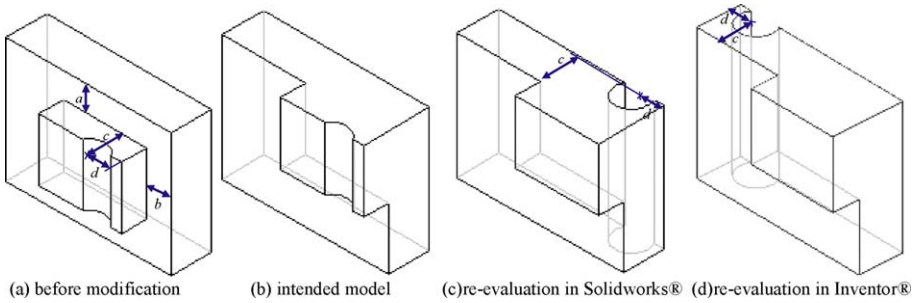


Fig. 2. Unpredictability and inconsistency of feature re-evaluation among different systems.

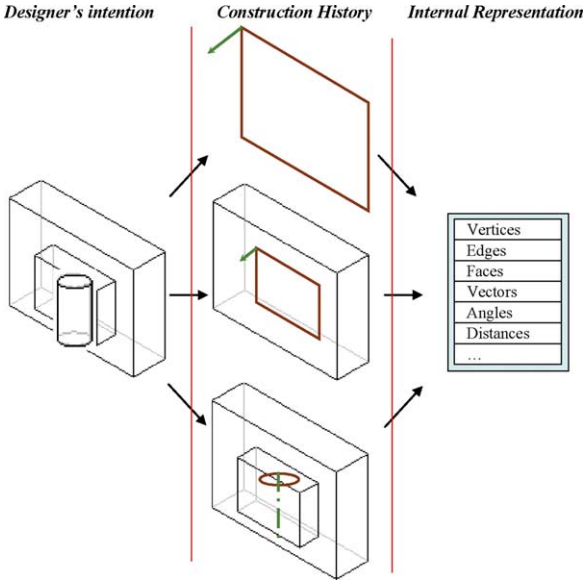


Fig. 3. Feature semantics is degenerated and lost during feature construction.

presented, in which topological entities are named based on geometric construction, and semantics of topology is embedded in ids. In Section 2, related research work is reviewed. Section 3 proposes a sufficient condition for B-Rep variance based on geometry continuity. Based on that, a semantic ID scheme is developed as described in Section 4. Detailed id update and implementation are discussed in Section 5.

2. Background

Some solutions of the persistent naming issue have been proposed. In the research of E-REP [3,4], a topology-based naming method is used. New topological entities are named based on the referred old entities during the feature construction. For example, in an extrusion, a new edge is named by reference to the sweeping vertex, whereas a new face is named by reference to the sweeping edge. When model is re-evaluated, new entities should be identified and matched to old entities. The matching of an entity is realized through a local comparison of topological neighborhoods by a spectral graph isomorphism algorithm, as well as entities' orientation information. This work had a creative idea of naming entities by construction semantics. However, graph isomorphism used in topology matching has combinatorial computational costs when dealing with a complex part.

Comparatively, Kripac's topological ID system [5,6] names a face based on a step id (identifying the particular step that the face is created during the feature operations), a face index within that particular step, and the type of corresponding surface. Edges and vertices are identified by the names of adjacent faces. Each model maintains a face modeling history during the construction. This face modeling history is used to map the new entities to the old ones if the topology of the model is changed. This global matching approach involves expensive graph isomorphism procedures in each model re-evaluation.

The Open Cascade Application Framework [7] had some indirect exposure on its naming scheme, though commercial software vendors normally keep the naming scheme as their technical secrets. It uses the local topology matching method, but supporting more features based on faces and face history. Operators are defined to retrieve construction history. Similarly, the difficulty is that local matching is not robust and reliable as the complexity of design increases.

Wu et al. [8] identify faces by two names. The Original Name (ON) of a face records the feature's generating mode and the location of the face in the feature, while the Real Name (RN) of the face contains its ON and the parametric space information. New faces generated by Boolean operations will inherit the original faces' ONs. Edges and vertices are named only by RNs, consisted of adjacent faces' RNs and parametric space information. The authors had a good observation to include parametric information of surfaces in topological ids, but ended into the old trap of enumeration method to identify parameter values.

Regarding the basic issue of parameterization, a parametric family of solids is defined based on topological mapping between cell complexes in the work of Stewart [9] and Raghothama-Shapiro [10,11]. That is, if any cell of B-Rep model K can be mapped to a cell of B-Rep model L, K belongs to the parametric family of L. This approach provides a necessary condition for B-Rep variance (BR-variance) and parametric family classification. Nevertheless, sufficient conditions for BR-variance in parametric modeling still remain unresolved.

In principle, names serve identification, access, and mnemonic purposes. For identification purpose, name should be unique within the boundary of a system. Furthermore, names should provide enough information of how to access entities. Names may also contain necessary

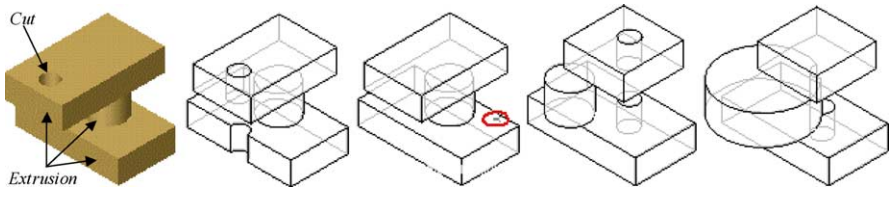


Fig. 4. Parametric family members exhibit different topology as parameter values change.

logical information for interpretation and induction. A general naming system should support names with the following characteristics: uniqueness (names should be globally unique), scope (names should be globally valid and location neutral), scalability (the number of entities and accesses are independent of naming scheme), readability (readable, interpretable, and transcribable), mobility (subject to re-location because of varying parameters and attributes), and descriptiveness (support lookup based on certain attributes).

To systematically solve the reference persistency issue of modeling system, we need to understand the fundamental problem of parametric family and feature semantics of B-Rep models. Compared to geometry, topology is rather unstable and volatile in feature-based parametric modeling. Small adjustment of some parameters may cause topology to change dramatically, which is the root of the naming persistency problem. A general solution is to identify geometric entities (e.g. surfaces, curves, and points) and topological entities (e.g. faces, edges, and vertices) by information that is more stable during model construction. This reference information should explicitly specify how feature is constructed, that is, the ids of entities should possess the semantics of features. Feature modeling through semantic-level representation and reliable and meaningful reference mechanisms are vital to capture more design intent, improve interoperability, and enhance reliability. Modifying the semantics of feature should be carefully monitored in order to make feature modeling more powerful than pure geometric modeling.

3. Sufficient condition for BR-variance

To generally define the parametric family of a solid, sufficient conditions for BR-variance are needed. A sufficient condition for BR-variance based on geometric continuity is proposed for a general definition of parametric family. Here, continuity means: throughout a valid parameter range, small changes in a solid's parameter values result in small changes in the geometry of B-Rep. It is difficult to organize variational or parametric families based on topology continuity. While adjacency of bounded geometric information (topology) is volatile in the family of variational geometry, the unbounded geometric information (geometry) is more stable.

3.1. Continuity of parametric geometry

Poncelet's continuity principle states that if, from the nature of a particular problem, a certain number of solutions are expected, and if in any particular case this number of solutions is found, then there will be the same number of solutions in all cases, although some solutions may be imaginary [12,13]. For instance, two circles intersect in two points, so it can be stated that every two circles intersect in two points, although the points may be imaginary or may coincide, as shown in Fig. 5. If considered in a complex space instead of a real one, the loci of the two intersection points of the circles are continuous with respect to the distance between the two centers of the circles.

If we extend Euclidean space to complex Euclidean space, continuity of geometry is easily observable. In an even-dimensional Euclidean space $\mathbf{R}^{2n}$, points are ordered sets of $2n$ real numbers $(x_1, \ldots, x_n, y_1, \ldots, y_n)$, where x_k, $y_k \in \mathbf{R} (k=1, \ldots, n)$. If a complex structure is introduced as $z_k = x_k + \mathbf{i}y_k$ $(k=1, \ldots, n)$, we shall call the space whose points are ordered sets of n complex numbers

$$Z = (z_1, \ldots, z_n) \tag{1}$$

the n-dimensional complex Euclidean space, denoted by $\mathbf{C}^n$.

For any point p, $p \in \mathbf{R}^n$, there is an infinite number of points q's, $q \in \mathbf{C}^n$, such that there is an orthogonal projection function f: $\mathbf{C}^n \to \mathbf{R}^n$, $f(q)=p$. The 3-dimensional Euclidean space $\mathbf{E}^3$ is the projected real subspace of complex Euclidean space $\mathbf{C}^3$.

In the domain of parametric design, if we add p more dimensions which represent real parameter t_j's ($t_j \in \mathbf{R}$, $j=1,\ldots,p$) into $\mathbf{C}^3$, we have a $p \times 3$-dimensional parametric complex Euclidean space denoted by $\mathbf{P}^p\mathbf{C}^3$, where $\mathbf{P}^p\mathbf{C}^3 = \mathbf{R}^p \times \mathbf{C}^3$. There are two types of parameters, shape parameters (s-parameters) and relation parameters (r-parameter), associated with each geometric object. For example, in a planar circle

$$\begin{cases} x = a + r\cos\theta \\ y = b + r\sin\theta \end{cases} \tag{2}$$

θ is an s-parameter and a, b, r are r-parameters. A $\mathbf{P}^p\mathbf{C}^3$ space including m-dimensional s-parametric subspace and n-dimensional r-parametric subspace can be further denoted by $\mathbf{P}^{m\times n}\mathbf{C}^3 = \mathbf{R}^m \times \mathbf{R}^n \times \mathbf{C}^3$. The BR-variance and continuity for parametric family are defined in $\mathbf{P}^p\mathbf{C}^3$.

A curve in $\mathbf{C}^3$ is a map $\gamma(t)$: $\mathbf{R} \to \mathbf{C}^3$, where t ($t \in \mathbf{R}$) is an s-parameter of γ. In $\mathbf{P}^p\mathbf{C}^3$, γ: $\mathbf{R} \to \mathbf{R}^{(p-1)}\mathbf{C}^3$ $(p \geq 1)$ is

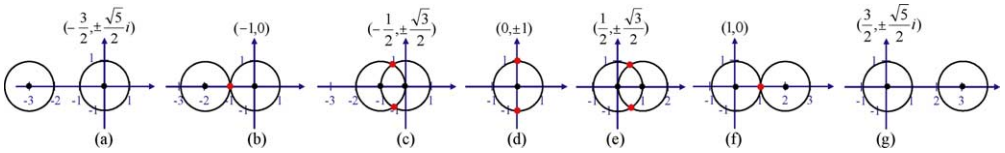


Fig. 5. An example of geometric continuity of intersection points in complex space.

a hyper-curve. Similarly, a surface in C^3 is a map $\sigma(u,v)$: $\mathbf{R}^2 \rightarrow \mathbf{C}^3$, where u and v ($u, v \in \mathbf{R}$) are s-parameters of σ. In $\mathbf{P}^p\mathbf{C}^3$, σ: $\mathbf{R}^2 \rightarrow \mathbf{R}^{(p-2)}\mathbf{C}^3$ ($p \geq 2$) is a hyper-surface.

A curve $\gamma(t)$ is $\mathbf{C}^k$ continuous with respect to t in the neighborhood of t_0 if and only if

$$\lim_{t \to t_0} \frac{\partial^k \gamma(t)}{\partial^k t} = \frac{\partial^k \gamma(t_0)}{\partial^k t},$$

and $\gamma(t)$ is $\mathbf{C}^{k-1}$ continuous. $\gamma(t) \in \mathbf{P}^{(p-1)}\mathbf{C}^3$ ($t \in \mathbf{R}$) is $\mathbf{C}^0$ continuous with respect to t in the neighborhood of t_0 if and only if $\lim_{t \to t_0} \gamma(t) = \gamma(t_0)$. Similarly, a surface $\sigma(u,v)$ is $\mathbf{C}^k$ continuous with respect to u and v in the neighborhood of (u_0, v_0) if and only if

$$\lim_{u \to u_0} \frac{\partial^k \sigma(u,v)}{\partial^k u} = \frac{\partial^k \sigma(u_0,v)}{\partial^k u},$$

$$\lim_{v \to v_0} \frac{\partial^k \sigma(u,v)}{\partial^k v} = \frac{\partial^k \sigma(u,v_0)}{\partial^k v},$$

$$\lim_{u \to u_0} \frac{\partial^k \sigma(u,v)}{\partial^{k-1} v \partial u} = \frac{\partial^k \sigma(u_0,v)}{\partial^{k-1} v \partial u},$$

$$\lim_{v \to v_0} \frac{\partial^k \sigma(u,v)}{\partial^{k-1} u \partial v} = \frac{\partial^k \sigma(u,v_0)}{\partial^{k-1} u \partial v},$$

and $\sigma(u,v)$ is C^{k-1} continuous.

$\sigma(u,v) \in \mathbf{P}^{(p-2)}\mathbf{C}^3$ ($u, v \in \mathbf{R}$) is $\mathbf{C}^0$ continuous with respect to u and v in the neighborhood of (u_0, v_0) if and only if

$$\lim_{\substack{u \to u_0 \\ v \to v_0}} \sigma(u,v) = \sigma(u_0, v_0).$$

The set of bounding surfaces of a solid object o in space $\mathbf{P}^p\mathbf{C}^3$, bs(o), is a set of surfaces, $\forall \sigma$, $\sigma \in$ bs(o), such that $\exists p$, $p \in \sigma$, $\exists a$, $a \in \varepsilon$-neighborhood of p, $\exists b$, $b \in \varepsilon$-neighborhood of p, $a \in$ o, $b \notin$ o. The set of bounding curves of a solid object o in space $\mathbf{P}^p\mathbf{C}^3$, bc(o), is the set of curves, $\forall \gamma$, $\gamma \in$ bc(o), such that $\forall p$, $p \in \gamma$, $\exists \sigma$, $p \in \sigma$, $\exists \delta$, $p \in \delta$, $\sigma \in$ bs(o), $\delta \in$ bs(o).

In $\mathbf{P}^p\mathbf{C}^3$ space, two curves always intersect, either at real points, imaginary points, or infinity. If two curves have an r-parameter r, the locus of intersection of the curves is a curve with r as its s-parameter. Similarly, if two curves have r-parameters q and r, the locus of intersection of the curves is a surface with q and r as its s-parameters. As an example, Fig. 6 shows the shape of the intersection by the two circles in Fig. 5, which is projected to 4D subspaces.

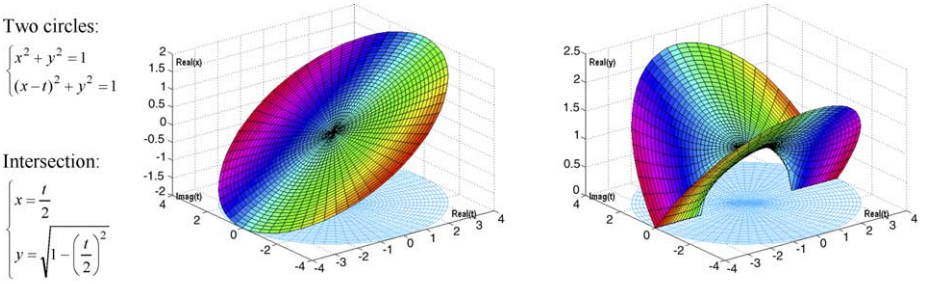
An intersection curve with respect to r ($r \in \mathbf{R}$) of two curves $\gamma(s)$ and $\xi(t)$, $\chi(\gamma(s), \xi(t), r)$, is a curve of r, where $\forall p$, $p \in \chi(\gamma(s), \xi(t), r)$, $p \in \gamma$, $p \in \xi$. An intersection surface with respect to q and r ($q, r \in \mathbf{R}$) of two curves $\gamma(s)$ and $\xi(t)$, $\chi(\gamma(s), \xi(t), q, r)$, is a surface of q and r, where $\forall p$, $p \in \chi(\gamma(s), \xi(t), q, r)$, $p \in \gamma$, $p \in \xi$.

A solid object o is $\mathbf{C}^0$ continuous with respect to an r-parameter r within interval $[a, b]$ in space $\mathbf{P}^p\mathbf{C}^3$, if $\forall \gamma(s)$ ($s \in \mathbf{R}$), $\gamma(s) \in$ bc(o), $\forall \xi(t)$ ($t \in \mathbf{R}$), $\xi(t) \in$ bc(o), such that $\chi(\gamma(s), \xi(t), r)$ ($r \in \mathbf{R}$) is $\mathbf{C}^0$ continuous with respect to r on $r \in [a, b]$. Similarly, a solid object o is $\mathbf{C}^0$ continuous with respect to r-parameters q and r within interval $[a_1, b_1] \times [a_2, b_2]$ in space $P^p\mathbf{C}^3$, if $\forall \gamma(s)$ ($s \in \mathbf{R}$), $\gamma(s) \in$ bc(o), $\forall \xi(t)$ ($t \in \mathbf{R}$), $\xi(t) \in$ bc(o), such that $\chi(\gamma(s), \xi(t), q, r)$ ($q, r \in \mathbf{R}$) is $\mathbf{C}^0$ continuous with respect to q and r on $q \in [a_1, b_1]$, $r \in [a_2, b_2]$.

3.2. Sufficient condition for BR-variance

If a solid object o_1 can be transformed to another solid object o_2 with $\mathbf{C}^0$ continuity with respect to an r-parameter r, o_2 belongs to the parametric family of o_1 with respect to r. Similarly, if a solid object o_1 can be transformed to another solid object o_2 with $\mathbf{C}^0$ continuity with respect to r-parameters q and r, o_2 belongs to the parametric family of o_1 with respect to q and r. High-order parametric family can be defined in a similar way.

It is noted that parametric family should be defined with respect to r-parameters. $\mathbf{C}^0$ continuity of solid objects gives the sufficient condition of BR-variance. If a solid has the property of $\mathbf{C}^0$ continuity on certain intervals of r-parameters, the variance of boundary representation can be asserted based on bounding curves.



(a) Intersection curve in 4D complex subspace [Real(t), Image(t), Real(x), Image(x)]. The value of Image(x) is represented by shading.

(b) Intersection curve in 4D complex subspace [Real(t), Image(t), Real(y), Image(y)]. The value of Image(y) is represented by shading.

Fig. 6. The intersection of the two circles in Fig. 5 with t as s-parameter is projected to parametric complex subspaces.

In brief, if solid geometry is considered in parametric complex Euclidean space, the parametric family of a solid can be defined based on the continuity of unbounded geometry. Geometry possesses good properties of continuity. This leads to the idea of identifying topological entities with the associated geometry. Entity relations are built based on semantics of topology. Topology thus is referred indirectly by geometric construction information. This is the general principle of the semantic ID scheme described in the Section 4.

4. Semantic ID scheme

In the semantic ID scheme, information of construct relation is included in geometric ids, and geometric meaning of identification is in topological ids. Two basic components of this naming system are namespace and topology semantics.

4.1. Namespace of entity ID

The problem of simple enumeration of entity ids is that entity identification is exposed globally for the whole product structure. The data structure of enumeration is simply a linked list. Any change within the sequence will affect the identification of all following entities. Therefore, it is more protective if id assignments are localized.

Therefore, the concept of namespace of entity ID is important. If a group of entities have some common properties, these properties can form a boundary for their names, and a prefix based on these properties can be attached on each of these entity ids. In this way, the namespace of entities is divided based on the prefix. Simply from the name of an entity, some characteristics of the entity can be inferred. Re-evaluating some entities in one namespace does not affect the names of entities in other namespaces. The namespace can be organized in a hierarchical tree structure, as illustrated in Fig. 7. One namespace can be divided further into multiple subspaces

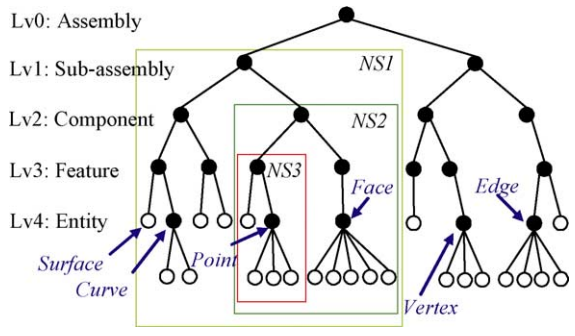


Fig. 7. Hierarchical namespace is the first step to separate entity creation and identification.

with an extra layer of prefixes in the names so on and so forth, thus forming a name tree. Two namespaces can be merged into one namespace by introducing a new root node.

Below the component level, feature is a natural selection for the boundary of namespaces. The id of a newly created geometric or topological entity will be prefixed with the id of the feature during which this feature operation is performed. The namespace of features localizes entities based on construct history. This compartmentalization process is the first step to isolate entity creation and identification. For example, each entity that is created during the constructing of the first protrusion will have Protrusion1:: as the beginning of the entity's name.

Localizing names reduces the chances of conflict. The namespace of one feature could be partitioned further to differentiate implicit (or intentional) and explicit (or geometric) features. A reliable modeling system should keep implicit and explicit features as independent as possible. An implicit feature may have multiple steps to finish the feature definition. Each step then can be assigned an independent sub-namespace. For example, a protrusion feature operation needs two steps to finish. One is defining profile, and another is trajectory definition. Each entity generated at each step is prefixed by the feature step id. The entities generated when the profile of the protrusion is defined will have Protrusion1::Profile:: as part of the ids. The entities created when the trajectory is defined will have a prefix Protrusion1::Trajectory:: in their ids. Ideally, entities defined in implicit features are independent from entities generated in explicit features. New entities created in implicit features are free of turmoil from feature re-evaluation. Thus, enumeration in implicit features will not cause big problems. However, there is no system exists that defines entities in implicit features in an absolutely independent way.

Major problems come from topological entities created in the domain of explicit features. For each of these entities, no feature steps are included in the entity names. Within the namespace of each feature, entities should be named in a meaningful and stable way. The consideration is to include stable geometric information of the entities in their identification. One way is to include all geometric information (e.g. coordinate and vector values), which is cumbersome since each re-evaluation requires complete mapping and update. A more feasible way is to include the references of geometric entities in topological ids.

4.2. Topology semantics

To improve topological entities' naming stability further, logical information of construct relations of geometry is included in geometric ids. Because surfaces are generally much more stable than curves and points, curves and points are named by surfaces. A curve is named by the ids of the two intersecting surfaces, and a point is named by the ids of the three intersecting surfaces.

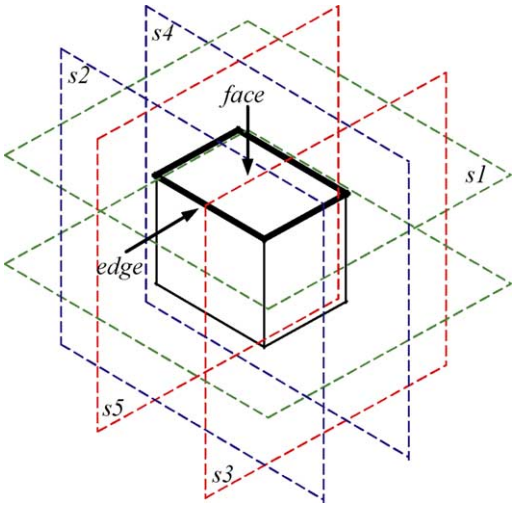
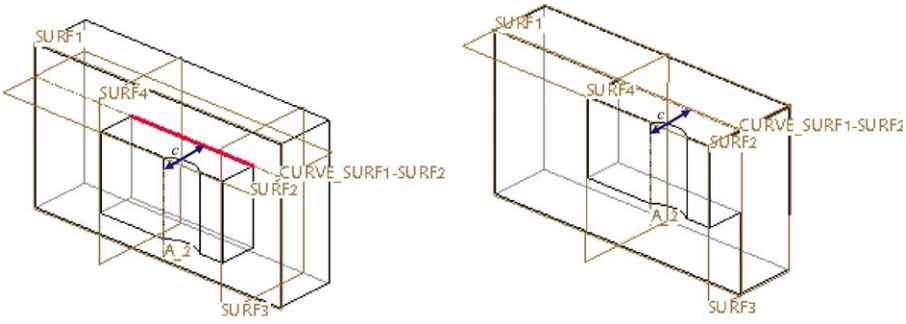


Fig. 8. An example of face bounded by surfaces and edge bounded by surfaces.

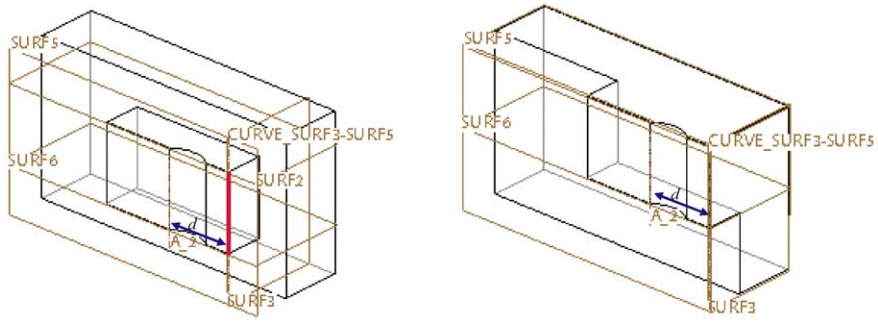
A topological entity id includes the reference to its corresponding unbounded geometric entity and the references to its boundary surface ids. By including boundary information in terms of geometry, the topological ids contain the actual semantics of topology. For example, in Fig. 8, if a face is generated by a protrusion feature $p1$ and is referring a surface $s1$ and bounded by planes $s2$, $s3$, $s4$, and $s5$, this face will have the name *FACE*(*PROTRUSION*($p1$)*::SURFACE*($s1$)−*SURFACE*($s2$), *SURFACE*($s3$), *SURFACE*($s4$), *SURFACE*($s5$)). And the edge that is referring the line formed at the intersection of planes $s1$ and $s2$ will have the name *EDGE*(*PROTRUSION*($p1$)*:: CURVE*(*SURFACE*($s1$)+*SURFACE*($s2$))−*SURFACE*($s3$), *SURFACE*($s5$)). A face id has the references of the feature namespace, the corresponding surface, and the bounding surfaces if there are any. Similarly, an edge id has the references of the feature namespace, the corresponding curve, and the bounding surfaces if there are any. There are some special geometry curves and surfaces that do not have intrinsic boundaries in B-Rep, such as circles and spheres. In these cases, extra boundary entities shall be introduced in order to identify topological entities. Localized features and surfaces can be named based on enumeration because of their relative stableness.

The semantic naming method takes a general and passive approach to identify curves, points, and topological entities, compared to enumeration that is a direct and active approach. This passive approach requires memory space to store the name tree and surface neighbor information for curves, points, faces, edges, and vertices when feature is evaluated. If feature is the final subspace as in Fig. 7, the depth of the name tree d is known and pre-determined. The maximum breadth of the tree b appears at the maximum number of components, features, or boundary surfaces of a face. The space requirement to store the name tree is $O(bd)$, and the time to resolve a name is $O(d)$. Each id contains extra construct information about surface-based boundary, which is a desirable property to increase stableness and retain topological semantics.

Fig. 9 shows how surface-based entity ids can provide stable references in the example of Fig. 2. As topological



(a) reference c before modification (b) reference c after modification



(a) reference d before modification (b) reference d after modification

Fig. 9. Surface-based entity ids provide stable references in the example of Fig. 2.

entities are volatile as parameters are modified, surface information persists within a session. Geometry persistently exists even though its corresponding topology is eliminated. The intersection of surfaces is guaranteed in complex space. Thus, surfaces can be ultimate references in feature construction.

Until now, we assume that two surfaces intersect at one curve. For polyhedrons, faces are corresponding to planes, which is the simplest case. If some faces are corresponding to quadratic or higher order surfaces. The assumption is not always true. Fig. 10 shows some examples of intersecting surfaces. For linear surfaces intersection (plane–plane), a line (as in Fig. 10(a)) is generated. For a linear surface intersecting with a quadratic surface, either one curve (as in Fig. 10(b–f)) or two curves (as in Fig. 10 (g–i)) will be generated. For higher-order surface intersections, one or two curves (as in Fig. 10(k–o)) will be generated. One exception is the special case that a plane intersects a cubic cylinder or even higher order at three or more parallel lines (as in Fig. 10(j)).

Uniqueness issue thus arises if two surfaces intersect at two or more curves, which should be resolved in a surface-based topological entity identification system. Further, even if only one intersection curve is generated, boundary surfaces may divide the curve into two or more edges. To identify curves and edges based on surfaces, extra information is needed if ambiguity exists. For parametric surfaces, curves can be identified based on the parameter ranges. Not all surfaces have parametric forms, whereas surfaces in parametric forms can be transformed to implicit forms. A general method is needed for surfaces in implicit forms.

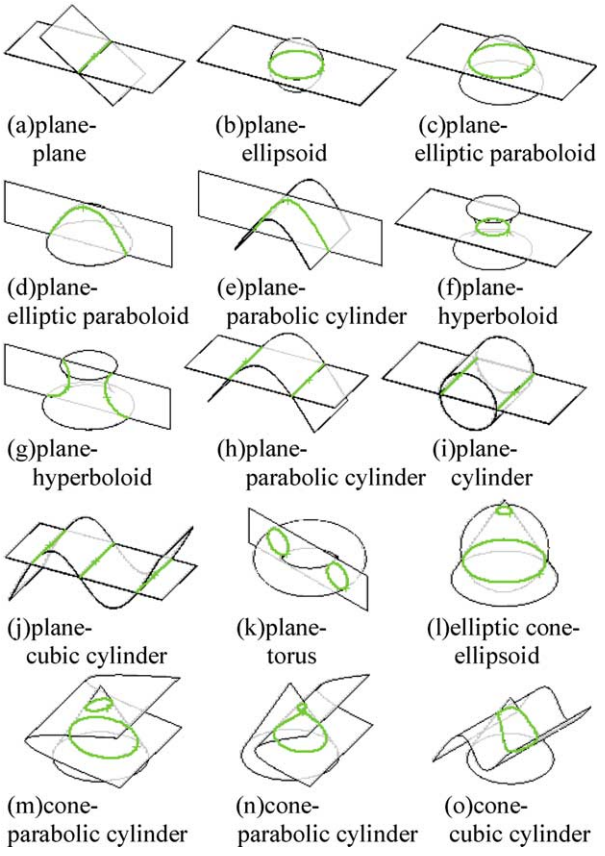


Fig. 10. Examples of multiple intersecting curves generated by two surfaces.

One consideration is to add orientation information of curves. If a k^{th} gradient operator ∇^k in Cartesian coordinates is

$$\nabla^k = \left[\frac{\partial^k}{\partial x^k} \quad \frac{\partial^k}{\partial y^k} \quad \frac{\partial^k}{\partial z^k}\right]^{\mathrm{T}} \quad (k>0), \tag{3}$$

a k^{th} gradient of the surface $f(x, y, z)=0$ at point $\mathbf{p}=(x, y, z)$ is

$$\mathbf{I}_\sigma^k(f,p) = \nabla^k f(x,y,z) \quad (k>0). \tag{4}$$

The orientation of the surface $f(x, y, z)=0$ at point $\mathbf{p}=(x, y, z)$ can be defined as the 1st gradient

$$\mathbf{I}_\sigma(f,\mathbf{p}) = \nabla f(x,y,z). \tag{5}$$

Let $f(x, y, z)=0$ and $g(x, y, z)=0$ be two surfaces intersecting at $c=\{(x, y, z)|f(x, y, z)=0, g(x, y, z)=0\}$. And the orientation of the curve c at point $\mathbf{p}=(x, y, z)$ is defined as

$$\mathbf{I}_{11}(f,g,\mathbf{p}) = \mathbf{I}_\sigma(f,\mathbf{p}) \times \mathbf{I}_\sigma(g,\mathbf{p}). \tag{6}$$

If the orientations of the intersecting curves at some interior points are included, edges can be identified. A simple way is to include the orientation information of bounding points of the curves. For example, in Fig. 10(g), plane $y=0$ intersects hyperboloid $x^2+y^2-z^2-1=0$, and two intersecting curves are bounded by planes $z+1=0$ and $z-1=0$. The orientations of two ending points are $[\pm 2, 0, 2\sqrt{2}]^{\mathrm{T}}$ for the left curve and $[\pm 2, 0, -2\sqrt{2}]^{\mathrm{T}}$ for the right curve, respectively, if the orientation is defined as the cross product of normal vectors for the plane and the hyperboloid. In Fig. 10(h), plane $z=0$ intersects with parabolic cylinder $x^2+z-1=0$, and two intersecting lines are bounded by planes $y+1=0$ and $y-1=0$. The orientations of two ending points are $[0,-2,0]^{\mathrm{T}}$ for the left line and $[0,2,0]^{\mathrm{T}}$ for the right line, respectively, if the orientation is defined as the cross product of normal vectors for the plane and the hyperboloid. Here, the sequence of the vector product is important in the definition of orientation. If the positions of f and g in (6) are switched, the orientation will have opposite direction. If the orientations of the curves at two ending points are the same, orientations at some other corresponding points on the curves should be derived to differentiate the two curves. If two surfaces are tangent at some points, the orientations of intersection curves at these points are zero vectors.

Extra care should be given to the special cases that three or more intersecting curves are formed and two curves have same orientation information, such as in Fig. 10(j). Plane $z=0$ intersects with cubic cylinder $x^3-x-z=0$. The orientation of the left and right line at any point is always

$[0,2,0]^T$, if the orientation is defined as the cross product of normal vectors for the plane and the cubic cylinder. In this case, additional information besides orientation is needed to identify the left and the right edge. One can include second-order gradients of surfaces or curves as the supplementary information of curve orientation for edge identification.

The adaptation of the surface $f(x, y, z)=0$ at point $\mathbf{p}=(x, y, z)$ is defined as second-order gradient

$$\mathbf{I}^2_\sigma(f, \mathbf{p}) = \nabla^2 f(x, y, z). \tag{7}$$

The adaptation of the intersection curve by surfaces f and g can be defined as

$$\mathbf{I}_{12}(f, g, \mathbf{p}) = \mathbf{I}_\sigma(f, \mathbf{p}) \times \mathbf{I}^2_\sigma(g, \mathbf{p}), \tag{8a}$$

$$\mathbf{I}_{21}(f, g, \mathbf{p}) = \mathbf{I}^2_\sigma(f, \mathbf{p}) \times \mathbf{I}_\sigma(g, \mathbf{p}), \tag{8b}$$

$$\mathbf{I}_{22}(f, g, \mathbf{p}) = \mathbf{I}^2_\sigma(f, \mathbf{p}) \times \mathbf{I}^2_\sigma(g, \mathbf{p}). \tag{8c}$$

When orientation of curve cannot differentiate the intersection curves, either adaptations of surfaces or curves need to be included. In the previous example of Fig. 10(j), the adaptation of the cubic cylinder is $[-6,0,0]^T$ at any point on the left intersecting line and is $[6,0,0]^T$ at any point on the right intersecting line. With the second-order gradients, these two curves can be identified even though curve orientations are equal.

If the adaptations of surfaces or curves still cannot differentiate the curves (e.g. in higher-degree surfaces), higher order gradients can be derived further to identify edges. This method can be extended beyond surfaces in implicit format. If some surfaces cannot be represented in closed form, they can be interpolated and approximated in polynomial forms, or in pragmatic sample data forms. The gradients and orientations can be approximated numerically, which makes this id format generally acceptable.

Similar to curve and edge identification, points or vertices are identified by the orientation/adaptation/gradient information of the intersecting curve of the first two surfaces at the particular positions if multiple curves or edges are generated by the same set of surfaces.

In summary, topological entities can be identified based on surfaces in evaluated solid geometry. Faces are named by the ids of corresponding surfaces with bounding surfaces. Edges are named by the ids of corresponding curves with bounding surfaces and extra orientation and gradient information of curves at boundary points if necessary, because it is possible that several edges are corresponding to one curve and same boundary surfaces. Curves are named by the ids of intersecting surfaces, as well as additional orientation and gradient information about the involved surfaces at some points (e.g. boundary points) if necessary, because it is possible that several curves are generated by intersecting surfaces. Vertices are named by the ids of corresponding points, which in turn are named by the ids of intersecting three or more surfaces with additional gradient information. The general syntax of topological and geometric entities' ids is shown in Fig. 11. The curve orientation and gradients for a curve name are derived based on the sequence of surfaces shown in its surface list in the first segment.

```
<feature_id> ::= <feature_type> ( <feature_name> ) |
                 <feature_type> ( <feature_name> ) :: <feature_step_name>
<face_id> ::= <face_type> ( <feature_id> :: <face_name> )
<edge_id> ::= <edge_type> ( <feature_id> :: <edge_name> )
<vertex_id> ::= <vertex_type> ( <feature_id> :: <vertex_name> )
<face_name> ::= <surface_id> | <surface_id> + <surface_list>
<edge_name> ::= <curve_id> | <curve_id>  + <surface_list> |
                <curve_id>  + <surface_list> - ( <additional_curve_info> )
<additional_curve_info> ::= <curve_orientation> & <curve_orientation> |
                            <curve_orientation> & <curve_orientation> -
                            <curve_adaptation> & <curve_adaptation>
<vertex_name> ::= <point_id>
<surface_list> ::= <surface_id> | <surface_id> & <surface_list>
<surface_id> ::= <surface_type> ( <surface_name> )
<curve_id> ::= <curve_type> ( <curve_name> )
<point_id> ::= <point_type> ( <point_name> )
<curve_name> ::= <surface_id> & <surface_list> |
                 <surface_id> & <surface_list> -
                 ( <additional_surface_info> )
<additional_surface_info> ::= <surface_orientation> & <surface_orientation> |
                              <surface_orientation> & <surface_orientation> -
                              <surface_adaptation> & <surface_adaptation>
<point_name> ::= <surface_id> & <surface_id> & <surface_list> |
                 <surface_id> & <surface_id> & <surface_id> & <surface_list> -
                 ( <additional_point_info> )
<additional_point_info> ::= <curve_orientation> |
                            <curve_orientation> - <curve_adaptation>
<feature_type> ::= <global_feature_type> | <local_feature_type>
<global_feature_type> ::= PROTRUSION | CUT | HOLE | LOFT | ...
<local_feature_type> ::= FILLET | CHAMFER | THREAD | ...
<face_type> ::= FACE
<edge_type> ::= EDGE
<vertex_type> ::= VERTEX
<surface_type> ::= PLANE | QUADRATIC_SURFACE |
                   CUBIC_SURFACE | QUARTIC_SURFACE |
                   FREE_FORM_SURFACE
<curve_type> ::= LINE | QUADRATIC_CURVE | CUBIC_CURVE |
                 QUARTIC_CURVE | FREE_FORM_CURVE
<point_type> ::= POINT
```

Fig. 11. General syntax of semantic id.

5. Naming service

Semantic ids need to be assigned and connected to system-dependent identifiers or physical addresses. This process is executed by a naming server, which can be independent of modeling systems. The naming services include binding (assign semantic ids to physical addresses in internal representation), resolution (look up physical addresses from semantic ids), and update (update semantic ids after model re-evaluation). The structure of the naming server is shown in Fig. 12. Through an interface between the naming server and modeling system, names can be resolved locally or remotely. Semantic ID scheme provides a scalable way to name entities. An independent naming server allows

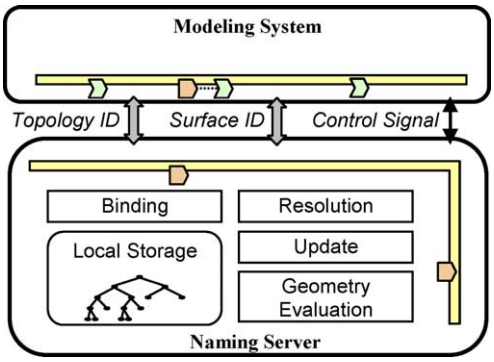


Fig. 12. Structure of naming server.

for binding either during design process or after a design is finished. Within the server, name hierarchy can be resolved either iteratively or recursively.

During model re-evaluation, surface-based semantic ids for re-evaluated boundary representation need to be updated because of the change of surface geometry. We simply call the entity id in the old solid before modification old id, and the id of its counterpart in the new solid after modification new id. This update requires mappings between old and new surface ids for reference consistency as well as mappings between other entities if ambiguity exists for multiple intersecting curves.

5.1. Surface id update

For each re-evaluation, new surfaces should be named referring to corresponding old surfaces. Naming server needs to record surfaces that have been created and modified. When a new surface is created, a global unique number (e.g. IP address||local time||random number) can be assigned to it. If a surface is changed due to the modification of its own *r*-parameters, it can be immediately determined that the resulting surface is the original one. No id update is necessary. Otherwise, surface id update can take a pointer forwarding approach. If the new surface is generated as a result of parameter change in one feature definition, such as dimensions of protrusion profile, protrusion distance, and radius of chamfer, the new surface id should be linked to the corresponding old one. Old surface ids should be kept for future reference until the particular design session ends. A flag needs to be attached to the most recent one. In most modeling systems, surface identification within one session is persistent. Thus, the above general surface mapping process usually is not required.

5.2. Curve, edge, and point mapping

The first segment (surface segment) of ids for curves, edges, and points is rather stable and independent of topological faces. Even if a face is eliminated from a solid, the geometry of a surface still exists in Euclidean space. The second segment (orientation/adaptation/gradient segment), which contains vector values, may be changed each time when geometry is altered. That is, orientations, adaptations, and higher-order gradients of curves and surfaces at edges' boundary points and inner points may be changed if the geometry of some surfaces is modified.

The mapping here is based on geometric properties instead of topological correspondence. The surface segment of the new id is the same as that of the old one, which reduces the complexity of mapping. If only one curve is generated at an intersection, or no additional geometric information (orientation/adaptation/gradient) is included in either of the old and new ids, there is an exact match for ids. If two or more curves are generated at the intersection, and additional surface information is included in both old and new ids, the mapping is based on closeness of curves.

Suppose c_1 is the intersection curve of surfaces f_1 and g_1, and c_2 is the intersection curve of surfaces f_2 and g_2. Points $\mathbf{p}_1$ and $\mathbf{p}_2$ are on curves c_1 and c_2, respectively. The k-closeness of curve c_1 and c_2 at $\mathbf{p}_1$ and $\mathbf{p}_2$, k-close(f_1, g_1, f_2, g_2, $\boldsymbol{p}_1$, $\boldsymbol{p}_2$), can be defined as

$$k\text{–close}(f_1,g_1,f_2,g_2,\mathbf{p}_1,\mathbf{p}_2)=|\mathbf{p}_1-\mathbf{p}_2| + \sum_{i=1}^{k}\sum_{j=1}^{k}|\mathbf{I}_{ij}(f_1,g_1,\mathbf{p}_1)-\mathbf{I}_{ij}(f_2,g_2,\mathbf{p}_2)| \quad (k\geq 0) \tag{9}$$

0-closeness of curve c_1 and c_2 at $\mathbf{p}_1$ and $\mathbf{p}_2$ is the distance between $\mathbf{p}_1$ and $\mathbf{p}_2$.

Curve mapping can be done based on the values of k-closeness. If m curves (c_1, c_2, …, c_m) ($m>1$) are generated by the intersection of surfaces f_1 and g_1 in the new solid, and n curves (d_1, d_2, …, d_n) ($n>1$) were generated by the corresponding surfaces f_2 and g_2 in the old solid, there is a point $\mathbf{p}_i$ selected on each of the c_i ($i=1, 2, \ldots, m$) and a point $\mathbf{q}_j$ selected on each of the d_j ($j=1, 2, \ldots, n$), where $\mathbf{p}_i$ and $\mathbf{q}_j$ are the intersecting points between the curves and a plane $x=a$ (or $y=b$, or $z=c$). For each pair of c_i and d_j, k-close(f_1, g_1, f_2, g_2, $\mathbf{p}_i$, $\mathbf{q}_j$) is calculated. If only orientation is included in curve ids, $k=1$. If adaptation information is included in curve ids, $k=2$. Generally, k is the highest order of surface gradient in the curve ids. Then an $m\times n$ closeness matrix $\mathbf{R}$ is generated by listing each of the new curves as row indices and each of the old curves as column indices. In each row r_i of $\mathbf{R}$, the elements r_{ij} is the rank of closeness based on k-close(f_1, g_1, f_2, g_2, $\mathbf{p}_i$, $\mathbf{q}_j$) for $j=1, 2, \ldots, n$. The smallest k-closeness is ranked as 1, and the largest k-closeness is ranked as n. If a tie appears in the k-closeness matrix, $(k+1)$-closeness ($k>0$) of the curves is calculated and inserted into the closeness matrix.

Once the closeness matrix is built, the mapping of curves can be done by selecting the lowest rank values. Each new curve will be mapped to its corresponding old curve with rank 1. In special cases, it is possible that one new curve is mapped to multiple old curves when a curve is split into multiple curves (i.e. an old curve has the lowest rank value in multiple rows). For example, plane $z=0$ intersects with

cubic cylinder $x^3-x-z=0$ (as in Fig. 10(j)) and three curves are generated. If the plane is changed to $z=0.25$, three new curves need to be mapped to old curves. The 2-closeness matrix is calculated at the intersection points with plane $y=0$. According to the matrix values

$$0\text{−close} = \begin{bmatrix} 0.2981 & 0.772 & 2.1219 \\ 0.8741 & 0.3677 & 1.135 \\ 1.8545 & 1.294 & 0.272 \end{bmatrix},$$

$$1\text{−close} = \begin{bmatrix} 1.1936 & 3.554 & 2.7993 \\ 2.9786 & 0.5857 & 4.8124 \\ 2.75 & 4.0759 & 0.9494 \end{bmatrix},$$

$$2\text{−close} = \begin{bmatrix} 2.1682 & 7.9364 & 15.442 \\ 8.004 & 2.2033 & 11.455 \\ 13.775 & 11.694 & 1.5924 \end{bmatrix},$$

curves can be identified.

After a curve is identified, it is possible that multiple edges are generated by bounding the curve by the same set of boundary surfaces. Then edge mapping is needed based on k-closeness of the curve at boundary points to identify the corresponding edges between new and old solids. Suppose m edges ($m>1$) are generated by the same set of boundary surfaces with the same intersection curve of surfaces f_1 and g_1 after re-evaluation. Each edges a_i was bounded by starting point $\mathbf{p}_{is}$ and ending point $\mathbf{p}_{ie}$ ($i=1, 2, \ldots, m$). Before re-evaluation, n edges ($n>1$) are created by corresponding surfaces f_2 and g_2. And each edges b_j was bounded by starting point $\mathbf{q}_{js}$ and ending point $\mathbf{q}_{je}$ ($j=1, 2, \ldots, n$). For each pair of edges a_i and b_j, $k\text{-close}_{\text{edge}}(a_i, b_j)$, can be calculated as

$$\begin{aligned} k\text{−close}_{\text{edge}}(a_i, b_j) = {} & k\text{−close}(f_1, g_1, f_2, g_2, p_{is}, q_{js}) \\ & + k\text{−close}(f_1, g_1, f_2, g_2, p_{ie}, q_{je}) \end{aligned} \quad (10)$$

Similar to the closeness matrix for curves, a closeness matrix for edges can be derived with each element as $k\text{-close}_{\text{edge}}(a_i, b_j)$. The mapping of edges can be conducted based the ranks of closeness matrix. And the mapping of points is based on the closeness of curves.

It is noted that not every entity in each model re-evaluation has mapping involved. Only surfaces with multiple intersection curves or edges should include the mapping process. The complexity of mapping is a function of k, which is the highest order of surface gradient in the curve ids. k can be estimated based on the number of intersections or the order of surface equations, which is usually less than 4. The time complexity to calculate matrices is $O(k^2)$, and the space requirement to store matrices is $O(k^2)$.

5.3. Implementation

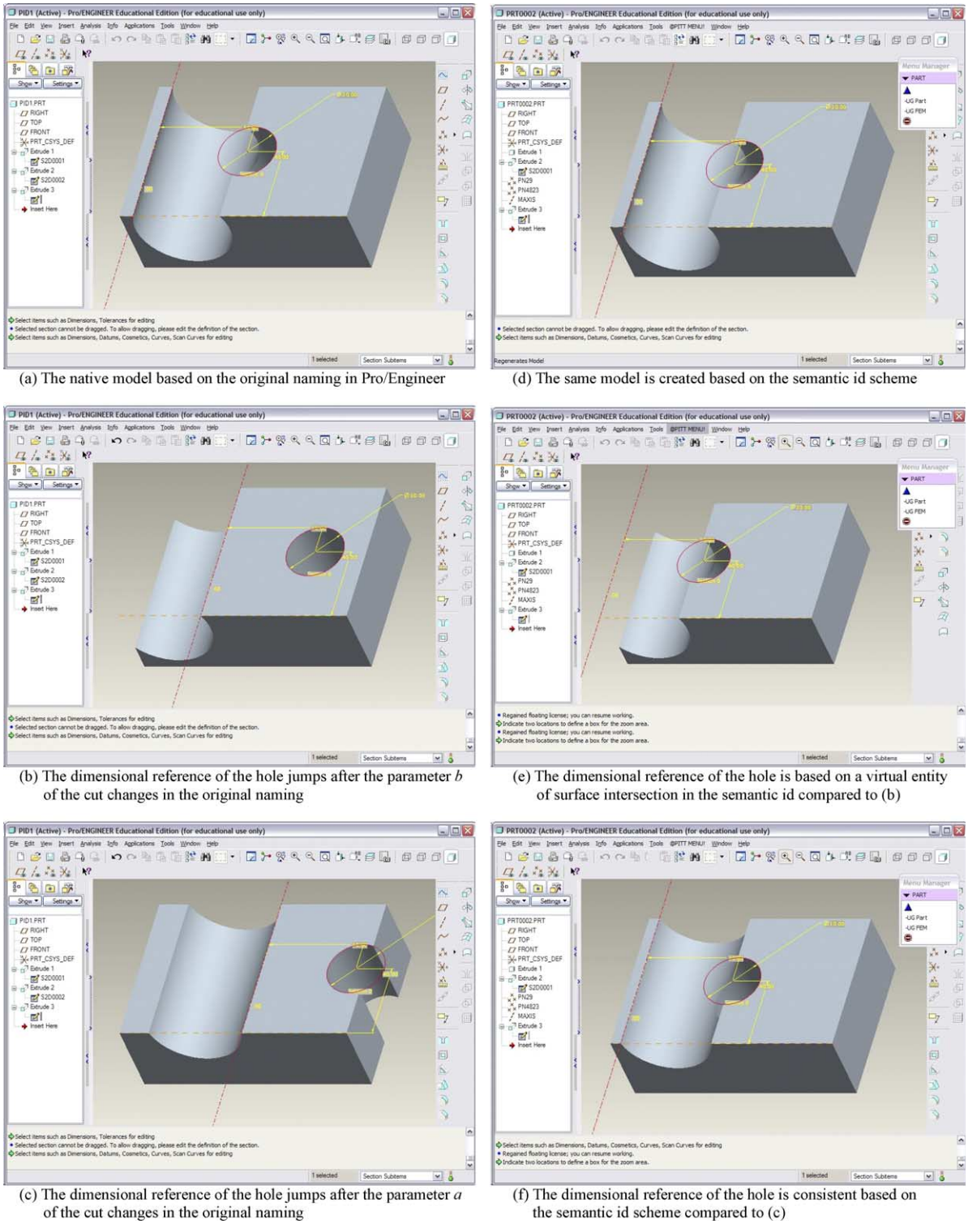
The semantic naming scheme is developed and integrated into Pro/Engineer® Wildfire 2.0 based on Pro/Toolkit®. In most real-world design cases, only intersections within real Euclidean subspace need to be considered, and the number of intersection curves for two surfaces is normally up to two. The geometry-based scheme can take advantage of original geometry evaluation functions existing in geometric modelers. These factors reduce the complexity of implementation. Surface geometry evaluation functions in Pro/Engineer for both parameters and coordinates are used to find the intersection during name resolution. Surface ids are queried through Pro/Toolkit application programming interfaces. Surface id update now depends on Pro/Engineer's native function.

To improve the performance of naming server such that Pro/Engineer, as client, does not have to query the names of curves or points starting from surfaces each time, a caching mechanism is introduced. Pro/Engineer creates and stores virtual entities locally such as datum points and datum curves corresponding to surface intersection. Virtual entities then become the indirect references used in models. During each model re-evaluation, virtual entities are updated based on the new intersection positions of surfaces.

Fig. 13 illustrates how the semantic id scheme can easily resolve the naming issue in the example of Fig. 1. (a) is the native model created based on the original naming method in Pro/Engineer. (b) and (c) show how the relation parameter changes in the cut feature affect the dimensional references in the hole feature defined afterwards. In contrast, (c) is the model created based on the semantic reference scheme. The parameter changes do not affect the definition of the hole, as shown in (d) and (e). Fig. 14 shows the calculated orientations of the two intersecting surfaces (the plane and the cylinder) at boundary points before parameter change as in Fig. 13(d) and after parameter change as in Fig. 13(e). The two intersection curves and edges can be differentiated based on the orientations in their semantic ids.

6. Conclusion

In this paper, a sufficient condition of B-Rep variance based on geometry continuity in parametric complex Euclidean ($\mathbf{P}^p\mathbf{C}^3$) space is proposed. Shape and relation parameters are differentiated in $\mathbf{P}^p\mathbf{C}^3$, thus parametric family can be defined. A semantic id scheme based on continuity of geometry is developed such that entities are named based on persistent geometry to solve the problem of topology inconsistency in parametric modeling, which aims to improve the robustness and interoperability of CAD feature modeling. In this surface-based scheme, prefixing ids with feature namespaces transform the original flat namespace to an organized logical naming hierarchy.



(a) The native model based on the original naming in Pro/Engineer

(d) The same model is created based on the semantic id scheme

(b) The dimensional reference of the hole jumps after the parameter *b* of the cut changes in the original naming

(e) The dimensional reference of the hole is based on a virtual entity of surface intersection in the semantic id compared to (b)

(c) The dimensional reference of the hole jumps after the parameter *a* of the cut changes in the original naming

(f) The dimensional reference of the hole is consistent based on the semantic id scheme compared to (c)

Fig. 13. Model in Fig. 1 created based on the semantic id scheme is stable in comparison with the one generated based on the native naming method in Pro/Engineer.

The ids identify themselves descriptively by the procedure of feature operations. The inclusion of geometric information and boundary association in topological ids lets a topological id possess geometric and topological semantics. The geometric ids possess construct relations of surfaces, curves, and points. Additional geometry information (orientation and gradient) is included in ids when multiple curves are formed by intersecting the same set of surfaces,

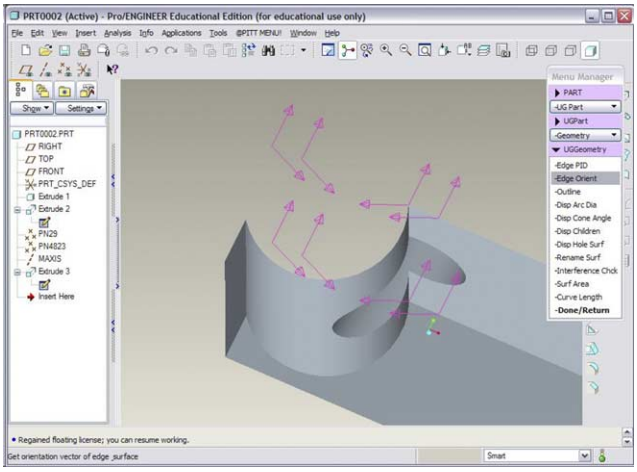


Fig. 14. Comparison of orientations of intersecting surfaces and generated curves at boundary points before modification as in Fig. 13(d) and after modification as in Fig. 13(e).

or multiple edges are created by same curve and boundary surfaces. This semantic id scheme provides a stable and efficient reference structure.

Acknowledgements

This work was supported in part by the Office of Naval Research, US Department of Defense (Grant number: N00014-02-1-0649).

References

[1] Bidarra R, Bronsvoort WF. Semantic feature modeling. Comput Aided Des 2000;32(3):201–25.
[2] Shapiro, V., Vossler DL. What is a parametric family of solids. Proceedings of the third ACM symposium on solid modeling and applications, May 17–19. Utah: Salt Lake City; 1995. p. 43–54.
[3] Chen X, Hoffmann CM. On editability of feature-based design. Comput Aided Des 1995;27(12):905–14.
[4] Capoyleas V, Chen X, Hoffmann CM. Generic naming in generative constraint-based design. Comput Aided Des 1996;28(1):17–26.
[5] Kripac, J. A Mechanism for persistently naming topological entities in history-based parametric solid models (Topological ID System). Proceedings of the third ACM symposium on solid modeling and applications, May17–19. Utah: Salt Lake City; 1995. p. 21–30.
[6] Kripac J. A mechanism for persistently naming topological entities in history-based parametric solid models. Comput Aided Des 1997; 29(2):113–22.
[7] Marcheix D, Pierra GA. Survey of the persistent naming problem. Proceedings of the seventh ACM symposium on solid modeling and applications, June 17–21; 2002. p. 13–22.
[8] Wu J, Zhang T, Zhang X, Zhou J. A face based mechanism for naming. Recording and retrieving topological entities. Comput Aided Des 2001;33(10):687–98.
[9] Stewart NF. Sufficient condition for correct topological form in tolerance specification. Comput Aided Des 1993;25(1):39–48.
[10] Raghothama S, Shapiro V. Necessary conditions for boundary representation variance. Proceedings of the thirteenth ACM annual symposium on computational geometry, June 4–6. France: Nice; 1997. p.77–86.
[11] Raghothama S, Shapiro V. Topological framework for part families. Proceedings of the seventh ACM symposium on solid modeling and applications, June 17–21. Germany: SaarbrŰcken; 2002. p.1–12.
[12] Lachlan R. An elementary treatise on modern pure geometry. London: Macmillan; 1893 [Chapter 1, p. 4–5].
[13] Bell ET. The development of mathematics. 2nd ed. New York: McGraw-Hill; 1945 [Chapter 15, p. 340].

Yan Wang is a Research Assistant Professor at the Department of Industrial Engineering, University of Pittsburgh. He received his BS in Electrical Engineering from Tsinghua University in Beijing, MS in Electrical Engineering from Chinese Academy of Science, and PhD in Industrial Engineering from University of Pittsburgh. He is a principle investigator at the US National Science Foundation Industry/University Cooperative Research Center for e-Design at the University of Pittsburgh. His research interests include geometric modeling and reasoning, constraint-driven systems, interoperable and secure information sharing in network-centric computer-aided design.

Bart O. Nnaji has a BS in Physics from St John's University with distinction; an MS and PhD in Industrial and Systems Engineering from Virginia Tech, and obtained a certificate of Postdoctoral studies in Artificial Intelligence and Robotics from MIT. He was Professor of Department of Mechanical and Industrial Engineering at the University of Massachusetts at Amherst till 1996. He is currently the Director of the National Science Foundation Center for e-Design and the William Kepler Whiteford Professor on Engineering at the University of Pittsburgh. Professor Nnaji has received 3 honorary doctorates from international universities. He received 1988 Outstanding Young Manufacturing Engineer Award by the Society of Manufacturing Engineering; 1992 Outstanding Young Industrial Engineer Award; He is a Fellow of Nigerian Academy of Sciences; Fellow of the Institute of Industrial Engineers, and Fellow of the Society of Manufacturing Engineers. He was honored with the US Secretary of State's Distinguished Public Service Award in 1995; *Distinguished Scientist Award* by the World Bank-IMF in 1998; and the Nigerian President national honor-*Officer of the Order of Niger (OON)* in 2000. Professor Nnaji has served as principal or co-principal investigator on over $40 million research. He has published 5 books and over 100 technical articles. One of his books, *Computer Integrated Manufacturing and Engineering,* won the 1994 world best text book prize for Manufacturing Engineering. Professor Nnaji is the founding Editor-in-Chief of the *International Journal of Design and Manufacturing* and also served as the Editor for the Design Department of *Institute of Industrial Engineers Transactions on Design and Manufacturing*. Professor Nnaji served as Nigeria's Federal Minister of Science and Technology in 1993.