Consistent updates in dual representation systems *

Srinivas Raghothama Vadim Shapiro
Spatial Automation Laboratory †
University of Wisconsin-Madison

Abstract

Dual representation systems rely on a parametric model to create and to manipulate the boundary representation of a solid. Parametric and boundary representations remain consistent if they continue to model the same solid, after updates applied to the parametric model and/or to the boundary representation. We consider the problem of verifying the consistency between the boundary representation and the CSG representation which is a restricted case of parametric representation. We determine the necessary and sufficient conditions for consistency of the two representations, identify the required computational utilities, and describe two implemented and tested algorithms.

Our algorithms rely on sign-invariant space decompositions and can be used to verify consistency of other representations, including parametric definitions such as feature attachments. They could also be used to verify consistency of boundary and parametric representations in different systems, or corresponding representations in integrated modeling environments.

1 Introduction

1.1 Multiple representations

Most modern solid modeling systems use dual representation architecture in order to take advantage of the well known complementary strengths of parametric and boundary representations [12]. In the classical dual representation systems the solids were typically constructed using Constructive Solid Geometry (CSG) representation; they were concise, high-level, parameterized, and robust, but had no explicit boundary data. The system then would compute the boundary representation (b-rep) to be used by applications that require boundary traversals; but direct manipulation of b-reps remained limited because they are difficult to parameterize and are prone to robustness problems. This architecture of CSG-Brep dual representation systems has evolved into the modern parametric feature-based representation and boundary representation [22]. The fundamental distinction of the new systems is that CSG operations have been extended to include more general "feature-attaching" operations that can be characterized as "local" CSG operations, i.e., Boolean operations whose result is further modified by local references to existing geometry [6]. The advantages of dual representations are clear: each representation's individual strengths are utilized. But these advantages come at a price: the system now must ensure that the two representations remain consistent at all times. In other words, both representations should always correspond to the same valid solid.

Multiple representations of the same solids also arise in integrated modeling environments [1, 8] and as the result of data transfer: different systems tend to employ distinct and often incompatible representations of parametric and/or boundary model. Verifying the consistency of models in different systems is usually difficult and often appears impossible. In this paper we focus on consistency of representations within the same system, but we explain in section 5.1 how our methods are applicable to the more general problem involving multiple representations in different systems.

1.2 Consistency via representation conversion

The traditional wisdom has been that the problem of consistency can be solved through bilateral representation conversions [12]. Thus, every change in a parametric model is accompanied by a boundary evaluation process that updates

*To appear in Journal of C.A.D

†Complete address: 1513 University Avenue, Madison, WI 53706, USA. E-mails: vshapiro@engr.wisc.edu (corresponding author), raghotha@sal-cnc.me.wisc.edu

the corresponding b-rep (refer to Figure 1(a)). This approach is inefficient when the parametric model is CSG, though local and incremental algorithms for boundary evaluation have been developed to deal with the problem [26, 16]. But if the parametric model makes references to existing boundary representation, the boundary re-evaluation process also becomes unreliable [22], and is further complicated by the necessity to correctly match the (a priori unknown) portions of boundary representations before and after the parametric change. This problem is known as "persistent naming" and is currently handled using heuristic methods [5, 9]. The principle of continuity proposed by the authors in [24] alleviates the problem to a large extent, but does not solve it completely.

The situation is even more complicated if a boundary representation is edited directly, either by Euler operators [10] or by "tweaks" that are local modifications of the boundary representation. Direct modification of boundary representations has been an important and popular paradigm [3], that is supported by most modeling kernels. However the formal semantics of such operations are just beginning to emerge [17, 24]. Direct modifications of b-reps are usually not allowed in dual representations systems, because converting a boundary representation to a parametric representation is difficult or infeasible in some cases. The problem of boundary to CSG conversion is now well understood and has been largely solved for a useful class of three-dimensional solids [19, 20, 21], but the solution does not easily generalize to other parametric definitions and solids bounded by free-form surfaces. Furthermore, every representation *conversion* is likely to produce a *new* parametric definition, thus destroying the history and the "intent" of the original parametric construction.

1.3 Update and test

We consider an alternative approach to maintaining consistency between representations: apply the implied modifications directly to both representations simultaneously and then *verify* that the two representations remain consistent. The approach is diagrammed in Figure 1(b) and requires solving two technical problems:

- Defining an equivalent semantics for every edit in *both* representations. For example, transforming (moving, deforming, etc) a face in a b-rep usually (but not always) corresponds to the same transformation of the halfspace in the parametric representation, and vice versa.
- Given the modified parametric and boundary representations, determine whether or not they are consistent.

In this paper we focus on the second task of verifying that the two representations are consistent. Our solution relies on the established representation conversion technology [18], but is easier because no new representations are actually constructed. And in particular, consistency verification does not require solutions to difficult problems associated with separation and free-form surfaces that prevent the wider use of b-rep to CSG conversion [21]. In the context of parametric modeling, our approach allows more efficient incremental and bilateral updates that avoid the problem of "persistent naming".

The main limitation of this approach is in the restricted range of modifications that may be administered simultaneously to both representations. Clearly, parametric changes may lead to significant topological changes in b-rep, and small changes in b-rep may result in inconsistency between the two representations. The first of these difficulties appears to be resolvable [17]; we briefly discuss the latter problem in the concluding section of this paper.

1.4 Focus and outline

We will assume that the representation $F(\mathbf{P})$ is parameterized in terms of the parameters of its halfspaces [1] from a set $\mathbf{P}$. Since all faces in a b-rep are subsets of boundaries of the same set of halfspaces [14], it is reasonable to assume that there is an explicit link between the two. Thus any transformation of parametric halfspaces should be accompanied by the same transformation applied to the corresponding face(s). For the purposes of this paper, we consider only rigid body motions of halfspaces and faces. If a parametrized halfspace is shared by more than one sub-expression (node) of $F(\mathbf{P})$, then transformation of this one halfspace will affect all of them. Similarly a single halfspace may contribute to more than one face in a b-rep, in which case the transformation is applied to all such faces.

Transformation of a face in a b-rep is carried out using the tweak function available in most solid modelers and should result in a valid b-rep L. Assuming that the transformation is continuous, the tweak must satisfy the adjacency

[1] We use the term 'halfspace' in a broad sense: any representation of a closed regular semi-algebraic set with a well defined point membership classification qualifies.

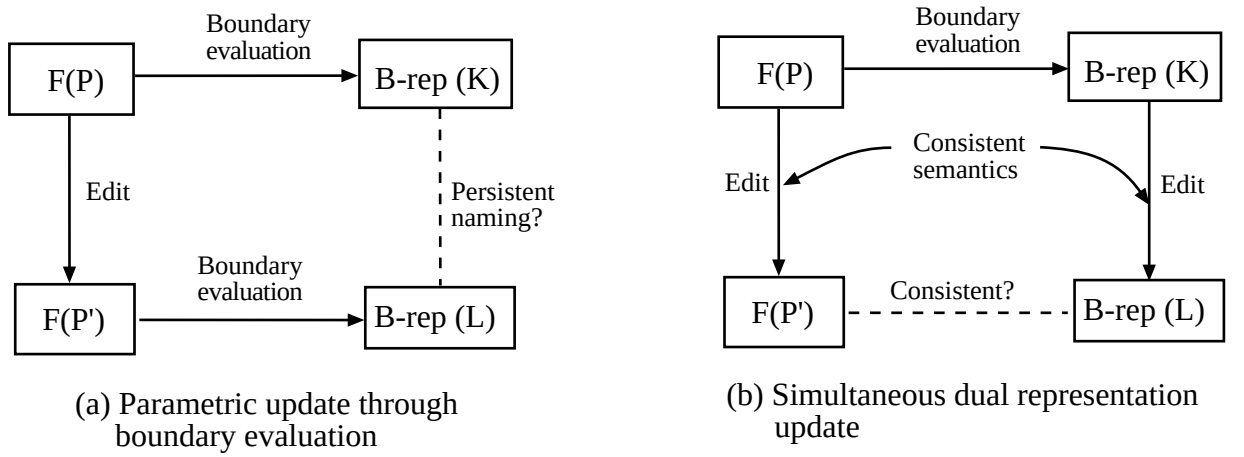


Figure 1: Two approaches to parametric update

and orientation conditions formally derived in [24]; large tweaks may require additional information [17]. In order to verify consistency between the new b-rep L and the updated parametric definition $F(\mathbf{P}')$, we need to know the exact semantics of the parametric definition as well as the b-rep. The semantics of b-reps are well understood [12, 23], but the semantics of modern parametric representations has not been fully developed. Thus, we will first focus on the special case when a parametric representation is CSG, and then discuss how the techniques may be extended to more general parametric definitions. Two-dimensional examples in the paper are used only for illustration purposes; all algorithms have been implemented for three-dimensional solids using Parasolid kernel modeling system.

The organization of the rest of the paper is as follows. In section 2 we will summarize the properties of b-rep and CSG representations in terms of space decompositions. Space decompositions are the main tool for formulating and solving all representation conversion problems [18], because they systematically reduce all such problems to a finite number of point membership classification tests.

After reviewing the general process of representation conversion, we turn to the problem of consistency verification in Section 3. The problem of verifying consistency between a CSG and a b-rep is similar to the classical problem of same object detection (SOD) arising in many applications [26], except that in our case the two objects are given by *different* representations. This view of the problem leads to formulation of four distinct consistency verification tests. We consider the computational requirements for each test, and in Section 4 study in detail one of them. We discuss specific implementations of brute-force and incremental algorithms and compare the running times for simple three-dimensional solids.

We conclude in section 5, by explaining how our approach to consistency verification can be applied to other representations (possibly in different systems), and discuss briefly the possible remedies when the two representations are found to be inconsistent.

2 Representation Conversion Technology

This section briefly summarizes the properties of b-rep and CSG representations in terms of cell decompositions. These decompositions reduce all representation conversion problems to a finite number of point membership classification tests.

2.1 Solid representations as unions of cells

Boundary representations are a restricted form of cell complexes and their validity as well as other formal properties are rooted in algebraic topology. The cells σ_i of a boundary representation K satisfy the following properties:

- Every k-dimensional cell (k-cell) is relatively open with respect to a k-manifold; in particular every two-dimensional cell (also called as a face) is a subset of the boundary of some halfspace $h_i(x, y, z) \geq 0$;
- The closure of every cell is a union of some other cells in K;
- Dimension$(\sigma_i) \leq 2$ for boundary representations of three-dimensional solids.

The underlying space of a b-rep K is denoted by $|K|$ and is the set of points ∂S. The solid S is the union of some bounded connected components of the complement of $|K|$. The boundary of every solid is a unique set of points, but there are many distinct boundary representations for the same solid. However, every b-rep can be put into a unique canonical form by further partitioning every cell of K into subsets of points that have the same classification (in, on, or out) with respect to every halfspace h_i induced from b-rep. Such subsets are called *sign-invariant* with respect to the set of polynomials $\mathbf{P} = \{h_i\}$.

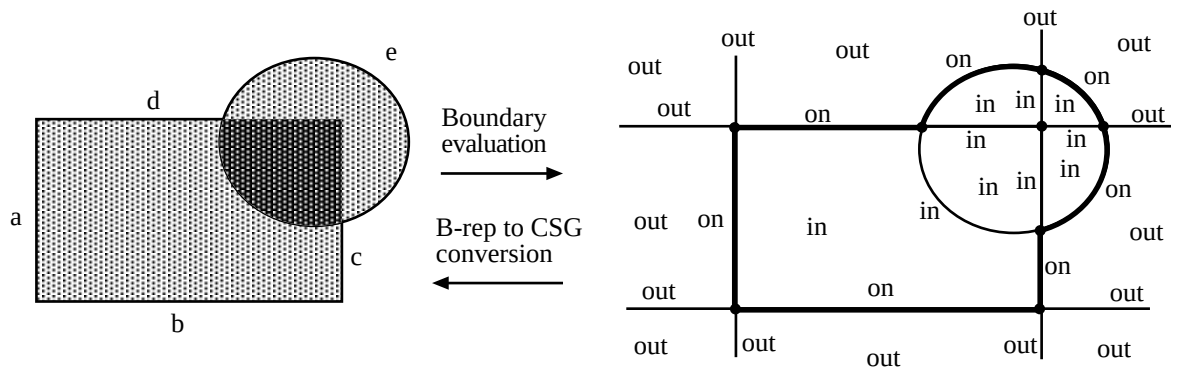
CSG representations are expressions $F(\mathbf{P})$ constructed from regularized Boolean operations on a set $\mathbf{P}$ of parametrized halfspaces. The solid represented by $F(\mathbf{P})$ is a set of points denoted by $|F(\mathbf{P})|$. Though CSG representations are not unique, every CSG representation can be written in a unique disjunctive canonical form as a union of canonical intersection terms $\Pi_i = \{x_1 \cap^* x_2 \cap^* \ldots \cap^* x_n\}$ with $x_j \in \{h_j, \overline{h_j}\}$ for all halfspaces $h_j \in \mathbf{P}$ [19]. The collection of all non-empty sets $|\Pi_i|$ constitute a decomposition of space into sets with disjoint and possibly disconnected interiors. By definition, all points in interior of each $|\Pi_i|$ have identical classification with respect to all halfspaces h_j in $\mathbf{P}$, i.e. each intersection term represents a three dimensional set of points that is sign-invariant with respect to $\mathbf{P}$.

2.2 Representation conversions

It is shown in [18] that *all* geometric representation conversions may be carried out as a sequence of five steps. Suppose we are given a representation of a set S (which may or may not be solid) in some representation scheme called X-rep, and we seek a representation of S in a scheme called Y-rep (X-rep and Y-rep may be the same scheme). The X-rep $\rightarrow$ Y-rep conversion could be approached in the same five steps:

1. From every primitive in the given X-rep of a set S, we induce a polynomial h_i.
2. If the induced set of polynomials $\mathbf{P} = \{h_i\}$ is not sufficient, additional 'separating' polynomials may be required.
3. A decomposition of space into cells by all present polynomials is then computed; the type of cells is determined by the properties of Y-rep.
4. Every cell (represented by a single representative point) in the decomposition of space is classified against the given representation of S in X-rep. Those cells passing the test give rise to a canonical representation of S in Y-rep.
5. Optimization of the obtained representation in Y-rep concludes the conversion.

Consider boundary evaluation as a representation conversion process for the two-dimensional solid shown in figure 2. The rectangular block is the intersection of four linear halfspaces, and it is easy to write a CSG expression defining the resulting solid. From every primitive in the given CSG definition, we obtain the primitive's boundary. In the example of Figure 2(a), the boundaries are 4 lines and one ellipse (step 1). Step 2 is trivial, because $\partial S \subseteq (\partial h_1 \cup \ldots \cup \partial h_n)$ [13]. Thus cells in b-rep can constructed from pieces of primitive boundaries, assuming that every such cell can be somehow individually represented. We trim all primitive boundaries against each other to obtain a collection of tentative cells; these are the linear and elliptic segments as well as the intersection points in Figure 2(b). All points of a tentative cell have the same classification with respect to every primitive in CSG and with respect to the defined solid S: IN, ON or OUT. The boundary ∂S consists of all those cells that are ON S. To classify a given tentative cell with respect to CSG representation of S, its representative point must be first classified with respect to each primitive, and the results combined according to the Boolean definitions and certain neighbourhood computations [26]. Finally, the adjacent cells in ∂S can be merged into larger cells leading to a more compact b-rep. In our example, the three



(a) CSG is the union of IN cells (b) B-rep is the union of ON cells

Figure 2: Representation conversion between a CSG and a b-rep

adjacent elliptic ON segments together with the points between them in Figure 2(b) could be combined in a single elliptic edge. The first systematic study of boundary evaluation was done by [14]. More efficient algorithms rely on computing the b-rep heirarchically and incrementally [16, 26].

Now consider the reverse process of b-rep to CSG conversion, following similar sequence of steps (refer to Figure 2). From every 2-cell (face) in the b-rep of a solid (Figure 2(b)) induce a halfspace. The five halfspaces induced in figure 2(b) are four linear halfplanes and the elliptical disk. It may turn out that the halfspaces induced from a b-rep of S are not sufficient to represent S with a CSG and additional *separating* halfspaces must be constructed (see [21] for examples). We decompose the d-dimensional space (in this case the plane E^2) using the boundaries of all halfspaces into a collection of tentative d-dimensional cells (bounded by straight and elliptic segments). All points in every cell have the same classification with respect to every halfspace induced from b-rep and with respect to the solid S. Because all cells are also d-dimensional, they can classify in only two distinct ways: IN or OUT. A CSG representation can be obtained by taking a union of CSG representations of the individual IN-cells. To classify a given cell, it is sufficient to pick any point inside the cell and perform a PMC test against the b-rep of S. The produced CSG is verbose and inefficient, but near minimal (though non-unique) CSG representations can be constructed using Boolean optimization technique [19].

2.3 The common cell decomposition

Rather than identifying the required space decomposition in a case by case fashion, we will assume that given a set of primitives $\mathbf{P}$, a single common partition of space $\{C_i\}$ can be computed satisfying the following requirements:

1. each cell C_i is *connected* submanifold of dimension $p = 0, 1, 2, 3$ and is *sign-invariant* with respect to all n halfspaces in $\mathbf{P}$;

2. the closure of every cell C_i is the union of some other cells in the partition; this implies that the sufficiently small neighborhoods of any two points in a cell intersect the same cell in the decomposition. [2]

The smallest partition satisfying the above requirements is sometimes called Whitney regular sign-invariant connected *stratification*, and individual cells are called *strata* [18]. Figure 2(b) shows the required sign-invariant partition for the solid in Figure 2(a), with all the 2- and 1-cells labeled according to their classification with respect to the solid. Each cell in the partition is uniquely represented by a single representative point from that cell. Once such a set of points is computed, all representation conversions, comparisons, and optimizations reduce to a finite number of PMC tests and logical decisions [18]. We shall see in the next section that the cells in the common partition are the key to formulating and maintaining consistency between CSG and b-rep. In practice, we may choose to represent only some cells in the decomposition, or to compute the decomposition hierarchically or incrementally.

[2] This condition is often stated as a *constant neighborhood* requirement [14].

3 Formulation

3.1 Consistency defined

The general notion of consistency between any two representations is defined in [12].

Definition 1 (Consistency of Representations) *Let X-rep and Y-rep be two representation schemes, and M denote the modeling space. Representations x in X-rep and y in Y-rep are consistent if there is an object m of M having representations x and y.*

In other words, two representations x and y are consistent with each other if both represent the same pointset. But this definition does not say anything about how consistency may be verified for a particular choice of representations (in our case, CSG and b-rep).

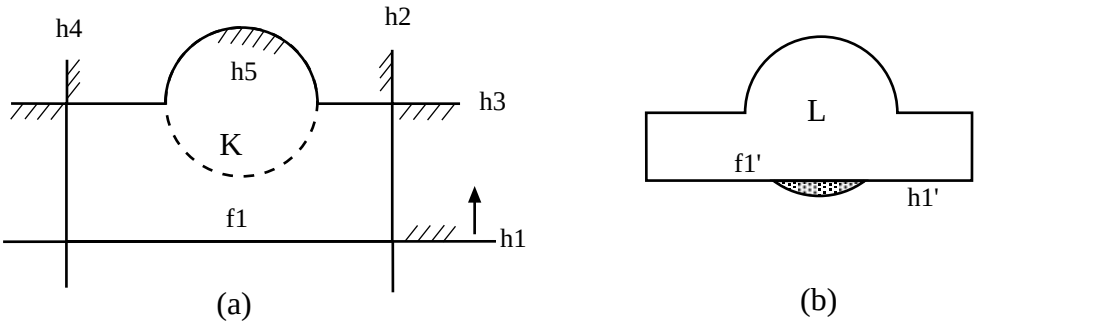


Figure 3: Inconsistent update. (a) B-rep K is consistent with CSG $(h_1 \cap^* h_2 \cap^* h_3 \cap^* h_4) \cup^* h_5$, (b) B-rep L is inconsistent with updated CSG

Suppose that a given CSG representation $F(\mathbf{P})$ and a b-rep K are consistent, for example because K is the result of boundary evaluation on $F(\mathbf{P})$. However after applying a parametric edit to a halfspace in $\mathbf{P}$ and updating the corresponding cell $\sigma_i \in K$, the new b-rep L may not be consistent with the set represented by the updated CSG $F(\mathbf{P}')$. Figure 3 shows an example: moving the halfspace h_1 and updating the corresponding cell f_1 results in inconsistency, as the shaded portion of the set represented by $F(\mathbf{P}')$ is not contained in the set represented by L.

In principle, we could ignore the original CSG and b-rep, and perform a representation conversion after every edit either to compute a new consistent b-rep or a new consistent CSG. But as we discussed earlier, we would like to avoid the complete representational conversion process because it is inefficient (particularly for small changes), unreliable (due to "persistent naming" problem), and is likely to destroy the original structure of the CSG model.

3.2 Consistency tests

Formally, two pointsets A and B are equal if ($A \subseteq B$ and $B \subseteq A$), or ($\bar{B} \subseteq \bar{A}$ and $\bar{A} \subseteq \bar{B}$). If A and B are two solids represented in the same scheme, the equality test is known as same object detection (SOD) [26] and is convenient to formulate in terms of symmetric difference: $(A -^* B) \cup (B -^* A) = \emptyset$. In our case, A and B are two different representations of a solid, and it is important to consider all possible formulations of the equality test. Noting that $A \subseteq B$ is equivalent to the relation $\bar{B} \subseteq \bar{A}$, and $B \subseteq A$ is equivalent to the relation $\bar{A} \subseteq \bar{B}$, we have four distinct but equivalent formulations of equality:

$$(A \subseteq B, \bar{A} \subseteq \bar{B}) \tag{1}$$

$$(B \subseteq A, \bar{B} \subseteq \bar{A}) \tag{2}$$

$$(A \subseteq B, B \subseteq A) \tag{3}$$

$$(\bar{B} \subseteq \bar{A}, \bar{A} \subseteq \bar{B}) \tag{4}$$

regardless of how A and B are represented. Without loss of generality, let us now assume that A is given by a CSG representation $F(\mathbf{P}')$ and B is given by a boundary representation L. Recall from section 2 that set $A = |F(\mathbf{P}')|$ is

the closure of the union of some d-dimensional sign-invariant cells C_j in the common space decomposition of E^d, and boundary $\partial B = |L|$ is a union of some lower-dimensional cells σ_k in the same cell decomposition. Because all cells are sign-invariant and connected, every d-cell C_j can classify only as IN or OUT with respect to any solid represented by the given set of primitives. Similarly, every lower-dimensional cell σ_k can classify as IN, ON or OUT with respect to such solid. (Details on set membership classification can be found in [25].)

We can now restate the four consistency tests in terms of classification of cells in the common decomposition of space. Each subset relationship $X \subseteq Y$ is established by testing if the cells represented by X are also in Y, and each cell is represented by a single representative point. Thus, all consistency tests will be reduced to a finite number of PMC tests, but the details of where these points are chosen and what representation they are classified against are different in each of the four cases.

CSG-based Consistency When set A is represented by CSG and set B represented by a b-rep, the consistency condition (1) is naturally restated for every d-cell C_j in the common decomposition:

$$C_j \subset A \Leftrightarrow C_j \subset B. \tag{5}$$

In other words, every d-cell C_j should have the same classification with respect to both the CSG representation $F(\mathbf{P}')$ and the boundary representation L. For example, CSG and b-rep in Figure 3(b) are not consistent because the shaded 2-cell classifies IN with respect to the shown CSG and OUT with respect to the updated b-rep. This consistency test does not require computing lower-dimensional cells σ, and computing points in each d-cell can be done quite reliably by halfspace offsetting, assuming that all cells in the decomposition are sufficiently large [19]. Cell classification test is also very robust, because the d-cells (and therefore the points representing the d-cells) can classify with respect to either representation in only one of two ways: IN or OUT.

Boundary-based Consistency Mathematically equivalent consistency condition (2) requires checking if a boundary-represented set B (or its complement) is a subset of a CSG-represented set A (or $\overline{A}$). Recalling that b-rep L is a collection of lower-dimensional cells σ_k, the two subset relationships may be combined and restated as

$$classify(\sigma_k, B) = classify(\sigma_k, A) \tag{6}$$

Assuming that $|L|$ is a valid boundary of a solid, it is sufficient to check only the $(d-1)$-cells in the common decomposition of space. These can classify with respect to either representation as IN, ON, or OUT.

Refer again to Figure 3(b). The inconsistency between L and $F(\mathbf{P}')$ can be easily identified by noting that the 1-dimensional circular cell bounding the shaded region classifies ON with respect to CSG representation, but OUT with respect to the b-rep; similarly the linear segment bounding the same region classifies IN with respect to CSG, but ON with respect to the boundary representation.

The boundary-based test does not require computing the d-cells in the common decomposition of space. On the other hand, computing, representing, and classifying the lower-dimensional cells in the decomposition is significantly more difficult than d-cells.[3] Because the cells σ_k can classify as ON with respect to the CSG and b-rep, the classification test requires resolving ON/ON ambiguities which rely on non-trivial neighborhood computations [25] and suffers from the usual robustness problems.

Hybrid Consistency The remaining two consistency conditions (3) and (4) involve classification of both d-cells and lower-dimensional cells in the common decomposition of space. For example, condition (3) requires checking if (a) every d-cell IN CSG-represented A is also IN boundary-represented B, and (b) every $(d-1)$-cell ON the boundary-represented B is ON the CSG-represented solid A. Consistency condition (4) implies similar test for $(d-1)$-cells and d-cells that classify OUT.

The main drawback of the hybrid tests is that they require computing the full sign-invariant decomposition of space; they also require utilities for resolving ON/ON ambiguities and may suffer from associated robustness issues. On the other hand, they appear to require a much smaller number of classification tests that are restricted to cells IN or OUT of $|F(\mathbf{P}')|$ (but not both).

[3] However, it should be noted that such computations are routinely performed in geometric modeling systems for solids bounded by planar or quadric surfaces and other restricted geometric domains.

It should be apparent that the CSG-based consistency test is more robust and simpler to implement, while the boundary-based and hybrid tests do not appear to offer significant computational advantages. For these reasons we chose to focus on the CSG-based consistency verification.

4 CSG-based consistency verification

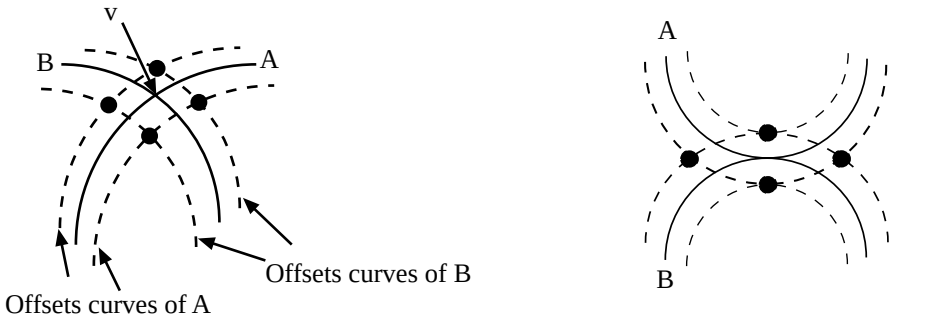
We describe two fully implemented algorithms for the CSG-based consistency verification: a brute force algorithm follows immediately from the formulation, and an incremental algorithm that achieves significant improvement in speed by incrementally maintaining the sign-invariant decomposition.

4.1 Computing sign-invariant decomposition

The CSG-based test requires only the d-cells of the sign-invariant decomposition of d-dimensional space with respect to all primitive halfspaces (surfaces) appearing in either b-rep or CSG representation. Recall that if K is a b-rep consistent with CSG representation $F(h_1, \ldots, h_n)$, then $|K| \subseteq (\partial h_1 \cup \ldots \cup \partial h_n)$ [13]. In other words, any halfspace induced from a boundary representation, must also appear in the consistent CSG representation. The reverse is not true: some halfspaces in CSG representations may not contribute to the boundary of the represented solid. [4] Recall also that every cell in the sign-invariant decomposition is adequately represented by a single point from that cell. Therefore, our requirements reduce to computing *at least* one point in every d-cell of the sign-invariant decomposition with respect to the set of halfspaces $\mathbf{P}$ used in the CSG representation $F(\mathbf{P})$.

Definition 2 (Characteristic set) *A set of points* $\mathbf{S}$ *will be called* characteristic *for a set of* d*-dimensional halfspaces* $\mathbf{P}$ *if* $\mathbf{S}$ *contains at least one point in every non-empty sign-invariant* d*-cell with respect to* $\mathbf{P}$.

Theoretically, if all halfspaces are algebraic, one could use exact (though often impractical) methods from real algebraic geometry to compute the characteristic set of points, with a single point in every sign-invariant cell [7]. But in fact, there is no harm in computing *more* than one point in every cell – this simply means that either some d-cells will be tested more than once, or that additional post-processing is required to remove the redundant points. We shall see below that the additional computational overhead implied by this approach is well compensated for by the resulting simplicity of implementation and efficiency of the incremental algorithm.



(a) Curves A and B with simple intersection

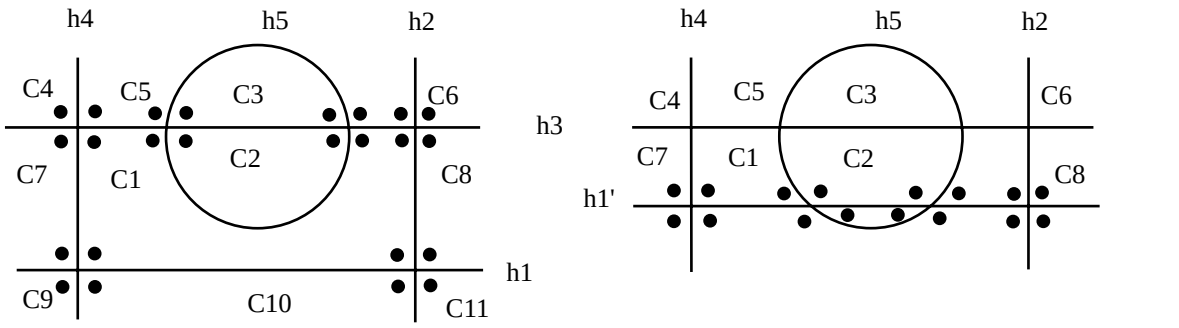
(b) Curves A and B with tangential intersection

Figure 4: Generation of characteristic points by offsetting two intersecting curves

The key observation is that the boundary of most sign-invariant d-cells contain one or more 0-cells (points) and/or 1-cells (curves), formed by the intersection of halfspace boundaries. Stated differently, the neighborhood of every

[4] Such halfspaces are often critical for CSG construction and are called *separating* in [21].

lower dimensional cell in the sign-invariant decomposition contains the characteristic points from the incident sign-invariant d-cells. The above observation immediately suggests a simple method of generating points in the sign-invariant d-cells; the same method was also used to implement the b-rep to CSG conversion algorithms in [19, 20]. Computing all points in the sign-invariant decomposition is achieved by testing all triples of halfspaces in $\mathbf{P}$ for intersections. Then the characteristic set of points for $\mathbf{P}$ in the sufficiently small neighborhoods of the 0- and 1-cells are computed by locally offsetting each contributing halfspace. If the boundary of a d-cell does not contain any 0- or 1-cell (for instance the 3-cell enclosed by two concentric spheres), then one can use any point on the offset halfspace. Figure 4 illustrates the procedure for two curves A and B in a plane. Figure 5(a) shows the full characteristic set for the two-dimensional decomposition induced from the solid in Figure 3(a). The offsetting procedure generalizes to three dimensions as well.



(a) Initial decomposition

(b) Updating halfspace h1

Figure 5: Incremental update of a sign-invariant decomposition for consistency verification

4.2 Brute force algorithm

Thus, we are led to a straightforward but inefficient implementation of the CSG-based consistency test for the updated CSG representation $F(\mathbf{P}')$ and b-rep L. Induce halfspaces from CSG $F(\mathbf{P}')$ and compute the characteristic set $\mathbf{S}$ of points for $\mathbf{P}'$, with at least one point from every non-empty d-cell. The points from $\mathbf{S}$ are then classified against L and $F(\mathbf{P}')$. Any point that classifies differently with respect to the two representations signals inconsistency. Referring again to Figure 5(a), it is easy to see that CSG and b-rep for two-dimensional solid in Figure 3(a) are consistent, because all shown characteristic points have the same classifications with respect to both representations. This leads to the brute force algorithm 1.

The running time of the algorithm is determined by the two tasks: generating the characteristic set of points, and classifying these points against CSG and b-rep. There are at most $O(n^3)$ (and usually much fewer) d-cells in the sign-invariant decomposition for n halfspaces of bounded algebraic degree [4]. Classifying a single point against a CSG representation takes $O(n)$ time, while classification of a point against b-rep is in the worst case linear in the number of faces in b-rep. We also note that the number of faces in the b-rep may be quadratic or worse in the number of the CSG halfspaces. Thus, the combined worst case running time appears to be growing faster than $O(n^4)$. The average running time of the algorithm may be improved by testing new points as soon as they are generated, but all points have to pass the test when the two representations are consistent.

It may be instructive to compare this algorithm with the b-rep to CSG conversion procedure described in section 2. The main difference between the two procedures is that the consistency verification does not require the more difficult steps in the conversion process: construction of additional halfspaces (separation) and Boolean optimization. Clearly, the size of the characteristic set is the single most important factor that determines the computation time. Unfortunately, the characteristic set of points cannot be reduced until *after* the consistency is assured, or at least all points are classified with respect to the b-rep.

```
procedure ConsistentBF(F(P'), L);
  // F(P') is the modified CSG
  // L is modified b-rep
  // Returns true if F(P') and L are consistent and false otherwise

  S = ∅;
  // Compute all the characteristic points using halfspaces in P'
  for every triplet (h_i, h_j, h_k) ∈ P',
    S = S∪ {characteristic points (h_i, h_j, h_k)}
  end-for;

  // Classify the characteristic points against the CSG and b-rep
  for every point X ∈ S,
    if Classify(X, L) ≠ Classify(X, F(P')), then
      error 'update resulted in inconsistency';
      return (false);
  end-for;

  return (true);

end-procedure;
```

Algorithm 1: Brute Force Consistency Test

4.3 Incremental algorithms

The above algorithm does not take into account locality of changes and invariance of classification of some cells with respect to CSG and b-rep during the update. An alternative to computing and classifying the characteristic set for every parametric edit is to compute the characteristic set of points once and then incrementally check and update it during every subsequent edit.

Suppose that after executing the brute force consistency check we found that CSG representation $F(\mathbf{P})$ and b-rep K are consistent. We can now significantly reduce the size of characteristic set $\mathbf{S}$ for $\mathbf{P}$, by observing that characteristic points with the same classifications with respect to all halfspaces in $\mathbf{P}$ must also have the same classification with respect to CSG $F(\mathbf{P})$ and therefore with respect to b-rep K (otherwise the two representations would not be consistent). Thus, a simple post-processing will reduce the size of $\mathbf{S}$ to one point per sign-invariant subset of the space, which corresponds to the (possibly disconnected) canonical intersection term Π_i defined in section 2.

A parametric edit is applied to both representations that transforms a halfspace h_e to a new halfspace h'_e leaving all other halfspaces unchanged. We update both representations and test for consistency by incrementally updating and re-testing the characteristic set $\mathbf{S}$ (see algorithm 2).

Incremental generation Transformation of h_e into h'_e is accompanied by two kinds of changes in the sign-invariant decomposition of space: some d-cells are merged into larger cells, while some other d-cells are split into smaller cells by the new halfspace h'_e. But the merged sign-invariant cells are *already* represented by some points in the characteristic set $\mathbf{S}$. Thus we only need to *add* some characteristic points to $\mathbf{S}$ representing the newly created d-cells. This in turn implies that the new characteristic points may be found in the neighborhood of those new 0 and 1-cells where h'_e intersects other halfspaces in $\mathbf{P}$.

Incremental testing The second observation is that, if the classification of a d-cell is the same with respect to $h_e{}'$ as it was with respect to h_e, then the cell's classification with respect to the CSG has not changed either. This follows directly from the existence of the CSG disjunctive canonical form summarized in section 2, and can be used to dramatically reduce the number of characteristic points that need to be classified against the updated CSG representation; however *all* points in $\mathbf{S}$ (old and new) must be classified with respect to the updated b-rep L.

After the characteristic set is incrementally updated and classified, and consistency of the updated representations is established, the characteristic set $\mathbf{S}$ can be reduced once again to one point for every sign-invariant subset of space.

procedure ConsistentINCR($F(\mathbf{P}')$, L, Π, h_e);
  // $F(\mathbf{P}')$ is the modified CSG
  // L is modified b-rep
  // Π corresponds to the set of points along with their classifications, in the
    canonical decomposition of the original CSG $F(\mathbf{P})$
  // h_e is original halfspace in $\mathbf{P}$
  // h'_e is the transformed version of h_e

  $\mathbf{H}^{int}$ = {set of all the halfspaces in $\mathbf{P}$ which intersect h'_e};

  $\mathbf{S} = \emptyset$;
  // Compute the new characteristic points locally using $\mathbf{H}^{int}$
  for every pair $(h_i, h_j) \in \mathbf{H}^{int}$,
    $\mathbf{S} = \mathbf{S} \cup$ {characteristic points (h_i, h_j, h'_e)}
  end-for;

  // Classify the new char. points against the new CSG and new b-rep
  for every point $X \in \mathbf{S}$,
    if Classify(X, L) $\neq$ Classify($X, F(\mathbf{P}')$), **then**,
      error '*Update resulted in inconsistency*';
      return (*false*);
  end-for;

  // Classify the old char. points selectively against new CSG and all of them against L
  for every point $Y \in \Pi$,
    $brepClass$ = Classify(Y, L);
    if Classify(Y, h_e) $\neq$ Classify(Y, h'_e), **then**,
      $csgClass$ = Classify($Y, F(\mathbf{P}')$);
    else, // Copy the old classification of the point Y
      $csgClass = Y.class$;

    if $csgClass \neq brepClass$, **then**,
      error '*Update resulted in inconsistency*';
      return (*false*);
  end-for;

  $\Pi = \Pi \cup \mathbf{S}$; // Merge the old and new cells, so that there is 1 pt/canonical term again
  return (*true*); // $F(\mathbf{P}')$ and L are consistent
end-procedure;

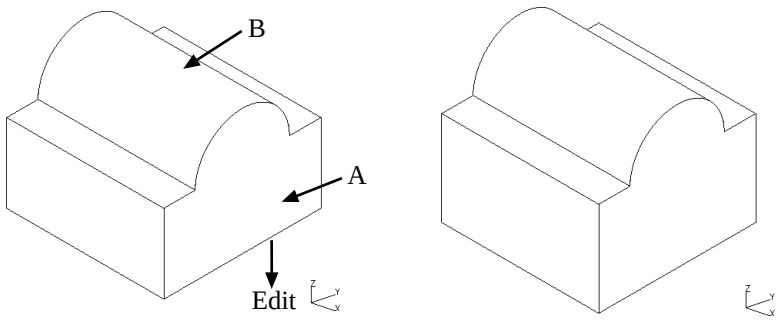
Algorithm 2: Incremental Consistency Test

Figure 5(b) shows the resulting space decomposition and all new characteristic points after halfspace h_1 in Figure 5(a) is transformed into a new halfspace $h_1{}'$. Classification of these points immediately signals the inconsistency, as two of them (the points inside the circle) classify IN with respect to updated CSG but OUT with respect to the updated b-rep. All other points have passed the consistency test. Also note that from these new points, only half classify differently with respect to h'_1 than with respect to h_1 and need to be tested against the updated CSG.

The resulting incremental algorithm 2 has the same asymptotic complexity as the brute force algorithm 1, since in the worst case we are still classifying $O(n^3)$ points against the updated b-rep L. However, even if the edited halfspace $h_e{}'$ intersects all n halfspaces, the algorithm would generate only $O(n^2)$ of new characteristic points. On average, as the number of halfspaces intersecting $h_e{}'$ will be far less than n, the total number of characteristic points is likely to be

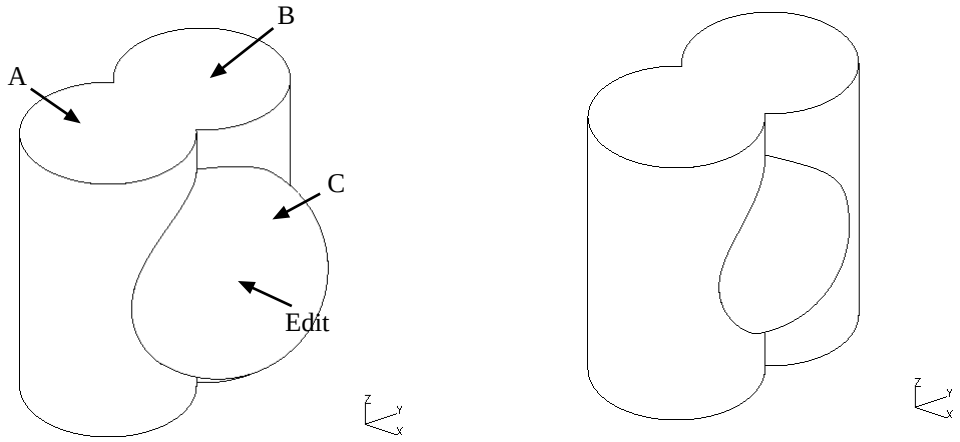
very close to one point per sign-invariant cell, and the number of cells is more likely to be linear or quadratic than cubic in the number of primitive halfspaces for most mechanical solids. We shall see below that the overall performance of the incremental algorithm is dramatically improved when compared to the original brute force algorithm.

4.4 Examples and timing



(a) Original b-rep K (b) Modified b-rep L

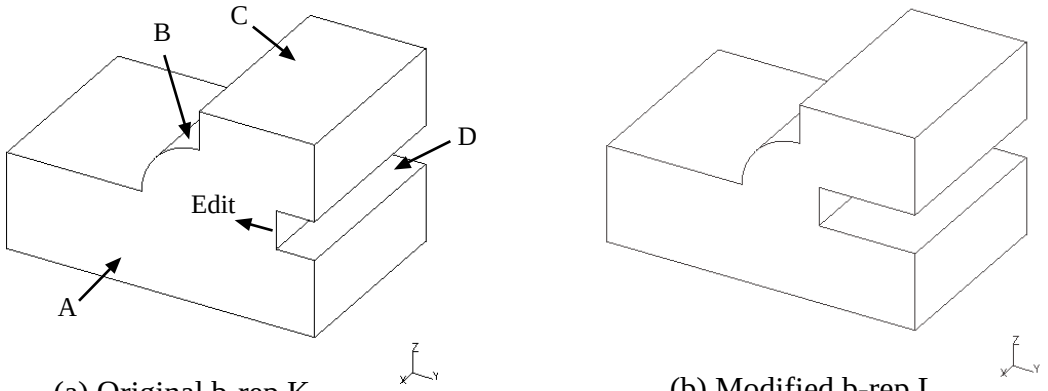
Figure 6: Solid is constructed as $A \cup^* B$ and the bottom face is moved down (consistent edit)



(a) Original B-rep K (b) Modified B-rep L

Figure 7: Solid is constructed as $((A \cup^* B) \cup^* C)$ and the indicated spherical face is moved backward (inconsistent edit)

Both algorithms were implemented in Parasolid and tested on Pentium II-400 workstation running WINDOWS-NT operating system. The five examples shown in figures 6, 7, 8, 9 and 10, were tested using both brute force and the incremental algorithms. All five solids were created using CSG operations on bounded halfspaces as shown in the figures, that were then stored as CSG expressions with unbounded halfspaces, and their b-reps were computed. Solid 1 in Figure 6 has 7 halfspaces, solid 2 in Figure 7 has 5 halfspaces, solid 3 in Figure 8 has 12 halfspaces, solid 4 in Figure 9 has 5 halfspaces and solid 5 in Figure 10 has 26 halfspaces. Then a face in each b-rep is transformed ("tweaked"), with Parasolid maintaining the integrity of the b-rep through its internal proprietary algorithms. The same transformations are then applied to the CSG halfspaces and consistency is verified using both algorithms. The specific



(a) Original b-rep K (b) Modified b-rep L

Figure 8: Solid is constructed as $(((A \cup^* C) -^* D) \cup^* B)$, and the indicated face is moved to the left, leading to inconsistency

Module/Test	*Solid 1*	*Solid 2*	*Solid 3*	*Solid 4*	*Solid 5*
Point generation (full)	0.15	0.11	0.88	0.17	4.14
Point classification (full)	0.00*	0.05	0.14	0.05	7.89
Brute force algorithm total	0.15	0.16	1.02	0.22	12.03
Point generation (incremental)	0.06	0.11	0.49	0.14	0.77
Point classification (incremental)	0.00*	0.02	0.03	0.02	0.76
Incremental algorithm total	0.06	0.13	0.52	0.16	1.53

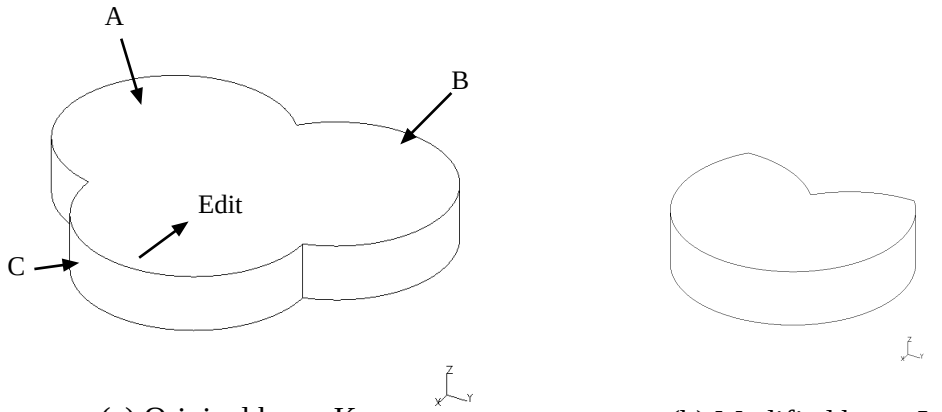
Table 1: Execution times in seconds on NT/PII-400 workstation for brute force and incremental consistency verification algorithms. Solid 1 is shown in Figure 6, Solid 2 is shown in Figure 7, Solid 3 is shown in Figure 8, Solid 4 is shown in Figure 9 and Solid 5 is shown in Figure 10. (* 0.00 indicates that the actual time is in the order of 10^{-4} and has been truncated.)

edits are shown in Figures 6, 7, 8, 9 and 10, and the summary of execution times is shown in Table 1. As expected, the incremental algorithm is much faster compared to the brute force algorithm, and the performance improvement increases rapidly with the complexity of the model.

5 Conclusions and extensions

5.1 Significance

We showed that consistency between a b-rep and a CSG representation may be maintained without repeated and expensive representation conversion procedures. Each representation may be edited independently, followed by a consistency check as described in this paper. If the two representations are consistent, then boundary evaluation of the CSG representation is guaranteed to produce the same b-rep. If the consistency check fails, then one of the representations has not been updated correctly. This may happen, for instance, when the CSG expression is no longer applicable, or when a direct boundary update was based on selecting a wrong root in a constraint solving system [2, 17]. This is the case, for example in Figure 9, where the system generated valid but inconsistent b-rep, even though the consistent update clearly exists. Since the consistency check may be done quickly, it may be practical to try several such solutions until a consistent update is found, if one exists. Ability to perform the direct and consistent updates of b-reps may significantly enlarge the class of operations allowed in solid modeling systems, while eliminating the need to solve the problem of "persistent naming".



(a) Original b-rep K (b) Modified b-rep L

Figure 9: Solid is constructed as $((A \cup^* B) \cup^* C)$ and the indicated cylindrical face is moved in the direction shown (inconsistent edit)

Our algorithms are provably correct, because our approach is based on the fact that *every* geometric representation has an equivalent canonical form as an union of sign-invariant cells in the corresponding space decomposition (minimal Whitney regular sign-invariant stratification of space)[18]. We saw that consistency may be checked in more than one way, but all algorithms followed directly from the properties of the corresponding decompositions of space. Maintaining such decompositions of space has been often considered impractical, but our work suggests that the benefits (such as correctness and simplicity of algorithms) may outweigh the required computational overhead. In particular, representing, maintaining, and computing with sign-invariant 3-cells can be done simply, robustly, and efficiently using a set of characteristic points as described in section 4. Furthermore, our implementations were relatively naive, and significant enhancements of the described algorithms are possible using the usual localization techniques (such as boxing). We demonstrated that sign-invariant decomposition facilitates systematic exploration and development of practical algorithms for consistency testing.

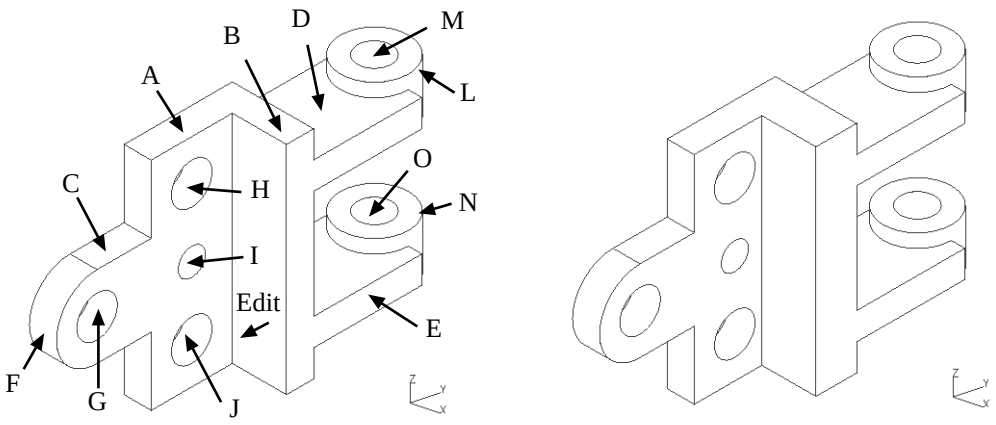
5.2 Semantic limitations

We tacitly assumed that a parametric update may be interpreted consistently in terms of both CSG and boundary representation; this assumption holds when the update may be expressed in terms of corresponding halfspaces. Semantics of updates that modify lower-dimensional entities in boundary representation (edges, vertices) is substantially more complicated and may have more than one valid interpretation [17]. More generally, it is well known [22] that parametric families associated with the two representations are not equivalent. In other words, some b-rep updates result in solids that cannot be described by the original CSG representation, and some CSG updates may produce drastically different (for example, disconnected) boundary representation.

The issues can be easily illustrated on a simple example of rectangle in Figure 11. Suppose that the vertex v_3 of boundary representation K is moved to a new location resulting in the transformed vertex v_3'. The semantics of this update on K can be defined in two different ways:

- v_3' could be considered as a vertex resulting from the *intersection* of translated halfspaces associated with edges e_2 and e_3, as shown in figure 11(b); or
- v_3' can be translated and the two edges e_2 and e_3 could be obtained by *interpolating* v_3' with vertices v_2 and v_4 respectively, as shown in figure 11(c).

Note that interpolation semantics is always well defined, but it is easy to construct examples where our intersection solution requires additional information, or may not exist [17]. Similarly, the CSG representation of rectangle may be parameterized several different ways in terms of positions and/or orientations of the four linear halfspaces, each leading to a distinct parametric family. For example, let us assume that the rectangle was parameterized in terms of



(a) Original b-rep K

(b) Modified b-rep L

Figure 10: Solid is constructed as $((A \cup^* B \cup^* C \cup^* D \cup^* E \cup^* F \cup^* L \cup^* N) -^* G -^* H -^* I -^* J -^* M -^* O)$ and the indicated planar face is moved forward (consistent edit)

only the position of the four halfspaces. In this case the b-rep update in figure 11(b) corresponds to a parametric change in the position of the two linear halfspaces corresponding to e_2 and e_3. But there is no equivalent parametric change for the update shown in figure 11(c). It should be apparent that, while direct b-rep edits appear to be more intuitive, their semantics is considerably more complex and could be restricted by topological constraints. At the same time, semantics of CSG edits is always well defined, but may be difficult or impossible to formulate consistently in terms of the corresponding boundary representation. In all cases, the consistency of the updates may be verified using our algorithms.

5.3 Consistency of other representations

Every geometric representation has a corresponding canonical form as a union of sign-invariant cells in the decomposition of space, and every cell can be uniquely represented by a single characteristic point [18]. The set of all such characteristic points may serve as a *certificate* for the representation of a pointset in terms of the given set of primitives. Because this certificate depends only on the choice of primitives and not on a particular representation scheme, it can be used to verify consistency between any and all representations relying on the same set of primitives. The notion of geometric certificate for individual cells in boundary representations was also proposed in [8], and cell-based (where a cell is represented by its boundary) techniques are used to deal with consistency of feature-based representations in [1].

We also note that CSG representations exist for all solids, and therefore the CSG-based consistency checks can be applied to non-CSG representations as well. Furthermore, the same method can be used to verify consistency of representations in different systems, provided that there is a one-to-one correspondence between the representational primitives. This may prove particularly useful in verifying consistency between boundary and/or parametric representations as they are being exchanged between distinct systems. As an illustration, consider how our algorithm for consistency checking may be extended to more general parametric definitions.

In a general setting, a parametric representation $F(\mathbf{P})$ is a procedural or declarative parametric definition whose semantics is determined by CSG operations, directed graph structures, and references to previously constructed geometry [6, 11, 15, 22]. Our algorithms may be applied to verify consistency of such representations with a corresponding b-rep under two reasonable assumptions:

- Set $\mathbf{P}$ of halfspaces must include all halfspaces that appear in, or induced from (for example from construction surfaces) the parametric definition;

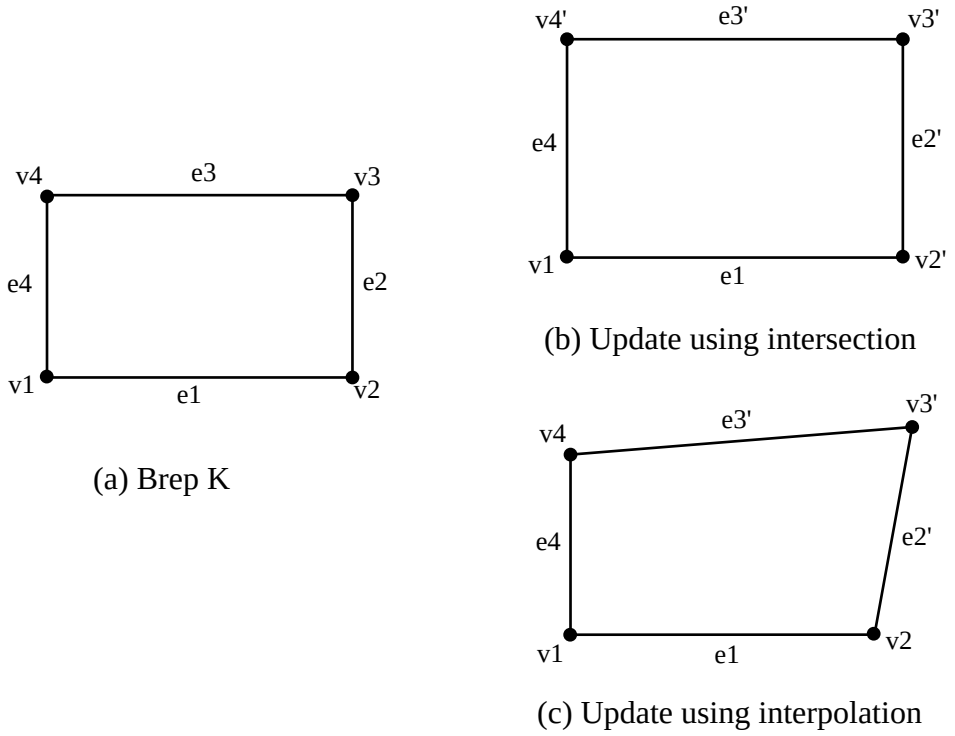


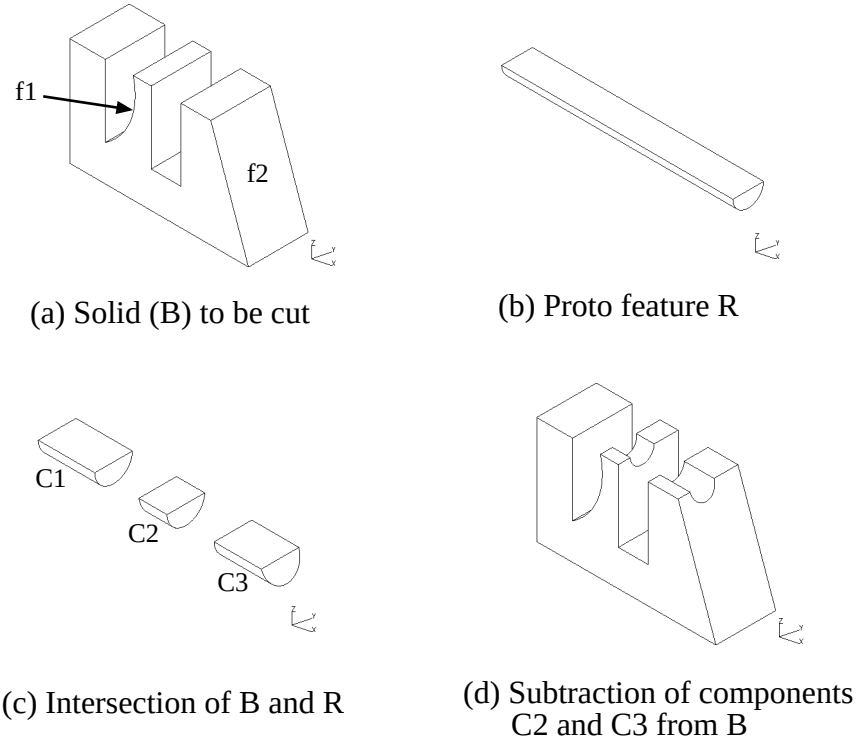
Figure 11: More general updates may or may not have consistent semantics in both representations

- Semantics of $F(\mathbf{P})$ is sufficiently defined to allow testing if an arbitrarily chosen point belongs to the constructed solid, hopefully without complete boundary evaluation. [5]

Satisfaction of these conditions is equivalent to admitting that the result of the procedural geometric construction is a well-defined union of sign-invariant cells in the decomposition of space. In this case, the described consistency algorithms generalize in a straightforward fashion: the sign-invariant 3-cells are generated and represented as before using a characteristic set of points, and these points are subsequently classified against both b-rep and parametric definition after every update.

Consider the feature attachment procedure defined in [6] and illustrated in Figure 12. A cut is to be made to solid B shown in Figure 12(a), using a proto feature R shown in figure 12(b), with attributes of *from* face f_1 *to* face f_2 of the solid. The procedural semantics of this construction consists of (1) defining the proto feature R, (2) regularized intersection of R with the solid body B (Figure 12(c)); (3) selecting the connected components of the intersection between the 'from-face' and 'to-face'; and (4) subtracting these components from the solid B (Figure 12(d)). For additional details see [6]. This semantics clearly supports a point membership classification test. Any given point p may be classified against the body B, proto feature R, and their regularized intersection. The point may be then tested to see if it also lies IN the two halfspaces induced from the 'from-face' and 'to-face', which would determine if the point classifies IN or OUT of the resulting solid. Thus, the CSG-based consistency check and our algorithm generalize to such constructions. Similar arguments may be used to show applicability of our techniques to procedural semantics proposed in [15], because it also relies on regularized set operations.

[5] Point membership classification algorithms may not be available for heuristically derived representations; nevertheless they exist for all unambiguous representations.



(a) Solid (B) to be cut

(b) Proto feature R

(c) Intersection of B and R

(d) Subtraction of components C2 and C3 from B

Figure 12: Major steps in a feature attachment[6]

5.4 Extensions

The algorithms described in this paper have been fully implemented for solids bounded by quadric surfaces. The proposed approach generalizes to solids bounded by any type of surfaces, provided that these surfaces may be used to induce the halfspaces, and that characteristic points may be generated in the neighborhood of intersections of the halfspaces. For example, it is conceivable that a parametric surface patch may be turned into a halfspace by creating a box with the patch as one of its faces.

It is easy to show that the local changes in a b-rep may be inconsistent with (previously consistent) CSG representation, and no consistent parametric update of CSG may be possible [22]. The present paper did not address the interesting issue of what can be done in such situations. One possibility would be to incrementally update the original CSG to account for discrepancy between the two representations by adding or subtracting those sign-invariant cells that have different classification with respect to b-rep and CSG. Another possibility would be to recompute a new CSG representation that is in some sense "close" to the original one, or to update the b-rep. The latter alternative amounts to incremental boundary evaluation. Deciding on these and numerous other possible solutions to this inconsistency problem requires identifying the precise measure of "closeness" between distinct parametric representations. All such solutions appear to demand the full machinery of the representation conversion technology described in [18].

ACKNOWLEDGMENTS

This research is supported in part by the National Science Foundation CAREER award DMI-9502728, grant DMI-9522806, DARPA contract N00014-96-C-0168 and Unigraphics Solutions Corporation.

References

[1] R. Bidarra, K. J. de Kraker, and W. F. Bronsvoort. Representation and management of feature information in a cellular model. *Computer-Aided Design*, 30(4):301–313, 1998.

[2] W. Bouma, I. Fudos, C. Hoffmann, J. Cai, and R. Paige. A geometric constraint solver. *Computer-Aided Design*, 27(6):487–501, 1995.

[3] I. C. Braid, R. C. Hillyard, and I. A. Stroud. Stepwise construction of polyhedra in geometric modeling. In *Mathematical Methods in Computer Graphics and Design, K. W. Brodlie ed.,*. Academic Press, 1980.

[4] J. Canny. *The Complexity of Robot Motion Planning*. PhD thesis, MIT, 1987.

[5] V. Capoyleas, X. Chen, and C. M. Hoffmann. Generic naming in generative, constraint-based design. *Computer-Aided Design*, 28(1):17–26, 1996.

[6] X. Chen and C. M. Hoffmann. Towards feature attachment. *Computer-Aided Design*, 27:675–702, 1995.

[7] J. Heintz, T. Recio, and M-F. Roy. Algorithms in real algebraic geometry and applications to computational geometry. In *Discrete and Computational Geometry: DIMACS special year, J. E. Goodman, R. Pollack, W. Steiger(Eds)*. American Mathematical Society, Baltimore, MD, 1991.

[8] C. M. Hoffmann and R. Joan-Arinyo. CAD and the product master model. *Computer-Aided Design*, 30(11):905–918, 1998.

[9] J. Kripac. A mechanism for persistently naming topological entities in history-based parametric solid models. *Computer-Aided Design*, 29(2):113–122, 1997.

[10] M. Mantyla. *An Introduction to Solid Modeling*. Computer Science Press, Maryland, USA, 1988.

[11] A. Rappoport. Breps as displayable-selectable models in interactive design of families of geometric objects. In *Theory and Practice of Geometric Modeling*. Blaubeuren II, Tubingen, Germany, October 1996.

[12] A. A. G. Requicha. Representations for rigid solids: Theory, methods and systems. *ACM Computing Surveys*, 12:437–464, December 1980.

[13] A. A. G. Requicha and R. B. Tilove. Mathematical foundations of constructive solid geometry: General topology of closed regular sets. Technical report, TM-27a, PAP, University of Rochester, Rochester, NY, June 1978.

[14] A. A. G. Requicha and H. B. Voelcker. Boolean operations in solid modeling: Boundary evaluation and merging algorithms. *Proceedings of the IEEE*, 73(2):30–44, January 1985.

[15] J. R. Rossignac, P. Borrel, and L. R. Nackman. Interactive design with sequences of parameterized transformations. In *Proc. of 2nd Eurographics Workshop in Intelligent CAD Systems: Implementation Issues*, april 1988.

[16] J. R. Rossignac and H. B. Voelcker. Active zones in CSG for accelerating boundary evaluation, redundancy elimination, interference detection and shading algorithms. *ACM Transactions on Graphics*, 8(1):51–87, January 1989.

[17] M. Sabin and V. Shapiro. Modifications in cellular models. Technical report, NA-09, DAMTP, University of Cambridge, UK, September 1998.

[18] V. Shapiro. Maintenance of geometric representations through space decompositions. *International Journal of Computational Geometry and Applications*, 7(4):383–418, 1997.

[19] V. Shapiro and D. L. Vossler. Construction and optimization of CSG representations. *Computer-Aided Design*, 23(1):4–20, January 1991.

[20] V. Shapiro and D. L. Vossler. Efficient CSG representations of two dimensional solids. *Transactions of the ASME, Journal of mechanical design*, 113:292–305, September 1991.

[21] V. Shapiro and D. L. Vossler. Separation for boundary to CSG conversion. *ACM Transactions on Graphics*, 12(1):35–55, January 1993.

[22] V. Shapiro and D. L. Vossler. What is a parametric family of solids? In *3rd ACM Symposium on Solid Modeling and Applications, Salt Lake City, Utah, May*, 1995.

[23] C. E. Silva. Alternative definitions of faces in boundary representations of solid objects. Technical report, TM-36, PAP, University of Rochester, Rochester, NY, November 1981.

[24] S.Raghothama and V.Shapiro. Boundary representation deformation in parametric solid modeling. *ACM Transactions on Graphics*, 17(4):259–286, October 1998.

[25] R. B. Tilove. Set membership classification: A unified approach to geometric intersection problems. *IEEE Transactions on Computer*, C-29(10), October 1980.

[26] R. B. Tilove. Exploiting spatial and structural locality in geometric modeling. Technical report, TM-38, PAP, University of Rochester, Rochester, NY, October 1981.