Extending CSG with Projections: Towards Formally Certified Geometric Modeling

George Tzoumas[a,*], Dominique Michelucci[a], Sebti Foufou[b]

[a] *CNRS UMR 5158, LE2I*
University of Burgundy, France
[b] *Qatar University, Qatar*

Abstract

We extend traditional Constructive Solid Geometry (CSG) trees to support the projection operator. Existing algorithms in the literature prove various topological properties of CSG sets. Our extension readily allows these algorithms to work on a greater variety of sets, in particular parametric sets, which are extensively used in CAD/CAM systems. Constructive Solid Geometry allows for algebraic representation which makes it easy for certification tools to apply. A geometric primitive may be defined in terms of a characteristic function, which can be seen as the zero-set of a corresponding system along with inequality constraints. To handle projections, we exploit the Disjunctive Normal Form, since projection distributes over union. To handle intersections, we transform them into disjoint unions. Each point in the projected space is mapped to a contributing primitive in the original space. This way we are able to perform gradient computations on the boundary of the projected set through equivalent gradient computations in the original space. By traversing the final expression tree, we are able to automatically generate a set of equations and inequalities that express either the geometric solid or the conditions to be tested for computing various topological properties, such as homotopy equivalence. We conclude by presenting our prototype implementation and several examples.

Keywords: geometric modeling, constructive solid geometry, projection, homotopy equivalence, topological properties, formal methods, constraint solving, disjunctive normal form

1. Introduction

Constructive Solid Geometry was one of the first representation schemes in Computer Aided Design and Manufacturing (CAD/CAM) [1, 2, 3]. However, dealing with complex scenes efficiently can be costly, since a rigorous mathematical description is obtained. As a result, other representations have been preferred such as Boundary Representations (Brep) or Selective Geometric Complex (SGC) [4, 5]. Modelers using the latter representations are hybrid and procedural, solids are no longer represented by systems of equations and inequalities. This lack of algebraic representation makes geometric models (and modeling software) hard to certify.

On the other hand, CSG modeling is quite prevalent in the video game industry and computer graphics, with traditional game engines like Quake or Unreal [6] making extensive use of it through binary space partitioning (BSP) [7]. The latter engine has been used by hundreds of commercial titles [8]. Moreover, it is the preferred method of some open-source and web-based modelers and there is recent work for speeding up CSG rendering through late polygonization [9].

In CSG, solids are represented as Boolean constructions via regularized[1] set operators (union, intersection, difference). CSG representations are essentially binary expression trees where non-terminal nodes represent operators and terminal nodes typically represent primitive solids (like spheres, cones, cuboids). By traversing the expression tree, one may derive an algebraic representation of the solid, in the form of a system of equations and inequalities. The set of Boolean operations is quite restrictive for practical use.

In this paper we extend the descriptive power of classical CSG by introducing the projection operator. This immediately allows us to deal with a greater variety of objects, *i.e.*, objects defined as projections of other objects or parametric objects, including those defined by extrusions and sweeps. We do so by describing the projected objects as classical CSG expressions in the projected (lower-dimension) space. We show that our method can be used to extend existing algorithms in the literature that compute various properties of CSG sets like connectivity and topology [10, 11]. Moreover, since (extended) CSG derives an algebraic representation of the geometric models, it provides an elegant way for certification tools to apply.

In hardware design, formal verification is a significant part of the design process [12]. The infamous Pentium

*Corresponding author
Email addresses: George.Tzoumas@u-bourgogne.fr (George Tzoumas), Dominique.Michelucci@u-bourgogne.fr (Dominique Michelucci), sfoufou@qu.edu.qa (Sebti Foufou)

[1]A regularized set is equal to the closure of its interior.

bug and more recently, the TSX Haswell bug are only a few cases of misdesigns. We naturally raise the question: What about misdesigns in CAD?

Current certification methods which are relevant to our applications include computer algebra methods, model checkers or proof assistants like PVS [13] or Coq [14] (Coq was recently used to check the proof of the four color theorem and the proof of the Feit-Thompson theorem for the classification of simple groups) and last but not least, interval analysis. The latter has been used to prove the Kepler conjecture by Hales [15]. Under a more algebraic-geometric context, in CSG modeling, it suffices to prove the correctness of an underlying interval solver, instead of proving all the procedures currently used in today's procedural modeling. See [16] for an example where interval analysis is being used, relying on the properties of the tensorial Bernstein basis, to certify critical software that prevents collisions in air traffic.

Other methods for certified computing are exact algebraic algorithms which can typically be combined with interval arithmetic for efficiency. See [17] for an application regarding exact boundary computation on low degree sculptured solids and [18] for accurate BRep generation from a CSG expression. Exact algebraic algorithms have also become popular in the Computational Geometry area [19]. All these efforts aim to redefine modern computing, through efficient algebraic algorithms allowing us to step further from the inexact floating-point arithmetic of the 80s. We believe our work is a small step in this direction, in particular towards formally certified geometric modeling.

Almost no literature exists regarding projections in CSG. There is some discussion in [20], where it is mentioned that projections are non-trivial to handle and the author deals only with the projection of unions, since projections propagate over unions. In this paper we show how to deal with intersections, by transforming intersections into disjoint unions.

A geometric primitive in $\mathbb{R}^d$ is represented as a manifold f in $(d+1)$-space $(x_1, \dots, x_d, s)$ where s is the *characteristic variable* of the manifold. This way, the geometric primitive consists of interior points where the s-coordinate is negative and of exterior points where the s-coordinate is positive. The case $s = 0$ corresponds (in general) to the boundary of the object. Thus the geometric primitive is essentially a solid in d dimensions. Through relation $f(\mathbf{x}; s) = 0$, $\mathbf{x} \in \mathbb{R}^d$, s is implicitly defined by a characteristic function. For example, manifold

$$f(x, y, s) = x^2 + y^2 - 1 - s = 0$$

describes the unit disk as a paraboloid in 3-space. For every point $(\hat{x}, \hat{y})$ inside the unit disk, there exists $\hat{s} \leq 0$ such that $f(\hat{x}, \hat{y}, \hat{s}) = \hat{x}^2 + \hat{y}^2 - 1 - \hat{s} = 0$. See Fig. 1, for a contour graph with respect to s. Due to the regularized set condition, we consider inequalities $s \leq 0$ or $s \geq 0$ instead of their strict counterparts. This modeling via the characteristic variable is in accordance with the classical representation as a finite Boolean combination of semi-algebraic or semi-analytic sets of the form $F(\mathbf{x}) \leq 0$, where $F : \mathbb{R}^d \to \mathbb{R}$. However in our approach, the use of characteristic variables proves more convenient.

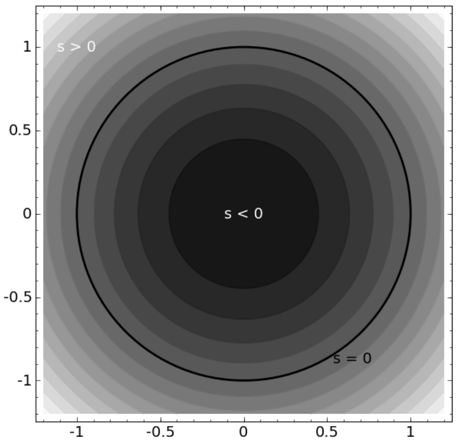

Figure 1: Contour graph of $s = x^2 + y^2 - 1$

We consider each node of the expression tree separately in order to compute a *contributing primitive* for each point in the set. That is in the case of projections, every point is associated with a point in the higher dimension space. The tree is traversed and a set of simple subsystems with inequality constraints is generated. We denote this *set* as $\mathcal{F}(\mathbf{x}; s)$, where s is the characteristic variable. Individual equations are still denoted as $f(\mathbf{x}; s)$.

A geometric set is described in *Disjunctive Normal Form (DNF)* as a union of intersections of primitives (Sec. 2). Having the union operator at the top level has the advantage that projections distribute over unions (Sec. 3). Such canonical forms are also considered in other algorithms dealing with CSG representations [21].

The paper is organized as follows. Section 2 presents the computation of the DNF for CSG operators. Section 3 deals with projections and parametric objects showing how to obtain a classical CSG expression in the lower-dimension space. Section 4 deals with gradient computations in the projected space, that allow us to compute topological properties. Section 5 presents several applications of our approach. Our reference implementation is presented in section 6 and finally, in section 7 we conclude with discussion and future extensions.

2. Constructive Solid Geometry operations

2.1. Disjunctive Normal Form

A CSG formula can be converted to DNF by applying De Morgan's laws and by distributing $\cap$ over $\cup$ as follows. Let A, B be geometric primitives. Let P, Q_i be geometric sets defined by an expression tree.

1. A and $\neg A$ are in DNF.

2. $A \cup B$ and $A \cap B$ are in DNF.
3. $\neg(Q_1 \cup Q_2) = \neg Q_1 \cap \neg Q_2$
4. $\neg(Q_1 \cap Q_2) = \neg Q_1 \cup \neg Q_2$
5. $P \cap (Q_1 \cup Q_2 \cup \ldots \cup Q_n) = (P \cap Q_1) \cup (P \cap Q_2) \cup \ldots \cup (P \cap Q_n)$

We also merge consecutive binary operators of the same kind as follows:

6. $(Q_1 \cap Q_2) \cap Q_3 = Q_1 \cap Q_2 \cap Q_3$
7. $(Q_1 \cup Q_2) \cup Q_3 = Q_1 \cup Q_2 \cup Q_3$

By applying the above rules we end up having an expression tree where the topmost operator is $\cup$ and each operand sub-expression contains only the $\cap$ operator applied to either a primitive or its complement. Note that conversion to DNF may result in an exponential explosion of the formula (*e.g.*, the DNF of $(A_1 \cup B_1) \cap \cdots \cap (A_n \cup B_n)$ has 2^n terms). On the other hand, classical methods like propagation of bounding boxes [22, 23] can discard useless DNF clauses and reduce the number of nodes in the expression tree.

2.2. Complement

The simplest operation is the complement. If for primitive A: $f(\mathbf{x}; s) = 0$ then $\neg A : f(\mathbf{x}; -s) = 0$.

Remark 1. *It is important that the complement is still represented by points from the manifold.*

Recall that we require the characteristic variable to be defined for all points in $\mathbb{R}^d$. As a counter-example, consider $f(x, y, s) = x^2 + y^2 - 1 + s^2 = 0$. Points outside the unit disk, like $(2, 2)$ are not represented at all, since $f(2, 2, s) = 0$ cannot be satisfied for $s \in \mathbb{R}$.

2.3. Intersection and dominant sets

Let $P = A_1 \cap \ldots \cap A_n$ be an intersection of primitives. We have to impose the constraint that point $\mathbf{x}$ belongs to all A_i, $i = 1 \ldots n$ *at the same time.* This is achieved by setting the characteristic variable s of the set to be $s = \max(s_1, \ldots, s_n)$ (cf. Fig. 2). In order to represent max, we consider the union of all possible cases for max, *i.e.*, $s = s_1$ and $\forall j : s_1 \geq s_j$ or $s = s_2$ and $\forall j : s_2 \geq s_j$, and so on.

Definition 1. *Let A, B_i be geometric sets and s_A and s_{B_i} their characteristic variables at point $\mathbf{x}$. We say that A* dominates *$B_1, \ldots, B_n$ and denote $A|B_1, \ldots, B_n$ for all $\mathbf{x} \in \mathbb{R}^d$ where $0 \geq s_A \geq s_{B_i}$, $i = 1, \ldots n$. In this case A is a* dominant set *over each B_i.*

The notion of dominant set allows us to express intersections as unions. Note that operator $|$ has lower precedence than the comma in the above notation. Consequences of the above definition are the following properties:

(i) $A \cap B = A|B \cup B|A$ (see Fig. 2 and 3)
(ii) $(A|B)|C = A|B, C = A|C, B$

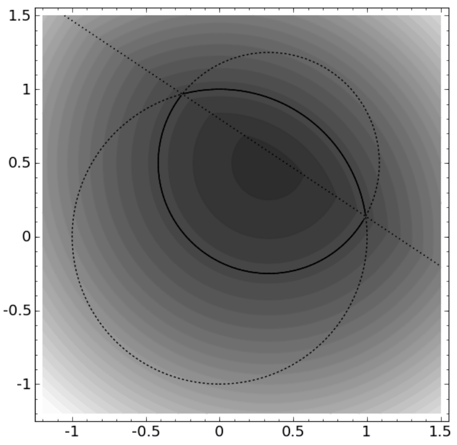

Figure 2: Contour graph for the intersection of two disks, where $s = \max(s_1, s_2)$

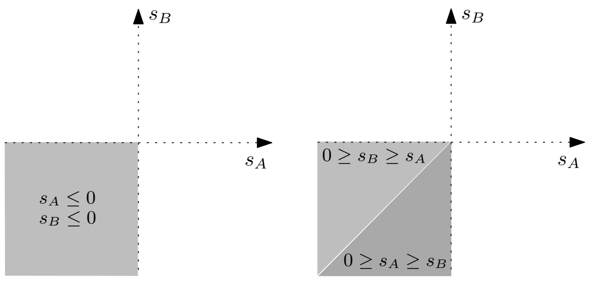

Figure 3: Plot of the characteristic variables for $A \cap B = A|B \cup B|A$

(iii) $A|(B|C) \cup A|(C|B) = A|B, C$
(iv) $A \cap B \cap C = (A|B, C) \cup (B|C, A) \cup (C|A, B)$
(v) $\neg(A|B) = \neg A \cup \neg B \cup B|A$

Finally, the difference $A \setminus B$ is handled trivially via intersection: $A \setminus B = A \cap \neg B$.

2.4. Union

Let $P = A_1 \cup \ldots \cup A_n$ be a union of primitives. Since the formula is in DNF form, it suffices to consider each primitive separately, that is $A_i : f_i(\mathbf{x}; s_i)$, $i = 1 \ldots n$. Note that a point $\mathbf{x}$ may not be uniquely associated with a primitive, since it may belong to the intersection of many primitives, *i.e.*, the unions may not be disjoint. In our context (for topological property computation) this does not cause any problems. In the case where one requires disjoint unions, we can impose the uniqueness constraint by satisfying $\forall j : s_i \leq s_j$, where s_j refers to the characteristic variable of each object in the considered union. We choose the characteristic variable of the set to be $\min(s_1, \ldots, s_n)$ and this way we obtain similar properties as those in the case of intersections.

3. Projection

The first non-trivial operation which concerns sets that cannot be described with CSG primitives is *projection.* Let

$A : \mathcal{F}(\mathbf{x}; s)$ where $\mathbf{x} = (x_1, \ldots, x_d)$, then the projection of A with respect to x_i is denoted as $\pi_i(A)$. Let

$$\mathbf{x^i} = (x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_d).$$

Then $\pi_i(A) = \mathcal{F}_\pi(\mathbf{x^i}; s)$. Projections with respect to more than one dimension are denoted with commas, *e.g.*

$$\pi_{i,j}(A) = \mathcal{F}_\pi(\mathbf{x^{i,j}}; s) = \pi_j(\pi_i(A)).$$

When the particular dimension is not of importance we may simply write $\pi^2(A) = \pi(\pi(A))$. An interesting property of the projection is that it distributes over $\cup$:

$$\pi(Q_1 \cup Q_2 \cup \ldots \cup Q_n) = \pi(Q_1) \cup \pi(Q_2) \cup \ldots \cup \pi(Q_n).$$

A naive way to deal with projections is to just "forget" coordinate x_i. Doing so however, may result in a k-dimensional ($k \geq 1$) set of values for x_i and s, such that $\mathcal{F}(\mathbf{x}; s)$ is satisfied. This may slow down an interval solver (used to find a cover of the set for example) since an infinite set of solutions will have to be covered. We remedy this problem by introducing extra constraints so as to limit the range of the characteristic variable to a 0-dimensional set of values for every projected point.

One way to consistently generate additional constraints is to choose the smallest value of the characteristic variable, among all possible values of coordinates $x_1, \ldots, x_k$ (the coordinates being eliminated). That is $\pi_{1,\ldots,k}(A)$ is the set of points $\mathbf{x^{1\ldots k}} \in \mathbb{R}^{d-k}$ such that $\exists\, x_1, \ldots, x_k$ where $\mathbf{x} \in \mathbb{R}^d \cap A$ and s is minimal. This way we pick the point that lies "deepest" in the set to map to the projected set. Note that the use of the term "minimal" in the above is abusive. We are actually looking for a *critical* point (without loss of generality), *i.e.*, the derivative of the characteristic function with respect to s vanishes. Thus, we don't have to perform extra computations to ensure that a critical point is actually a minimum. This is because we are interested in reducing the solution set to a hopefully 0-dimensional variety. It is perfectly acceptable for a point in the interior of the (projected) set to have not necessarily the smallest value of the characteristic variable, but some other (critical) value.

Note that coordinates $x_1, \ldots, x_k$ are no longer free variables, but take a value and become parameters. The minimization constraint can be written in terms of an optimization problem with constraints those exactly in $\mathcal{F}$ and the objective function s. Typical approach involves considering a Lagrangian (*i.e.*, the Fritz John conditions). This is quite powerful a technique, but it has the disadvantage that it introduces extra equations and unknowns and our experiments have shown that after 6 or 7 unknowns solution times become impractical. Here we make use of differential calculus and wedge products. For an introduction to wedge products and their applications in optimization problems the reader may refer to [24]. The wedge product vanishes if its operands are linearly dependent. Lagrange multipliers encapsulate linear dependence, therefore optimization problems may be efficiently transformed to calculus with wedge products.

The leaves of the DNF expression tree contain primitives (or complements of primitives) or dominant sets. Consequently, to distribute projections, we need to handle projections of primitives and projections of dominant sets. For the latter, we will define an auxiliary construction, called *join set* which we will show that it contributes to the projection.

Theorem 1 (Projection of geometric primitive)**.** *Let $A : f(\mathbf{x}; s)$ be a geometric primitive. When projecting down k dimensions (eliminating $x_1 \ldots x_k$), the projection can be specified by the following additional constraints:*

$$\frac{\partial f}{\partial x_1} = \frac{\partial f}{\partial x_2} = \ldots = \frac{\partial f}{\partial x_k} = 0.$$

Proof. $0 = ds \wedge df = ds \wedge (\frac{\partial f}{\partial x_1} dx_1 + \ldots + \frac{\partial f}{\partial x_k} dx_k + \frac{\partial f}{\partial x_s} dx_s) = \frac{\partial f}{\partial x_1} ds \wedge dx_1 + \ldots + \frac{\partial f}{\partial x_k} ds \wedge dx_k \iff \frac{\partial f}{\partial x_1} = \ldots = \frac{\partial f}{\partial x_k} = 0$. An alternative proof is also possible using Lagrange multipliers. □

Note that we assume that there exists some critical value in the interior of the set (we can guarantee that by construction), otherwise we would have to consider the boundary of the set.

Definition 2. *Let A, B be geometric sets and s_A and s_B their characteristic variables at point $\mathbf{x}$. We define the* join *set $A \bowtie B$ as:*

$$A \bowtie B \;:=\; \mathbf{x} \in \mathbb{R}^d :\; s_A = s_B \;\wedge\; s_A \leq 0.$$

We define the precedence of the new operators to be: $\neg \succ\, ,\, \succ \,|\, \succ \bowtie \succ \cap \succ \cup$. Observe the similarity with the join operator from relational algebra. Indeed, we join the two relations $A(\mathbf{x}; s_A)$ and $B(\mathbf{x}; s_B)$ on their characteristic variable. Since the join operator introduces an equation, the dimension of the join set drops. See for example Fig. 2. The dotted line is the join of the two disks. Obviously, this line is no longer a solid in 2 dimensions.

Definition 3. *We denote with $J_{i_1 i_2 \ldots i_n}(f_1, f_2, \ldots, f_n)$ the following $n \times n$ Jacobian determinant:*

$$\begin{vmatrix} \frac{\partial f_1}{\partial x_{i_1}} & \frac{\partial f_1}{\partial x_{i_2}} & \cdots & \frac{\partial f_1}{\partial x_{i_n}} \\ \frac{\partial f_2}{\partial x_{i_1}} & \frac{\partial f_2}{\partial x_{i_2}} & \cdots & \frac{\partial f_2}{\partial x_{i_n}} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial f_n}{\partial x_{i_1}} & \frac{\partial f_n}{\partial x_{i_2}} & \cdots & \frac{\partial f_n}{\partial x_{i_n}} \end{vmatrix}$$

Lemma 1 (Projection of join sets)**.** *Let $A : f_0(\mathbf{x}; s)$ and $B_i : f_i(\mathbf{x}; s), i = 1 \ldots n$ be geometric primitives. Then to describe $\pi^k(A \bowtie B_1 \bowtie \cdots \bowtie B_n)$ we additionally consider (i) no constraints when $k \leq n$; (ii) $J_{i_0 i_1 \ldots i_n}(f_0, f_1, \ldots, f_n) = 0$, $1 \leq i_0 < i_1 < \ldots < i_n \leq k$, when $k > n$.*

Proof. Basic idea: We shall exploit wedge products to capture the linear dependence in order to minimize s. Not surprisingly, these operations yield equivalently the Jacobian determinant.

Assume without loss of generality that we are projecting with respect to $x_1, x_2, \ldots, x_k$. Considering the wedge product (to find the critical value of s) we have $ds \wedge df_0 \wedge df_1 \wedge df_2 \wedge \cdots \wedge df_n = ds \wedge (\sum_{i=1}^{k} \frac{\partial f_0}{\partial x_i} dx_i + \frac{\partial f_0}{\partial s} ds) \wedge (\sum_{i=1}^{k} \frac{\partial f_1}{\partial x_i} dx_i + \frac{\partial f_1}{\partial s} ds) \wedge \cdots \wedge (\sum_{i=1}^{k} \frac{\partial f_n}{\partial x_i} dx_i + \frac{\partial f_n}{\partial s} ds) = \epsilon_{x_{i_0} x_{i_1} \ldots x_{i_n}} (\sum_{j=0}^{n} \frac{\partial f_j}{\partial x_{i_j}} dx_{i_j}) ds \wedge dx_{i_0} \wedge \cdots \wedge dx_{i_n}$. Symbol ϵ is the permutation sign determined by the number of inversions in the considered permutation, which appears in the combinatorial definition of the determinant [25]. Now, if $k \leq n$ the wedge product is identically zero, because of some dx_i being equal, due to the pigeonhole principle (we have a wedge product of $n+1$ factors with k choices for each factor, and we have that $dx_i \wedge dx_i = 0$). Otherwise, if $k > n$, the wedge product expands to $\binom{k}{n+1}$ coefficients which should all vanish. These coefficients are precisely $J_{i_0 i_1 \ldots i_n}(f_0, f_1, \ldots, f_n), 1 \leq i_0 < i_1 < \ldots < i_n \leq k$. □

Since no extra condition is required to describe a projection of a join with respect to a single variable, we have that

Corollary 1. $\pi(A \bowtie B) \; = \; A \bowtie B$.

Theorem 2 (Projection of dominant set)**.**

$$\pi^k(A|B) \; = \; \pi^k(A)|B \; \cup \; \pi^k(A \bowtie B).$$

Proof. Without loss of generality we assume that we project with respect to $x_1, \ldots, x_k$. Since we have a constrained optimization problem, the critical value can be attained either when a constraint is active or not.

$\pi^k(A|B) = \mathbf{x}^{\mathbf{1} \ldots \mathbf{k}} \in \mathbb{R}^{d-k} : \exists\, x_1, \ldots, x_k : (\mathbf{x} \in \mathbb{R}^d \cap A) \wedge (s_A \text{ is critical}) \wedge (s_A \geq s_B)$. This means that s_A takes its critical value on the critical points of $\pi(A)$ that happen to satisfy $s_A \geq s_B$, which is precisely $\pi^k(A)|B$ or somewhere where $s_A = s_B$, which is $\pi^k(A \bowtie B)$. Alternative proofs are possible using wedge products or Lagrange multipliers [26]. □

Care has to be taken here that the set B in the expression $\pi^k(A)|B$ lies in a lower dimension. That is we consider points in $\pi^k(A)$ that happen to lie in B. We could denote B in this case as $B_{/\pi^k(A)}$ but we avoid so due to abuse of notation. Let $[B_m]^{1:n}$ denote sequence B_m, $m = 1 \ldots n$. The following theorem comes as a generalization of Theorem 2.

Theorem 3 (Projection of dominant sets, generalized)**.** $\pi^k(A|[B_m]^{1:n}) =$

$$\begin{array}{cl}
 & \pi^k(A)|[B_m]^{1:n} \\
\bigcup\limits_{i=1}^{n} & \pi^k(A \bowtie B_i)|[B_m]^{1:n}_{m \neq i} \\
\bigcup\limits_{\substack{i,j=1 \\ i<j}}^{n} & \pi^k(A \bowtie B_i \bowtie B_j)|[B_m]^{1:n}_{m \neq i, m \neq j} \\
\bigcup & \cdots \\
\bigcup & \pi^k(A \bowtie B_1 \bowtie \cdots \bowtie B_n)
\end{array}$$

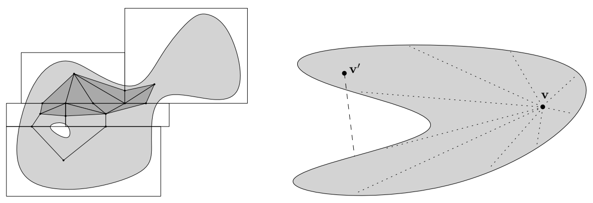

Figure 4: Left: Simplicial complex homotopy equivalent to the light-gray shaded object; Right: $\mathbf{v}$ is a star, while $\mathbf{v'}$ is not a star

3.1. Projection and parametric sets

The way join sets were defined allows us to express the intersection of manifolds: a join is performed on the common variables, hence the common variable is implicitly eliminated. The idea is therefore to use joins to eliminate the parameters, as illustrated in Sec. 5. For example $X : x - \cos(t) = 0$, $Y : y - \sin(t) = 0$. Now $X \bowtie Y$ expresses points (x, y) that lie on the unit circle. This way, the join variable t is implicitly eliminated, because we project with respect to t, due to the join operator being applied. Lemma 1 shows that projection of joins may be trivial if the number of variables projected is less than the number of terms in the join expression. Here $X \bowtie Y$ has 2 terms therefore, the resulting subspace consists of x and y only.

More generically, given a parametric solid G in $\mathbb{R}^d$ defined by $X_i = f_i(\mathbf{x}; t_1, \ldots, t_d)$, $i = 1 \ldots d$, we can represent this set as the join of the defining manifolds. That is $G = X_1 \bowtie X_2 \bowtie \ldots \bowtie X_d$. Assume that t_d is the characteristic variable. Now projection in the first $d-1$ dimensions eliminates the corresponding variables, and we have from Lemma 1 that $\pi^{d-1}(X_1 \bowtie \ldots \bowtie X_d) = X_1 \bowtie X_2 \bowtie \ldots \bowtie X_d$, since $k = d - 1 < d$. See Sec. 5 for examples.

4. Homotopy equivalence, star test and gradient

We shall adapt the algorithm of [10] to work with CSG sets extended with projection. The algorithm is called HIA (Homotopy via Interval type Analysis) and it works by computing a contractible cover of an initial set $\mathbb{S}$ and generating an abstract simplicial complex homotopy equivalent to $\mathbb{S}$ (Fig. 4 left). The basic predicate of the algorithm is the *star test*, which provides a sufficient condition of contractability. For completeness, we recall the corresponding definitions.

A point $\mathbf{v}$ is a star for a subset X of an Euclidean set if X contains all the line segments connecting any of its points and $\mathbf{v}$ (cf. Fig. 4 right). We call X star-shaped or $\mathbf{v}$-star-shaped. If X and Y are two $\mathbf{v}$-star-shaped sets, then $X \cap Y$ and $X \cup Y$ are also $\mathbf{v}$-star-shaped. A topological space X which is homotopy-equivalent to a point is contractible, therefore a star-shaped set is contractible. The following sufficient condition for contractability can be checked with interval analysis.

Proposition 1. *Let $f : \mathbb{R}^n \to \mathbb{R}$ be a C^1 function, D be a convex set and $\mathbb{S} = \{\mathbf{x} \in D \subset \mathbb{R}^n | f(\mathbf{x}) \leq 0\}$. If there exists $\mathbf{v}$ in $\mathbb{S}$ such that*

$$\{\mathbf{x} \in D | f(\mathbf{x}) = 0 \wedge \nabla f(\mathbf{x}) \cdot (\mathbf{x} - \mathbf{v}) \leq 0\} = \emptyset$$

then $\mathbb{S}$ is star-shaped.

Observe that the equation $\nabla f(\mathbf{x}) \cdot (\mathbf{x} - \mathbf{v}) = 0$ expresses the tangent plane at the boundary point $\mathbf{x}$, since it is satisfied for all points perpendicular to the normal vector. Taking into account the characteristic function, the boundary of a primitive is $f(\mathbf{x}; 0) = 0$, therefore the star condition is written as $\nabla f(\mathbf{x}; 0) \cdot (\mathbf{x} - \mathbf{v}) \leq 0$. We set $F(\mathbf{x}) := f(\mathbf{x}; 0)$. Now we may rewrite $\nabla F(\mathbf{x}) \cdot (\mathbf{x} - \mathbf{v}) \leq 0$.

For primitive $f(\mathbf{x}; s)$, the boundary of the projection is given from $f(\mathbf{x}; 0)$ and $\frac{\partial f}{\partial x_i}(\mathbf{x}; 0) = 0$. We define $F(\mathbf{x}) = f(\mathbf{x}; 0)$. Now we have $F(\mathbf{x}) = \frac{\partial F}{\partial x_i}(\mathbf{x}) = 0$. The latter equation implies a relation $x_i = G(\mathbf{x^i})$ and by substitution in the first one we obtain $R(\mathbf{x^i}) = 0$ (recall that $\mathbf{x^i}$ denotes vector $(x_1, x_2, \ldots, x^{i-1}, x^{i+1}, \ldots, x_n)$). The process is similar when more variables are eliminated. When the functions are multivariate polynomials, $R(\mathbf{x^i})$ is precisely the resultant of the polynomial and its derivative with respect to variable x_i (this particular resultant is a multiple of the discriminant).

Lemma 2. *Given $F(\mathbf{x}) = 0$ and $\frac{\partial F}{\partial x_i} = 0, i = 1 \ldots k$. Then there exist functions y_i so that $x_i = y_i(\tilde{\mathbf{x}}), i = 1 \ldots k$ where $\mathbf{x} \in \mathbb{R}^d$ and $\tilde{\mathbf{x}} = (x_{k+1}, x_{k+2}, \ldots, x_d)$. Therefore we may write $F(\mathbf{x}) = F(y_1, y_2, \ldots, y_k, x_{k+1}, x_{k+2}, \ldots, x_d)$.*

Proof. Basic idea: We take a pair of the given equations that has not been considered so far, and eliminate a variable using the implicit function theorem.

We have $F(\mathbf{x}) = 0$ and $F'(\mathbf{x}) = \frac{\partial F}{\partial x_1}(\mathbf{x}) = 0$. Subtracting the two equalities yields an implicit equation in $x_1, \ldots, x_d$. For readability purposes we denote the comma separated list $x_i, x_{i+1}, \ldots, x_j$ with $x_{i \ldots j}$. From the implicit function theorem we have $x_1 = g_1(x_{2 \ldots d})$. Replacing x_1 in F yields $F\big(g_1(x_{2 \ldots d}), x_{2 \ldots d}\big) = 0$. We also obtain $\frac{\partial F}{\partial x_i}\big(g_1(x_{2 \ldots d}), x_{2 \ldots d}\big) = 0, i = 2 \ldots k$. Now we apply the same process to $F\big(g_1(x_{2 \ldots d}), x_{2 \ldots d}\big)$ and $\frac{\partial F}{\partial x_2}\big(g_1(x_{2 \ldots d}), x_{2 \ldots d}\big)$ which will yield

$$\begin{aligned} F\Big(g_1\big(g_2(x_{3 \ldots d}), x_{3 \ldots d}\big), g_2\big(x_{3 \ldots d}\big), x_{3 \ldots d}\Big) &= 0 \\ \frac{\partial F}{\partial x_3}\Big(g_1\big(g_2(x_{3 \ldots d}), x_{3 \ldots d}\big), g_2\big(x_{3 \ldots d}\big), x_{3 \ldots d}\Big) &= 0 \end{aligned}$$

and so on until we arrive at $R(\tilde{\mathbf{x}}) = F\Big(g_1\big(g_2\big(\cdots g_k(\tilde{\mathbf{x}}) \cdots\big), \ldots\big), g_2\big(\cdots\big), \ldots, g_k, x_{(k+1) \ldots d}\Big) = 0$. Therefore

$$y_i(\tilde{\mathbf{x}}) = g_i\Big(g_{i+1}\big(\cdots g_k(\tilde{\mathbf{x}}) \cdots\big), \ldots\Big), i = 1 \ldots k.$$

□

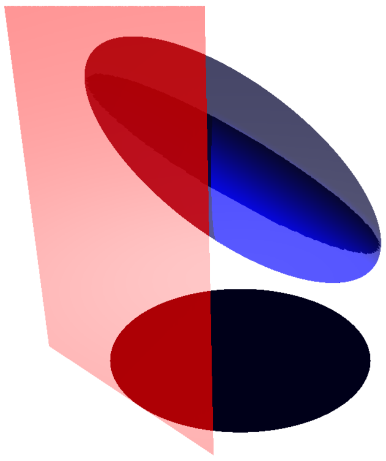

Figure 5: Tangent of projection of primitive

Theorem 4. *Let $R(\tilde{\mathbf{x}}) = 0$ be the boundary of the projection of primitive $f(\mathbf{x}; s) = 0$ with respect to $x_1, x_2, \ldots, x_k$, where $\mathbf{x} \in \mathbb{R}^d$ and $\tilde{\mathbf{x}} = (x_{k+1}, x_{k+2}, \ldots, x_d)$. Then*

$$\nabla R(\tilde{\mathbf{x}}) \cdot (\tilde{\mathbf{u}} - \tilde{\mathbf{x}}) = \nabla F(\mathbf{x}) \cdot (\mathbf{u} - \mathbf{x}),$$

where $F(\mathbf{x}) = f(\mathbf{x}; 0)$.

Proof. Since we project with respect to $x_1, x_2, \ldots, x_k$ we have $\frac{\partial F}{\partial x_i} = 0, i = 1 \ldots k$. It follows from Lem. 2 that there exist functions y_i so that $x_i = y_i(\tilde{\mathbf{x}}), i = 1 \ldots k$. We have that $R(\tilde{\mathbf{x}}) = F(y_1, y_2, \ldots, y_k, x_{k+1}, x_{k+2}, \ldots, x_d)$. We trivially set $x_i = y_i(\tilde{\mathbf{x}}), i = (k+1) \ldots d$ so that we may now write $R(\tilde{\mathbf{x}}) = F(\mathbf{y}(\tilde{\mathbf{x}}))$. Now from the chain rule we have $\frac{\partial R}{\partial x_i} = \sum_{j=1}^{d} \frac{\partial F}{\partial y_j} \frac{\partial y_j}{\partial x_i}, i = (k+1) \ldots d$. We also have that $\frac{\partial F}{\partial y_j} = 0$, $j = 1 \ldots k$ (by hypothesis due to projection, seeing F as a function in $\mathbf{y}$ instead of $\mathbf{x}$) and that $\frac{\partial y_j}{\partial x_i}$ equals $\frac{\partial y_j}{\partial x_i}$, when $j \leq k$, or 0, when $j > k \wedge j \neq i$ or 1, when $j = i$. This yields in the end $\frac{\partial R}{\partial x_i} = \frac{\partial F}{\partial y_i}$. Finally, $\nabla R(\tilde{\mathbf{x}}) \cdot (\tilde{\mathbf{u}} - \tilde{\mathbf{x}}) = \sum_{i=k+1}^{d} \frac{\partial R}{\partial x_i}(\tilde{\mathbf{x}})(u_i - x_i) + 0 = \sum_{i=k+1}^{d} \frac{\partial F}{\partial y_i}(\mathbf{y})(u_i - y_i) + \sum_{i=1}^{k} \frac{\partial F}{\partial y_i}(\mathbf{y})(u_i - y_i) =$

$$\nabla F(\tilde{\mathbf{y}}) \cdot (\mathbf{u} - \mathbf{y}) \equiv \nabla F(\tilde{\mathbf{x}}) \cdot (\mathbf{u} - \mathbf{x}).$$

□

The geometric interpretation of Thm. 4 is that the tangent plane at a point of the boundary in the projected space, is equal to the projection of the tangent plane in the original space (Fig. 5). As such, in order to compute the star test on the projection of a primitive, it suffices to compute the star test in the original space (before projection). Therefore, no computation of the implicit relation R is required, and the test can be evaluated using only the known relation f. The following lemma comes as a generalization of Lem. 2.

Lemma 3. *Given $F_i(\mathbf{x}) = 0, i = 0 \ldots n$. Then there exist functions y_i so that $x_i = y_i(\tilde{\mathbf{x}}), i = 1 \ldots k \leq n$ where $\mathbf{x} \in$*

$\mathbb{R}^d$ *and* $\tilde{\mathbf{x}} = (x_{k+1}, x_{k+2}, \ldots, x_d)$*. Therefore we may write* $R(\tilde{\mathbf{x}}) = F(y_1, y_2, \ldots, y_k, x_{k+1}, x_{k+2}, \ldots, x_d)$*.*

Proof. This is a generalization of Lem. 2. We have $F_0(\mathbf{x}) = F_1(\mathbf{x}) = 0$. Subtracting the two equalities yields an implicit equation $R_0(\mathbf{x}) = 0$. From the implicit function theorem we have $x_1 = g_1(x_{2\ldots d})$. Now we replace x_1 in $F_i, i = 0 \ldots n$ and get $F_i\big(g_1(x_{2\ldots d}), x_{2\ldots d}\big) = 0$, $i = 0 \ldots n$. Now we apply the same process to F_0 and F_2 and get $R_1(x_{2\ldots d}) = 0$, $x_2 = g_2(x_{3\ldots d})$ and

$$F_i\Big(g_1\big(g_2(x_{3\ldots d}), x_{3\ldots d}\big), g_2(x_{3\ldots d}), x_{3\ldots d}\Big) = 0, \ i = 0 \ldots n.$$

Finally we obtain $R(\tilde{\mathbf{x}}) = F_i(g_1(g_2(\cdots g_k(\tilde{\mathbf{x}}) \cdots), \ldots), g_2(\cdots), \ldots, g_k, x_{k+1}, x_{k+2}, \ldots, x_d) = 0$, $i = 0 \ldots n$. Therefore

$$y_i(\tilde{\mathbf{x}}) = g_i(g_{i+1}(\cdots g_k(\tilde{\mathbf{x}}) \cdots), \ldots), \ i = 1 \ldots k.$$

Note that in the end $F_0(\tilde{\mathbf{x}}) \equiv F_1(\tilde{\mathbf{x}}) \equiv \ldots \equiv F_k(\tilde{\mathbf{x}})$. Instead of $0 \ldots k$ we can apply the same elimination process to any $(k+1)$-subset of the $(n+1)$ equations. We have also assumed generic enough functions, *i.e.*, no two functions are identically the same. □

Theorem 5. *Let* $R(\tilde{\mathbf{x}}) = 0$ *be the boundary of the projection of primitive* $\pi^k(A \bowtie B_1 \bowtie \cdots \bowtie B_n)$ *with respect to* $x_1, x_2, \ldots, x_k$*, where* $\mathbf{x} \in \mathbb{R}^d$, $\tilde{\mathbf{x}} = (x_{k+1}, x_{k+2}, \ldots, x_d)$ *and* $k \leq n$, $d - k \geq 2$*. Let* $f_i(\mathbf{x}; s)$ *be the characteristic functions and* $F_i(\mathbf{x}) = f_i(\mathbf{x}; 0)$*. Then the value* S *of the star test* $S = \nabla R(\tilde{\mathbf{x}}) \cdot (\tilde{\mathbf{u}} - \tilde{\mathbf{x}})$ *is the* $(k+1)$*-th coordinate of the solution vector* $\mathbf{w}$ *of system* $(\mathbf{J}|[-1])\mathbf{w} = \mathbf{a}$*, where* $\mathbf{J}$ *the* $(k+1) \times k$ *Jacobian matrix of* $F_0, F_1, \ldots, F_k$ *with respect to* $(x_1, \ldots, x_k)$*,* $(\mathbf{J}|[-1])$ *is* $\mathbf{J}$ *augmented with column* $(-1, -1, \ldots, -1)^T$*,* $\mathbf{w} = (w_1, w_2, \ldots, w_k, S)^T$ *and* $\mathbf{a} = (a_0, a_1, \ldots, a_k)^T$ *with*

$$a_j = -\sum_{i=k+1}^{d} \frac{\partial F_j}{x_j}(u_i - x_i), \ j = 0 \ldots k.$$

Proof. From Lem. 3 we have that

$$R(\tilde{\mathbf{x}}) = F_m\big(y_1(\tilde{\mathbf{x}}), y_2(\tilde{\mathbf{x}}), \ldots, y_k(\tilde{\mathbf{x}}), x_{k+1}, x_{k+2}, \ldots, x_d\big),$$

$m = 0 \ldots n$. We trivially define

$$y_i(\tilde{\mathbf{x}}) := x_i, \ i = (k+1) \ldots d,$$

so that we may write $R(\tilde{\mathbf{x}}) = F_m(\mathbf{y}(\tilde{\mathbf{x}}))$. Applying the chain rule we obtain $\frac{\partial R}{\partial x_i} = \sum_{j=1}^{d} \frac{\partial F_m}{\partial y_j} \frac{\partial y_j}{\partial x_i}$. Now, as in the proof of Thm. 4, $\frac{\partial y_j}{\partial x_i}$ equals 0 when $j > k, j \neq i$ and 1 when $j = i$. Therefore $\frac{\partial R}{\partial x_i} = \sum_{j=1}^{k} \frac{\partial F_m}{\partial y_j} \frac{\partial y_j}{\partial x_i} + \frac{\partial F_m}{\partial y_i}$. We have that $S = \nabla R(\tilde{\mathbf{x}}) \cdot (\tilde{\mathbf{u}} - \tilde{\mathbf{x}}) \Leftrightarrow \sum_{i=k+1}^{d} \frac{\partial F_m}{\partial y_i}(u_i - x_i) + \sum_{i=k+1}^{d} \sum_{j=1}^{k} \frac{\partial F_m}{\partial y_j} \frac{\partial y_j}{\partial x_i}(u_i - x_i) - S = 0$, $m = 0 \ldots k$. We set $a_m := -\sum_{i=k+1}^{d} \frac{\partial F_m}{\partial y_i}(u_i - y_i)$ (recall that $x_i = y_i$ when $i > k$) and rewrite (swapping sums):

$$\sum_{j=1}^{k} \frac{\partial F_m}{\partial y_j} \sum_{i=k+1}^{d} \frac{\partial y_j}{\partial x_i}(u_i - x_i) - S = a_m.$$

Now we set $w_j := \sum_{i=k+1}^{d} \frac{\partial y_j}{\partial x_i}(u_i - x_i)$, $j = 1 \ldots k$ and $w_{k+1} := S$ which allows us to rewrite $(\mathbf{J}|[-1])\mathbf{w} = \mathbf{a}$, where $\mathbf{J}$, the $(k+1) \times k$ Jacobian matrix of $F_0, F_1, \ldots, F_k$ with respect to $(x_1, \ldots, x_k)$ is augmented with column $(-1, -1, \ldots, -1)^T$.

Note that the Jacobian can be defined using any $(k+1)$-subset of the $(n+1)$ functions. We have also assumed that $d \geq k+2$, so that the dimension of the space after projection is at least 2 and the notion of a tangent hyperplane makes sense. □

Lemma 4. *The geometric interpretation of Thm. 5 is that the tangent plane at the projected space, is equal to the projection of the intersection of the tangent planes in the original space, for* $k+1$ *primitives participating in the join (cf. Fig. 6).*

Proof. A tangent plane in the original space is given as $\nabla F_m(\mathbf{u} - \mathbf{x}) = 0$ (with $m = 0 \ldots k$). The intersection of all $k+1$ tangent planes is an over-constrained system in the first k dimensions. Equation in line m is

$$\sum_{i=1}^{d} \frac{\partial F_m}{\partial x_i}(u_i - x_i) = 0 \quad \Leftrightarrow$$
$$\sum_{i=1}^{k} \frac{\partial F_m}{\partial x_i}(u_i - x_i) + \sum_{i=k+1}^{d} \frac{\partial F_m}{\partial x_i}(u_i - x_i) = 0 \quad \Leftrightarrow$$

$$\sum_{i=1}^{k} \frac{\partial F_m}{\partial x_i}(u_i - x_i) = a_m, \tag{1}$$

with a_m defined as in Thm. 5.

The tangent plane at the projected space is obtained when the star test (Thm. 5) evaluates to zero. Replacing $S = 0$ in Thm. 5 we obtain the following system:

$$\sum_{i=1}^{k} \frac{\partial F_m}{\partial x_i} w_i = a_m, \tag{2}$$

with $m = 0 \ldots k$. It is obvious now that systems (1) and (2) are equivalent. □

Now we are able to perform the star test on $\pi(A)|B$. First, observe that $A \bowtie B \subseteq \pi(A)$, by definition. Moreover $\pi(A)|B$ appears in an expression paired with $A \bowtie B$, from Thm. 2. Let $a = \pi(A)|B$, $b = \pi(B)|A$ and $c = \pi(A \bowtie B) = A \bowtie B$. Now we have $\pi(A \cap B) = a \cup b \cup c = (a \setminus c) \cup (b \setminus c) \cup c$. The boundary of each set in the union consists either of the boundary of the projection of a primitive ($\pi(A)$, $\pi(B)$) or of the boundary of $A \bowtie B$. Moreover, we have that if two sets are star-shaped, then their intersection and union is star-shaped as well. Therefore Thm. 4 and 5 allow us to compute the gradient and as a result, the outcome of the star test.

5. Examples

In this section we present some examples of our proposed approach dealing with projections. Note that union operators pose no problems, since they generate independent systems and we can typically handle hundreds of

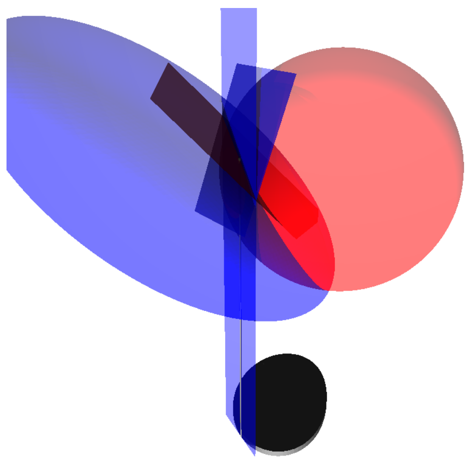

Figure 6: Tangent of projection of join set (subset of the projection of the intersection of two primitives)

unions independently. Therefore we focus on *projections* of intersections which provide the biggest challenge. Currently our algorithm is not optimized for handling many levels of projection (note the combinatorial explosion in Sec. 5.2). However, as mentioned in Sec. 2.1, techniques like propagation of bounding boxes allow us to detect terms that do not contribute to the final result and quickly skip over them. Finally, our examples can be easily converted to any dimension, however, we restrict to 2D and 3D because they are easier to visualize and understand.

5.1. Projection of ellipsoid

Consider the ellipsoid from Fig. 5:

$$f(x, y, z, s) = 2x^2 + y^2 + 3z^2 + 2\sqrt{3}xz - 1 - s = 0.$$

Then from Thm. 1 we have

$$\begin{array}{rcl} f(x,y,z,s) = 0 & \wedge & \frac{\partial f}{\partial z} = 6z + 2\sqrt{3}x = 0 \quad \Leftrightarrow \\ x^2 + y^2 - 1 - s = 0 & \wedge & z = -\frac{\sqrt{3}}{3}x \end{array}$$

which is effectively the unit disk.

5.2. Intersection of projections of intersection of primitives

Let E_1, E_2, E_3 be three primitives in $\mathbb{R}^d$, we want to express the object G defined as $G = \pi(E_1 \cap E_2) \cap \pi(E_1 \cap E_3)$. The above expression will be transformed in DNF. We have that: $G = [\pi(E_1|E_2) \cup \pi(E_2|E_1)] \cap [\pi(E_1|E_3) \cup \pi(E_3|E_1)] = [\pi(E_1|E_2) \cap \pi(E_1|E_3)] \cup [\pi(E_1|E_2) \cap \pi(E_3|E_1)] \cup [\pi(E_2|E_1) \cap \pi(E_1|E_3)] \cup [\pi(E_2|E_1) \cap \pi(E_3|E_1)] = [\pi(E_1)|E_2 \cup E_1 \bowtie E_2] \cap [\pi(E_1)|E_3 \cup E_1 \bowtie E_3] \cup \cdots = [\pi(E_1)|E_2 \cap \pi(E_1)|E_3] \cup \cdots = [\pi(E_1)|E_2, \pi(E_1)|E_3)] \cup [\pi(E_1)|E_3, \pi(E_1)|E_2] \cup \cdots = \bigcup_{i=1}^{18} S_i$. That is G is equal to the union of 18 sets which in fact can be grouped into five sets depending on the contributing set being $\pi(E_1)$, $\pi(E_2)$, $\pi(E_3)$, $E_1 \bowtie E_2$, $E_1 \bowtie E_3$, as shown in Tab. 1. Note how this expansion of the formula is independent of the dimension. To visualize the

set	contributing set	formula
S_1	$\pi(E_1)$	$\pi(E_1)\|E_2, \pi(E_1)\|E_3$
S_2		$\pi(E_1)\|E_3, \pi(E_1)\|E_2$
S_3		$\pi(E_1)\|E_2, E_1 \bowtie E_3$
S_4		$\pi(E_1)\|E_3, E_1 \bowtie E_2$
S_5		$\pi(E_1)\|E_2, \pi(E_3)\|E_1$
S_6		$\pi(E_1)\|E_3, \pi(E_2)\|E_1$
S_7	$\pi(E_2)$	$\pi(E_2)\|E_1, \pi(E_1)\|E_3$
S_8		$\pi(E_2)\|E_1, E_1 \bowtie E_3$
S_9		$\pi(E_2)\|E_1, \pi(E_3)\|E_1$
S_{10}	$\pi(E_3)$	$\pi(E_3)\|E_1, \pi(E_1)\|E_2$
S_{11}		$\pi(E_3)\|E_1, E_1 \bowtie E_2$
S_{12}		$\pi(E_3)\|E_1, \pi(E_2)\|E_1$
S_{13}	$E_1 \bowtie E_2$	$E_1 \bowtie E_2\|(\pi(E_1)\|E_3)$
S_{14}		$E_1 \bowtie E_2\|E_1 \bowtie E_3$
S_{15}		$E_1 \bowtie E_2\|(\pi(E_3)\|E_1)$
S_{16}	$E_1 \bowtie E_3$	$E_1 \bowtie E_3\|(\pi(E_1)\|E_2)$
S_{17}		$E_1 \bowtie E_3\|E_1 \bowtie E_2$
S_{18}		$E_1 \bowtie E_3\|(\pi(E_2)\|E_1)$

Table 1: DNF expression terms and their contributing sets

above, let (x, y, z, r) denote a sphere centered at (x, y, z) with radius r. If $E_1 = (0, 0, 0, 1)$, $E_2 = (\frac{1}{2}, 0, \frac{1}{2}, \sqrt{\frac{3}{2}})$ and $E_3 = (-\frac{3}{2}, 0, \frac{3}{2}, \frac{3}{2})$ then $\pi(E_1 \cap E_2)$, $\pi(E_1 \cap E_3)$ and $\pi(E_1 \cap E_2) \cap \pi(E_1 \cap E_3)$ are shown in Fig. 7 and 8. $E_1 \bowtie E_2$, $\pi(E_1)|E_2$ and $E_1 \bowtie E_3$ are also visible.

5.3. Parametric annulus in $\mathbb{R}^2$

Let $X(t, r) = x - (\frac{1}{2} + r)\cos t$, $Y(t, r) = y - (\frac{1}{2} + r)\sin t$, $R(r, s) = r(r - 1) - s$. Equation $R(r, s) = 0$ restricts the characteristic variable s, so that $s \le 0$ when $r \in [0, 1]$. Then $\pi_{r,t}(R \bowtie X \bowtie Y)$ is a 2D annulus in the xy-space, as shown in Fig. 9 left. The parametric construction $\pi_r(R \bowtie X \bowtie Y)$ in 3D (xyt-space), before being projected down with respect to t-axis, is an infinite spiral ribbon along the t-axis (Fig. 9 right). This example demonstrates an artificial example for the join operator, which however is not very practical, as we don't obtain a solid in every dimension (the ribbon is not a 3D solid – it unavoidably loses one dimension due to the fact that we have 2 joins).[2] There is an alternative formulation that gives us a solid, both in 3 and 2 dimensions, which also allows us to compute the gradient applying Thm. 4. We set $f(x, y, t; s) = (x\cos(-t))^2 + (y - \sin(-t))^2 - 1/4 - s$. Now we have a cylindrical spiral in the 3D space (x, y, t) which projects down to an annulus in 2D (cf. Fig.10). Note that the minus sign in t has been used merely for illustration purposes (to force a clockwise drawing order). Visible space (x, y) has dimension 2 for visualization convenience. The theory applies to any dimension, though it is nontrivial to visualize solids in more than three dimensions.

[2]The join operator is mainly used to describe the projection of dominant sets, in order to describe the projection of intersections.

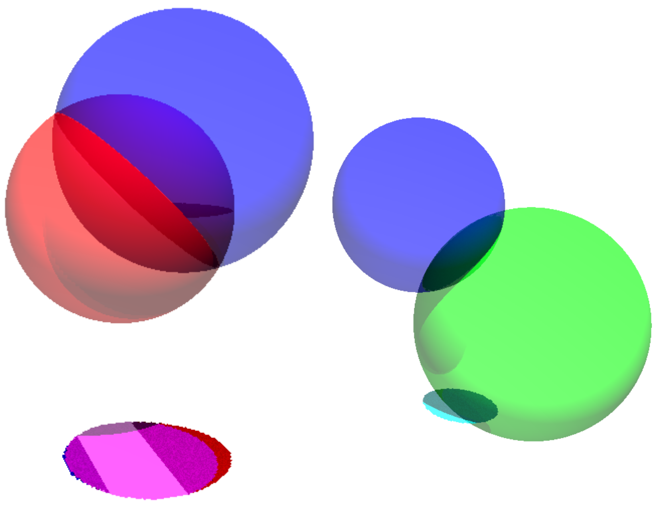

Figure 7: Left: $\pi(E_1 \cap E_2)$ in 3D; Right: $\pi(E_1 \cap E_3)$ in 3D

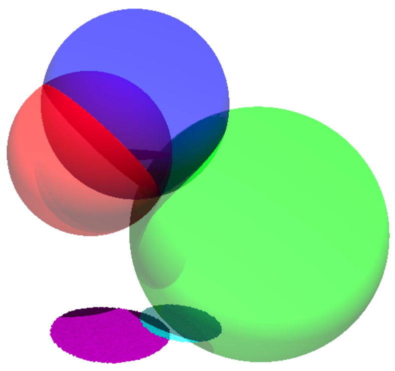

Figure 8: $\pi(E_1 \cap E_2) \cap \pi(E_1 \cap E_3)$ in 3D

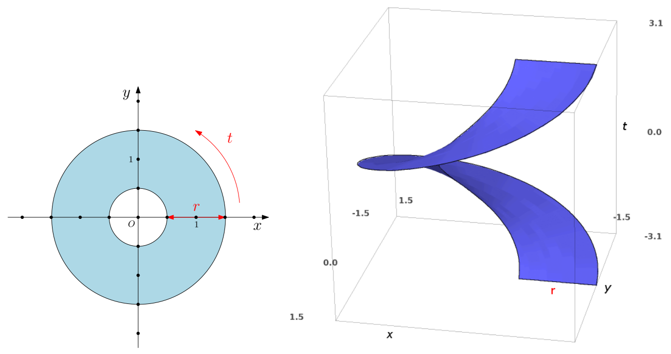

Figure 9: Left: $\pi_{r,t}(R \bowtie X \bowtie Y)$ in 2D; Right: Visualization of $\pi_r(R \bowtie X \bowtie Y)$ in 3D

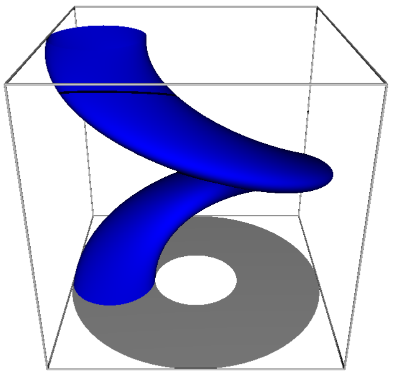

Figure 10: Projection of a 3D cylindrical spiral yields a 2D annulus

6. Implementation

We have implemented the approach presented in this paper in Python/SAGE [27]. The necessary algebraic systems to describe the geometric set are automatically generated and are then passed to Quimper [28] for solving. The expression tree is described by object constructors. For example, given spheres A and B, $\pi(A \cap B)$ is expressed as:

```
x,y,z = SR.var('x,y,z')
A = PrimitiveSet((x-3/4)^2+(y-3/4)^2+(z-3/4)^2<2/3,
   {x:RIF(-2,2), y:RIF(-2,2), z:RIF(-2,2)})
B = PrimitiveSet((x-1/4)^2+(y-1/4)^2+(z-1/4)^2<1,
       {x:RIF(-2,2), y:RIF(-2,2), z:RIF(-2,2)})
G = ProjectionSet(IntersectionSet(A,B), set([z]))
```

The code consists of definitions and is pretty straightforward to read. `RIF` stands for *Real Interval Field*, *i.e.*, it provides a way to denote intervals in SAGE. The output is the DNF expression:

$$\pi(A)|B \cup \pi(A \bowtie B) \cup \pi(B)|A \cup \pi(B \bowtie A).$$

Note that although $\pi(B \bowtie A)$ is identical to $\pi(A \bowtie B)$ it still appears in the expression. We hope to allow for such optimizations in future versions. The four systems generated are:

(i) $f_0 = f_1 = 2z - \frac{3}{2} = 0$ and $s_0 - s_1 \geq 0$
(ii) $f_0 = f_1 = s_0 - s_1 = 0$
(iii) $f_0 = f_1 = 2z - \frac{1}{2} = 0$ and $s_1 - s_0 \geq 0$
(iv) identical to (ii)

with

$$\begin{aligned} f_0 &= \tfrac{1}{16}(4z-3)^2 + \tfrac{1}{16}(4y-3)^2 + \tfrac{1}{16}(4x-3)^2 - s_0 - \tfrac{2}{3} \\ f_1 &= \tfrac{1}{16}(4z-1)^2 + \tfrac{1}{16}(4y-1)^2 + \tfrac{1}{16}(4x-1)^2 - s_1 - 1 \end{aligned}$$

Finally, the systems are solved with Quimper and the results are merged and plotted (Fig. 11 left).

As a second example, we present the code for the projection of the parametric annulus of Sec. 5.3 with respect to t, r, y.

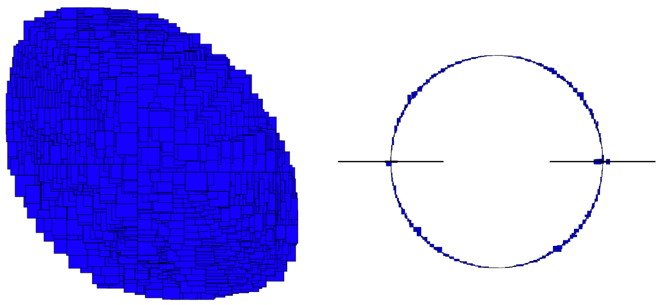

Figure 11: Left: Plot of $\pi(A \cap B)$ in (x, y); Right: Plot of $\pi_{r,t,y}(R \bowtie X \bowtie Y)$

```
x,y,t,r,s = SR.var('x,y,t,r,s')
R = PrimitiveSet(s-r*(r-1),{r:RIF(0,1),s:RIF(-2,2)},s)
X = PrimitiveSet(x - (1/2*cos(t) + r*cos(t)),
    {x:RIF(-2,2),t:RIF(-3.15,3.15),r:RIF(0,1)},None)
Y = PrimitiveSet(y - (1/2*sin(t) + r*sin(t)),
    {y:RIF(-2,2),t:RIF(-3.15,3.15),r:RIF(0,1)},None)
G = ProjectionSet(JoinSetMulti([R,X,Y]), set([t,r,y]))
```

The generated constraints are solved with Quimper and plotted in Fig. 11 right. The plot is shown as a 2D shape, however it represents an 1-dimensional object (the projection along the x-axis). It is evident that the projection is equal to the x range $[-\frac{3}{2}, \frac{3}{2}]$. The y-values shown are contributing points from the higher dimension, *i.e.*, the y-values that correspond to critical values of the characteristic variable s. According to Lem. 1, an additional Jacobian constraint has been taken into account, since the number of variables projected is greater than or equal to the number of terms in the join expression:

$$J = \begin{vmatrix} 0 & 1-2r & 0 \\ (1/2+r)\sin t & -\cos t & 0 \\ -(1/2+r)\cos t & -\sin t & 1 \end{vmatrix} = 0$$

The generated system is:

$$\begin{cases} x-(1/2+r)\cos t &= 0 \\ y-(1/2+r)\sin t &= 0 \\ s-r(r-1) &= 0 \\ J &= 0 \end{cases}$$

With the second formulation, the description of the annulus is much simpler:

```
x,y,t = SR.var('x,y,t')
A = PrimitiveSet((x-cos(t))^2 + (y-sin(t))^2<1/4,
  {x:RIF(-1.5,1.5),y:RIF(-1.5,1.5),t:RIF(-3.15,3.15)})
G = ProjectionSet(A,[t,y])
```

The generated system is

$$\begin{cases} (y-\sin t)^2+(x-\cos t)^2-s-1/4 &= 0 \\ 2y-2\sin t &= 0 \\ 2(x-\cos t)\sin t - 2(y-\sin t)\cos t &= 0 \end{cases}$$

The solution set is the same, but we obtain a smaller system. We observed that our solver managed to provide better convergence in the first case with the bigger system, in slightly more time. This may be explained by the fact that more constraints (equations) are taken into account. Solution strategies of the generated systems are not our focus in this paper and we plan to study them in a future work.

7. Conclusion

We have extended classical CSG constructs with the projection operator. Projections effectively allow us to model parametric solids and their boolean operations, and therefore deal with a greater variety of sets than classical CSG, such as extrusions or sweeps. Each point in the projection is associated with a point in the original space, a property that allows us to perform gradient computations in the projected space by equivalent gradient computations in the original space. Extending the CSG representation is essential to extend geometric and topological algorithms. And since it provides an algebraic representation of the geometric model, formal methods such as interval analysis and proof assistants can be applied in order to provide certification. Moreover, in the domain of computer graphics, our method could be used to directly deal with projections in a 3D world. Note that our framework can handle any analytic function, the only requirement for system generation is differentiability. Therefore it is not limited to only closed-form functions or more restricted families such as polynomial and rational functions.

Although we have made several assumptions regarding the manifolds with respect to the characteristic variable (it should span both interior and exterior parts and it should have critical points in the studied parts) we have seen that it is still possible to capture and describe many sets that arise in practice, especially when they are constructed ground-up from simple primitives. In a preliminary work we studied the simple idea of directly describing the set with a set of equations and inequalities, possibly introducing (nested) optimization problems with the aid of Lagrange multipliers in the case of projections. We discovered that merely considering a set of semi-algebraic equations (to be used with some black-box solver) is not efficient. Consider, for example, n unit disks centered at (x_i, y_i), $i = 1 \dots n$. Construct the $(n+1)\times(n+3)$ system consisting of the n equations $(x-x_i)^2+(y-y_i^2)-1-s_i = 0$, $i = 1 \dots n$ as well as the equation $\prod_{i=1}^{n}(s-s_i) = 0$. Now the zero-set of the system with respect to x, y, s, s_i where $s \leq 0$ describes the union of the n disks. This naive approach yields an $(n+1)\times(n+3)$ system, while the same set can be expressed by concatenating the solutions of n independent equations in 3 variables. Extensive benchmarks have shown that such an approach is non-practical even when state-of-the-art solvers like Quimper are considered. This led us to this paper's approach which manipulates the expression tree and avoids introducing Lagrange multipliers.

As mentioned, our framework allows modeling extrusions and sweeps, since they can be defined as parametric objects. However, for Minkowski sums things are more difficult, as we currently know of no way to model a continuous characteristic function (simply considering the sum of the characteristic variables is not enough). This will be a topic of future research.

Another interesting problem is to study the complement of projections, which boils down to dealing with complements of join sets, as a consequence of De Morgan's rules. The definition of $\neg(A \bowtie B)$ raises interesting questions. For example, let A, B be two spheres. Then $\neg(A \bowtie B) = \neg A \bowtie \neg B$, but there are cases (*e.g.*, when considering $\neg(A \bowtie \neg B)$) where the characteristic variable does not span both interior and exterior parts. An idea is to use some other equation $F_1 - \lambda F_2$ from the pencil of two primitives, instead of $F_1 - F_2$. Regarding projections, we would like to investigate if it is possible to simplify notation, by expressing the extra equations $\frac{\partial f}{\partial x_i} = 0$ as join sets. For example if $A : f(\mathbf{x}; 0) = 0$ then we may write something like $A \bowtie A'_{x_i}$. Finally, we can extend Thm. 5 for the case where $k > n$, although we don't know if it has any practical meaning.

Acknowledgment. This work has been funded by NPRP grant number NPRP 09-906-1-137 from the Qatar National Research Fund (a member of The Qatar Foundation). The statements made herein are solely the responsibility of the authors.

References

[1] A. G. Requicha, Representations for rigid solids: Theory, methods, and systems, ACM Comput. Surv. 12 (4) (1980) 437–464.

[2] A. A. G. Requicha, R. B. Tilove, Mathematical foundations of constructive solid geometry: General topology of closed regular sets, Tech. rep., UR Research (United States) (1978).

[3] G. Elber, M.-S. Kim, Rational bisectors of CSG primitives, in: Symposium on Solid Modeling and Applications, 1999, pp. 159–166.

[4] J. Rossignac, M. O'Connor, SGC: A dimension-independent model for pointsets with internal structures and incomplete boundaries, in: M. Wozny, J. Turner, K. Preiss (Eds.), Geom. Modeling for Product Engineering, North-Holland, 1989.

[5] J. Rossignac, CSG-BRep Duality and Compression, in: Proceedings of the Seventh ACM Symposium on Solid Modeling and Applications, SMA '02, ACM, New York, NY, USA, 2002, pp. 59–59.

[6] Epic Games, Inc., Unreal engine, https://www.unrealengine.com (1998–2014).

[7] Epic Games, Inc., BSP brushes, http://udn.epicgames.com/Two/BspBrushesTutorial.html.

[8] List of Unreal Engine games, http://en.wikipedia.org/wiki/List_of_Unreal_Engine_games (2014).

[9] S. Steuer, Methods for Polygonalization of a Constructive Solid Geometry Description in Web-based Rendering Environments, Diplomarbeit, Ludwig Maximilian University of Munich (2012). URL http://www.pms.ifi.lmu.de/publikationen/diplomarbeiten/Sebastian.Steuer/DA_Sebastian.Steuer.pdf

[10] N. Delanoue, L. Jaulin, B. Cottenceau, Guaranteeing the homotopy type of a set defined by nonlinear inequalities, Reliable computing 13 (5) (2007) 381–398.

[11] N. Delanoue, L. Jaulin, B. Cottenceau, Using interval arithmetic to prove that a set is path-connected, Theoretical Computer Science, Special issue: Real Numbers and Computers 351 (1) (February 2006) 119–128.

[12] M. H. Zaki, S. Tahar, G. Bois, Formal verification of analog and mixed signal designs: A survey, Microelectronics Journal 39 (12) (2008) 1395–1404.

[13] S. Owre, J. M. Rushby, N. Shankar, PVS: A prototype verification system, in: Proceedings of the 11th International Conference on Automated Deduction: Automated Deduction, CADE-11, Springer-Verlag, London, UK, UK, 1992, pp. 748–752.

[14] The Coq development team, The Coq proof assistant reference manual, LogiCal Project, http://coq.inria.fr (2004).

[15] T. C. Hales, A computer verification of the Kepler conjecture, Tech. rep., Proceedings of the International Congress of Mathematicians, Vol. III (Beijing, 2002) (Beijing), Higher Ed (2002).

[16] A. Narkawicz, C. Muñoz, Formal verification of conflict detection algorithms for arbitrary trajectories, Reliable Computing 17 (2012) 209–237.

[17] J. Keyser, S. Krishnan, D. Manocha, Efficient and accurate b-rep generation of low degree sculptured solids using exact arithmetic: II - computation, Computer Aided Geometric Design 16 (9) (1999) 861–882. doi:10.1016/S0167-8396(99)00033-3. URL http://dx.doi.org/10.1016/S0167-8396(99)00033-3

[18] S. Krishnan, D. Manocha, M. Gopi, T. Culver, J. Keyser, BOOLE: A boundary evaluation system for boolean combinations of sculptured solids, Int. J. Comput. Geometry Appl. 11 (1) (2001) 105–144. doi:10.1142/S0218195901000419. URL http://dx.doi.org/10.1142/S0218195901000419

[19] C. Li, S. Pion, C. Yap, Recent progress in exact geometric computation, The Journal of Logic and Algebraic Programming 64 (1) (2005) 85 – 111, practical development of exact real number computation.

[20] N. Tongsiri, Constructive Solid Geometry with Projection: An Approach to Piano Movers' Problem (2006).

[21] V. Shapiro, D. L. Vossler, Construction and optimization of CSG representations, Comp.-Aid. Design 23 (11) (1991) 4–20.

[22] S. Cameron, Approximation hierarchies and s-bounds, in: Symposium on Solid Modeling and Applications, 1991, pp. 129–137.

[23] S. Cameron, C.-K. Yap, Refinement methods for geometric bounds in constructive solid geometry, ACM Trans. Graph. 11 (1) (1992) 12–39.

[24] F. Zizza, Differential forms for constrained max-min problems: Eliminating Lagrange multipliers, The College Mathematics Journal 29 (5) (1998) pp. 387–396.

[25] E. W. Weisstein, Determinant. From MathWorld—A Wolfram Web Resource (2013). URL http://mathworld.wolfram.com/Determinant.html

[26] G. Tzoumas, D. Michelucci, S. Foufou, Extending Constructive Solid Geometry to Projections and Parametric Objects, in: 10th International Symposium on Tools and Methods of Competitive Engineering, Budapest, Hungary, 2014, pp. 707–718.

[27] W. Stein, et al., Sage Mathematics Software (Version 5.0.1), The Sage Development Team, http://www.sagemath.org (2012).

[28] G. Chabert, L. Jaulin, Contractor programming, Artificial Intelligence 173 (11) (2009) 1079 – 1100.