Journées du Groupe de Travail en Modélisation Géométrique 2013, Marseille

Une nouvelle représentation pour les objets géométriques

George Tzoumas, Arnaud Kubicki, Dominique Michelucci et Sebti Foufou

Université de Bourgogne, CNRS LE2I, Dijon

Résumé

Nous présentons une nouvelle représentation pour les objets géométriques et une application de cette représentation. Elle permet de représenter non seulement les opérations booléennes sur des primitives CSG (unions, intersection, complément) mais aussi des opérations plus complexes comme des projections, des sommes de Minkowski et des extrusions. Le système algébrique obtenu est résolu par un solveur utilisant l'arithmétique d'intervalle. Nous montrons comment le solveur peut être optimisé pour cette application spécifique et comment les algorithmes topologiques existants peuvent bénéficier de cette nouvelle représentation, ce qui étend leur portée.

We present a new representation for geometric sets and an application of this representation. It allows us to describe not only boolean operations of CSG primitives (union, intersection, complement), but also more complicated sets such as projections, Minkowski sums and extrusions. The algebraic system is solved by interval arithmetic. We show how the interval solver can be tuned for this particular application and how existing algorithms for topology computation can benefit from the new representation, allowing them to deal with more types of sets.

Mots-clés : Modélisation géométrique, geometric set, interval arithmetic, CSG, projection

1. Introduction

In this work we propose a new representation for geometric sets that is general enough to cover wide variety of sets that appear in geometric modeling such as CSG constructions, projections, extrusions and Minkowski sums. Given a d-dimensional working space $\mathbb{R}^d$, the term *geometric set* refers to some subset A of $\mathbb{R}^d$. A large family of these sets are semi-algebraic sets that can be described by a quantifier free boolean formula with atoms $P < 0$, $P = 0$, $P \in \mathcal{P}$ with $\mathcal{P}$ a finite set of polynomials. These sets can be combined with operators $\neg, \vee, \wedge$ (denoting complement A^C, intersection $\cap$ and union $\cup$), allowing us to describe CSG constructions.

Previous works study topological properties of semi-algebraic sets. In [Col75] an algorithm called *Cylindrical Algebraic Decomposition* is presented that partitions the set into semi-algebraic subsets homeomorphic to open boxes and a triangulation homeomorphic to the semi-algebraic set is computed in a second step. In [SH97] interval analysis and Morse theory are combined to compute the topology of an implicit surface in $\mathbb{R}^3$.

Our motivation comes from Delanoue, Jaulin and Cottenceau's method [DJC07,NDC06]. A a CSG tree is given as input and a simplicial complex homotopic to the geometric set defined by the CSG tree is obtained. The nodes of the CSG tree carry boolean operations : unions and intersections. The leaves of the CSG tree define geometric primitives with an implicit inequation, like $x^2+y^2+z^2-1 \leq 0$ for the unit 3D sphere. Inequations can also use transcendental functions, trigonometric or exp.

The main idea of the DJC method is to recursively subdivide a bounding box of the set E described by the CSG tree, until the studied box B is empty, or B contains a point $S \in E \cap B$ which "sees" all other points p of $E \cap B$: all points of the segment $[S, p]$ belong to $E \cap B$; it is said that the point S is a star for $E \cap B$. Delanoue, Jaulin and Cottenceau remark that if S is a star for two sets $E_1 \cap B$ and $E_2 \cap B$, then it is also a star for $(E_1 \cap E_2) \cap B$ and for $(E_1 \cup E_2) \cap B$; this remark permits to reduce the star test to primitives only : S is a star for $E \cap B$ when S is a star for all primitives intersecting B. Now S is a star for the primitive set $E = \{X \in \mathbb{R}^3 \mid f(X) \leq 0\}$ iff $f(S) < 0$ and there is no solution $X \in B$ to the system :

$$f(X) = 0,\ \nabla f(X) \cdot (X - S) < 0$$

The latter system is solved with interval analysis ; each interval is represented with its two bounds, the lower bound and the upper bound, represented by floating point numbers. The

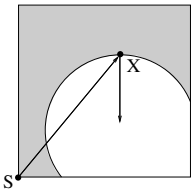


Figure 1: *The point S is not a star for $B \cap E$ (in gray) : $\overrightarrow{SX} \cdot \nabla f(X) < 0$.*

interval solver may output three possible answers : either it proves that there is no such X (thus S is a star), or it finds a counter example X (thus S is not a star, and B must be subdivided), or the solver terminates but can not conclude, due to limitations of interval analysis ; in this case also, the box B is subdivided.

Actually, all boundary faces F (edges for dimension 1, ... hyperfaces for dimension $d-1$) of a studied box B in dimension d must be either empty, or they must contain a point S_F which is a star for $E \cap F$. It may happen that a box which has a star needs to be subdivided because of one of its faces. It is then easy to build a star complex, and Delanoue, Jaulin and Cottenceau proved that this simplicial complex is homotopic to E.

The DJC method takes into account only CSG trees describing "fat sets" ; a fat set is equal to the adherence of its interior ; computationally, we can fill it with full dimensional boxes. Delanoue's method does not terminate, or fails, for CSG trees describing sets which are non fat. This implies also that the method fails when, by accident, the set E is tangent to some subdividing (hyper)plane supporting a face F. Then $F \cap E$ is an isolated point, the coordinates of which do not lie in $\mathbb{F}$, the set of floating point numbers. To avoid that, the initial bounding box can be perturbed with some random noise. Also, it can be a problem if the set E contains parts which are too thin, with width smaller than the resolution of the set of floating point numbers, but in practice the program falls short of memory in such cases.

Another limitation of Delanoue's method is that it does not take into account objects defined by projection. Projections occur for instance with parametric solids, described with : $x = X(u,v,w), y = Y(u,v,w), z = Z(u,v,w)$ and $(u,v,w) \in [0,1]^3$ where X,Y,Z are polynomials or rational functions most of the time. Delanoue's method can output a simplicial complex in the 6D-space x, y, z, u, v, w which is non relevant for us : we want the topology of the projection of this object in the x,y,z space for 3D aplications (or in the x,y space for 2D applications). We call this final, visible, space V (for visible space, or visualization space).

Is it possible to extend Delanoue's method to also account for objects defined by projection ? We studied this problem. It turns out that equations of the star test for the initial DJC method are very simple, but they are no more obvious when projections, projections of intersections, intersections of projections, etc are considered. We realized that a representation of geometric sets is needed to automatically generate systems of equations for the star tests. But which functionalities an ideal representation of geometric sets should provide ?

First, a representation of geometric sets should permit to represent :

- primitives geometric sets, *i.e.* the leaves of CSG trees in CADCAM. These primitives must be fat. In general, it is not a problem in the CADCAM field. A classical example of a 3D primitive set (*i.e.* defined with some polynomial inequality $f(x,y,z) \leq 0$) which is not fat is Whitney umbrella.
- the complement of a set. Remark that, in our context, a set and its complement share their boundary. This feature pertains to the notion of regular sets, and regularization of boolean operations, introduced by A. A. G. Requicha and R. B. Tilove [RT78] in CADCAM.
- the intersection, the union, and the difference of two sets. Remark that the intersection of two fat sets can be a non fat set, in degenerate cases, for example when the two sets are externally tangent to each other. In such case, the interval solver will fail, *i.e.* it will terminate but will not be able to decide if the set is empty or not : it will return an "I do not know" answer.
- the projection of a set, and the projection cylinder of a set (see Figure 3 ; for instance the projection of the 3D sphere $x^2+y^2+z^2-1 \leq 0$ orthogonally to the Oxy plane is a disk ; the projection cylinder of this 3D sphere is a cylinder, it is the volume swept by the sphere, or the circle, when they are moved along the z or the y axis respectively. Note, however,that extending DJC to projections does not need this notion of sweep volumes. Projections are treated in Sec. 4.2.
- the Minkowski sum of two sets. The minkowki sum of A and B is the set of points $a+b$ where $a \in A$ and $b \in B$. See Figure 2.
- the extrusion or sweep of a set, illustrated in 2D in Figure 3. In 3D, each affine transformation (translation, rotation, scaling) is typically represented by a 4×4 matrix, thus the motion is described by an explicit function $t \in [0,1] \rightarrow M(t)$ where t is the time variable and $M(t)$ the affine transformation at time t. Sometimes, extrusion can be described of a Minkowski sum of the trajectory curve and the object. However, we do not use the Minkowski sum, first because the trajectory curve is not a fat set, and second, Minkowski sum does not permit to rotate or scale objects. Remark that the projection cylinder can be represented by an extrusion.

Second, a representation of geometric sets should permit to decide if a set is empty or not. An interval solver will be used for deciding. The interval solver will always terminate. It has three possible outputs : (1) either the set is empty, or (2) it is not empty and a point (sometimes called a witness)

is provided ; actually, the solver can even output a non empty box which is completely included in the set ; or (3) the solver can not decide : for example the set is not fat (it has measure 0), or the accuracy of the underlying floating point arithmetic used by the interval solver is non sufficient, or the solver falls short of memory space because the set contains parts too thin.

Third, a representation of geometric sets should permit to pose constraints on sets : it should permit to specify that some sets are empty, and that some other sets are non empty.

2. Representation of geometric sets

This section presents the principle of our new representation for a first family of geometric sets : primitives, and boolean operations.

Each fat set is represented by a system of equations (i.e., no inequality) and a characteristic variable. In the next paragraph, we consider the case of primitives, of unions of two sets, of intersections of two sets, of differences of two sets. We will not need a special treatment for sets defined by projection, for extending the DJC method. We postpone the representation of projections, Minkowski sums and extrusions to section 4.

We use variables x, y or x, y, z for coordinates in the visible space V. Possibly we use x_1, y_1, z_1, x_2, y_2, z_2 when we need several points in the visible space. Other identifiers (u, v, w, t, or u_i, ..., t_i) are used for the non visible variables.

2.1. Representations of primitives

Each primitive is represented by one *equation* (*i.e.* no inequality) and a characteristic variable. For instance the 3D sphere $x^2+y^2+z^2-1 \leq 0$ is represented with a variable s and the equation : $x^2+y^2+z^2-1-s=0$. The points of the sphere fulfill : $s \leq 0$. The points of the boundary fulfill : $s=0$. The points of the complement of the sphere fulfill $s \geq 0$ (again, remark that the boundary belongs to the sphere and its complement : it is the regularization idea by Tilove and Requicha).

Let $C(E)$ denote the number of equations used for describing a geometric primitive E in the above representation. Clearly $C(E)=1$.

We assume that a bounding box in the visible space for the studied set is known. The interval solver we use needs a bounding interval for each variable. An interval for the characteristic variable s is computed by the evaluation with interval arithmetic : $s = x^2+y^2+z^2-1$. Note that this interval needs not be very sharp.

Second example. The half space primitive $x \leq 0$ is represented with a variable h and the equation : $h-x=0$. The points inside the half space fulfills $h \leq 0$, the points on the boundary fulfil : $h=0$. The points of the complement of the half space fulfill $h \geq 0$. A starting bounding interval for h can be computed from the bounding box of the visible variable x with interval arithmetic.

Here is an example where projections and intersections occur. A thick arc, with given thickness 2ρ, of a parametric curve defined by : $x = f(t) + [-\rho, \rho], 0 \leq t \leq 1$, is represented by the intersection of four primitives : a primitive set for $0 \leq t$, a primitive set for $t \leq 1$, a primitive set for $f(t) - \rho \leq x$, a primitive set for $x \leq f(t) + \rho$. The intersection is treated in Sec.2.3.

2.2. Representation for unions $E = E_1 \cup E_2$

Suppose now we have two geometric objects E_1 with variable e_1, E_2 with variable e_2. E_1 is described with systems of equation $E_1(e_1, x, y, z, \ldots) = 0$, and similarly E_2 is described with systems of equation $E_2(e_2, x, y, z, \ldots) = 0$. Points in E_1 fulfill $e_1 \leq 0$, points in E_2 fulfill $e_2 \leq 0$.

The object $E = E_1 \cup E_2$ is described by a new variable, e, and a system of equation which will guarantee that $e = \min(e_1, e_2)$.

Clearly $e = \min(e_1, e_2)$ is equivalent to maximize e with constraints $e \leq e_1$, $e \leq e_2 \Rightarrow e + \alpha_1^2 = e_1$, $e + \alpha_2^2 = e_2$ for some $\alpha_1 \in \mathbb{R}$, $\alpha_2 \in \mathbb{R}$

Then we consider the Lagrangian :

$$L : e + \lambda_1(e + \alpha_1^2 - e_1) + \lambda_2(e + \alpha_2^2 - e_2)$$

The solution fulfills this $\min(e_1, e_2)$ system :

$$\begin{array}{llll} L'_e : & 1+\lambda_1+\lambda_2 = 0 & L'_{\alpha_1} : & 2\lambda_1\alpha_1 = 0 \\ L_{\lambda_1} : & e+\alpha_1^2-e_1 = 0 & L'_{\alpha_2} : & 2\lambda_2\alpha_2 = 0 \\ L_{\lambda_2} : & e+\alpha_2^2-e_2 = 0 & & \end{array}$$

Indeed, when $e_1 < e_2$, the previous system has the solution $e = e_1$, $\alpha_1^2 = 0$, $\alpha_2^2 = e_2 - e_1$, $\lambda_1 = -1$, $\lambda_2 = 0$. This solution is unique, except for $\alpha_2 = \pm\sqrt{e_2 - e_1}$.

Symmetrically, when $e_2 < e_1$, the system has the solution $e = e_2$, $\alpha_2^2 = 0$, $\alpha_1^2 = e_1 - e_2$, $\lambda_1 = 0$, $\lambda_2 = -1$. This solution is unique, except for $\alpha_1 = \pm\sqrt{e_1 - e_2}$.

The system has both solutions when $e_1 = e_2$.

Thus the set $E = E_1 \cup E_2$ is described by the variable e ; its system of equations is the concatenation of the system for E_1, the system for E_2 and the system equivalent to $e = \min(e_1, e_2)$. Points inside E fulfill $e \leq 0$, points inside the complement of E fulfill $e \geq 0$, points on the boundary of E fulfil $e = 0$.

For the interval solver, we need to initialize a bounding interval for the variable e ; it is $\min(e_1, e_2)$ computed with interval analysis form the bounding intervals from e_1 and e_2. We remind that $\min([a,b],[a',b']) = [\min(a,a'), \min(b,b')]$.

The bounding interval for α_1^2 is $(e_1 - e) \cap [0, +\infty)$, computed again with intervals ; then the bounding interval for α_1 is the square root of the interval for α_1^2. For instance, if $e_1 - e = [-6,4]$, then the interval for α_1^2 is $[-6,4] \cap [0,+\infty) = [0,4]$ and the interval for α_1 is $[0,2]$. We proceed similarly for initializing the bounding intervals for α_2^2 and α_2. Finally, for λ_1 and λ_2, they both lie in the interval $[-1,0]$.

A possible optimization replaces λ_2 by its value $-1-\lambda_1$ and removes the first equation $L'_e = 0$. We also leave to the reader another optimization possible when, inside the studied box, the bounding interval for e_1 is strictly lower than the bounding interval for e_2, or vice-versa.

Assuming that λ_2 has been eliminated, we have that $C(E) = C(E_1) + C(E_2) + 4$.

Now we present an alternative representation for the union. The idea is that e_1, e_2 are considered as roots of a polynomial of degree 2. If Δ is the discriminant of this polynomial, then it holds that $\Delta = (e_2 - e_1)^2$ and if $e = \min(e_1, e_2)$ then $2e = (e_1 + e_2) - \sqrt{\Delta}$ (while if $e = \max(e_1, e_2)$ then $2e = (e_1 + e_2) + \sqrt{\Delta}$.) This leads to the following system :

$$\begin{aligned} \delta^2 - (e_2 - e_1)^2 &= 0 \\ 2e - (e_1 + e_2) + \delta &= 0 \end{aligned}$$

with δ standing for $\sqrt{\Delta}$, therefore $\delta \in [0, \sqrt{\Delta}]$. In this algebraic number approach we have $C(E) = C(E_1) + C(E_2) + 2$, which is more efficient in the number of equations and unknowns introduced, although performing variable elimination in the lagrangian approach may also lead to a similar system.

2.3. The representation for intersections $E = E_1 \cap E_2$

We just replace min in the previous paragraph by max, because $E = E_1 \cap E_2$ will be represented by the variable $e = \max(e_1, e_2)$.

Clearly $e = \max(e_1, e_2)$ is equivalent to minimizing e subject to constraints :

$$e_1 \leq e, e_2 \leq e \Rightarrow e_1 + \alpha_1^2 = e, e_2 + \alpha_2^2 = e$$

Then we consider the Lagrangian :

$$L = e + \lambda_1(e_1 + \alpha_1^2 - e) + \lambda_2(e_2 + \alpha_2^2 - e)$$

We know the solution fulfills this $\max(e_1, e_2)$ system :

$$\begin{array}{llll} L'_e : & 1 - \lambda_1 - \lambda_2 = 0 & L'_{\alpha_1} : & 2\lambda_1\alpha_1 = 0 \\ L_{\lambda_1} : & e_1 + \alpha_1^2 - e = 0 & L'_{\alpha_2} : & 2\lambda_2\alpha_2 = 0 \\ L_{\lambda_2} : & e_2 + \alpha_2^2 - e = 0 & & \end{array}$$

When $e_1 < e_2$, the previous system has the solution $e = e_2, \alpha_1^2 = e_2 - e_1, \alpha_2 = 0, \lambda_1 = 0, \lambda_2 = 1$.

Symmetrically, when $e_2 < e_1$, the previous system has the solution $e = e_1, \alpha_1^2 = 0, \alpha_2^2 = e_1 - e_2, \lambda_1 = 1, \lambda_2 = 0$.

The system has both solutions when $e_1 = e_2$.

Thus the set $E = E_1 \cap E_2$ is described by the variable e ; its system of equations is the concatenation of the system for E_1, the system for E_2 and the system equivalent to $e = \max(e_1, e_2)$. Points inside E fulfil $e \leq 0$, points inside the complement of E fulfill $e \geq 0$, points on the boundary of E fulfil $e = 0$.

The initialization of intervals for e, $\alpha_1, \alpha_2, \lambda_1, \lambda_2$ is very similar to the previous case of union. The interval for e is $\max(e_1, e_2)$, computed from intervals for e_1 and e_2. Remind that $\max([a,b],[a',b'])$ is $[\max(a,a'), \max(b,b')]$. The interval for α_1^2 is the interval $(e - e_1) \cap [0, +\infty)$, etc. This time λ_1 and λ_2 both lie in $[0,1]$.

Optimizations : A first possible optimization replaces λ_2 by its value $1 - \lambda_1$. Again, if the interval for e_1 and e_2 are disjoint, some straightforward optimization and simplification are possible and left to the reader. The second formulation with the discriminant differs in only one sign compared to the union : we just replace the second equation with $2e - (e_1 + e_2) - \delta = 0$.

2.4. The representation for the complement

Let E_1 be a set defined with the system of equations $E_1(e_1, X) = 0$ and having characteristic variable e_1. To define the complement $\overline{E_1}$ of the set E_1, we introduce a new variable $\overline{e_1}$, which will be the characteristic variable of $\overline{E_1}$, and the system of equations defining $\overline{E_1}$ is the concatenation of the system $E_1(e_1, X) = 0$ and of the new equation : $e_1 + \overline{e_1} = 0$. It holds that $C(\overline{E_1}) = C(E_1) + 1$.

Variant : replace e_1 in the definition of E_1 with $-\overline{e_1}$ to get the definition of $\overline{E_1}$. In this case we have $C(\overline{E_1}) = C(E_1)$.

2.5. The representation for differences $E = E_1 - E_2$

The representation of $E = E_1 - E_2$ is left to the reader : $E = E_1 - E_2$ is just $E_1 \cap (\overline{E_2})$. For the lagrangian approach we have that $C(E) = C(E_1) + C(\overline{E_2}) + 4 = C(E_1) + C(E_2) + 5$, while for the algebraic number approach we have that $C(E) = C(E_1) + C(E_2) + 3$.

3. Using the equations based representation to formulate predicates about sets

Let E be a set defined by its variable e and a system of equations $E(X) = 0$. X is the set of all unknowns, of course e is one of the variables in X. Space variables x, y in 2D (x, y, z in 3D), *i.e.* coordinates in the visible space V are variables in X.

We will follow DJC method, except that we recursively subdivide in the visible space. For example, if we are interested in the topology of a 3D objects the points of which have coordinates x, y, z, but there are other variables $u, v, w, \ldots$ in

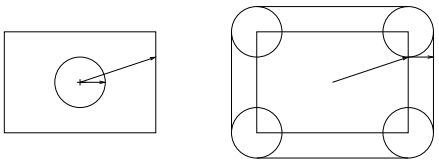


Figure 2: *The Minkowski sum of a disk and a rectangle.*

the system which defines E, we will recursively subdivide a 3D box in the x,y,z space. To find a point S in E, even if coordinates x,y,z of S are given (since we try the center and the vertices of the box, as DJC), the other variables $u,v,w,\ldots$ are unknown. Of course, the interval solver, which is called as a subroutine by our extension of DJC, will probably subdivide along the dimensions $u,v,w,\ldots$.

So we solve with interval analysis the system $E(X) = 0, e \leq 0$. If we want only equations, we can solve $E(X) = 0, e+\gamma^2 = 0$ where γ is a new, auxiliary, real unknown.

Remark : instead of imposing $e \leq 0$, we can rather compute the smallest e such that $E(X) = 0$. The solver will give a list of boxes enclosing local minima. If the interval of the smallest minima is completely negative, we are sure that E is non empty. If the interval of the smallest minima is completely positive, we are sure that E is empty. If the interval of the smallest minima contains 0, we can not decide. We will subdivide the studied box.

When we know a potential star $S \in E$, we need to prove that it is a star, or that it is not. We define the set of counter examples (which prove that S is not a star) as follows : we need 2 points X_1 and X_2 inside the studied box B, such that : S, X_1, X_2 are aligned (in this order) in the visible space, X_1 is outside E (thus inside the complement of E), and X_2 is inside E. The system of equations straightforwardly follows. We solve the system with some interval solver. If there is no counter example, then S is a star. If the solver does not know, or provides a counter example, then we can not prove that S is a star, so we subdivide the box.

4. Representation for some other sets

4.1. The Minkowski sum

The Minkowski sum of two 3D sets E_1 and E_2 is the set of points $E = \{(x,y,z) = (x_1+x_2, y_1+y_2, z_1+z_2) \mid (x_1,y_1,z_1) \in E_1, (x_2,y_2,z_2) \in E_2\}$ When E_1 is defined with the variable e_1 and the system of equations E_1, and similarly for E_2, then the set $E = E_1 \oplus E_2$ is defined with a new variable e, the concatenation of systems for $X_1 \in E_1$, for $X_2 \in E_2$, and $x_1+x_2-x = y_1+y_2-y = z_1+z_2-z = 0$. Moreover $e = \min(e_1+e_2)$. With this definition, the boundary of E is characterized by $e = 0$; points strictly inside E are characterized by $e < 0$; points strictly outside E are characterized by $e > 0$. This is not quite true however, since e may attain its minimum value with $e_1 e_2 < 0$. Therefore we impose an extra constraint that $e_1 e_2 \geq 0$. It is still an open problem to express e with a single symmetric expression that can be evaluated efficiently.

Now we present the corresponding system for the minimization problem. Let E_0 be the Minkowski sum of two geometric sets E_1 and E_2 in dimension d. That is

$$\mathbf{z} \in E_0 \iff \exists \mathbf{x},\mathbf{y},\mathbf{z} \in \mathbb{R}^d : \mathbf{z} = \mathbf{x}+\mathbf{y} \wedge \mathbf{x} \in E_1 \wedge \mathbf{y} \in E_2.$$

Let $\mathcal{S}_i$ be the ordered set of equations used in the description of geometric set E_i, $i = 0,1,2$. Let $f_k(\mathcal{S}_i)$ denote the k-th equation of $\mathcal{S}_i$. Let $\mathcal{V}_i$ be the set of variables in $\mathcal{S}_i$, $i = 0,1,2$. In general, $|\mathcal{V}_i| = |\mathcal{S}_i| + d$. Let $s_i \in \mathcal{V}_i$ be the slack variable of set E_i. We want to minimize $s_1 + s_2$. To do so we shall use the Fritz John conditions along with a normalization condition. Fritz John conditions allow for inequality constraints and also handle the case where the constraints are linearly dependent at the minimum. The normalization condition provides trivial bounds for the multipliers. †

The John conditions can be generated with the aid of the following quantity J, similar to a Lagrangian : $J := \mu_0(s_1+s_2) + \mu_1 s_1 + \mu_2 s_2 + \sum_{i=1}^{|\mathcal{S}_1|} u_i f_i(\mathcal{S}_1) + \sum_{i=1}^{|\mathcal{S}_2|} v_i f_i(\mathcal{S}_2) + \mathbf{w}\cdot(\mathbf{z}-\mathbf{x}-\mathbf{y})$

Now $\mathcal{S}_0$ contains $\frac{\partial J}{\partial a}$, $\forall a \in \mathcal{V}_i, i = 1,2$ ($|\mathcal{S}_1| + |\mathcal{S}_2| + 2d$ equations in general), as well as the following equations :

$$\begin{aligned}
z_i - x_i - y_i &= 0, \quad i = 1,\ldots,d \\
\mu_i s_i &= 0, \quad i = 1,2 \\
f_i(\mathcal{S}_1) &= 0, \quad i = 1,\ldots,|\mathcal{S}_1| \\
f_i(\mathcal{S}_2) &= 0, \quad i = 1,\ldots,|\mathcal{S}_2| \\
\mu_0+\mu_1+\mu_2+\sum_{i=1}^{d} w_i^2 + \sum_{i=1}^{|\mathcal{S}_1|} u_i^2 + \sum_{i=1}^{|\mathcal{S}_2|} v_i^2 &= 1, \quad \mu_i \geq 0 \\
s_0 - s_1 - s_2 &= 0,
\end{aligned}$$

with the normalization condition implying that $\mu_i \in [0,1]$ and $w_i, u_i, v_i \in [-1,1]$. It also follows that $C(E_0) = 2C(E_1) + 2C(E_2) + 3d + 4$.

4.2. The projection of a set

We present the projection through an example. Let E be the 3D set $\{(x,y,z) \mid E(x,y,z,e) = 0, e \leq 0\}$. We consider the set E_p which is the projection of E orthogonally to the plane Oxy. In other words, $E_p = \{(x,y) \mid \exists z, E(x,y,z,e) = 0, e \leq 0\}$. It can be formulated as a minimization problem : (x,y) lies in E_p if for the z which minimizes e for this x,y, it holds that e is negative or 0. Thus the system for E_p is the concatenation of the system which defines E, plus the system for : $e_p = \min e$, which is managed as in section **??** ; the variable e_p is the variable which characterizes E_p ; note

†. One should also check that inequalities apply ! In our case this is handled by the bounds on s_i.

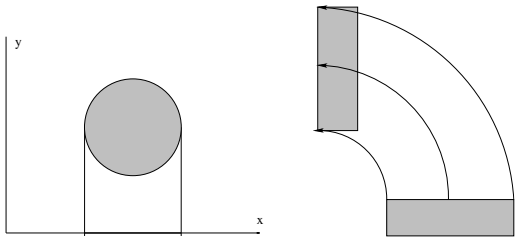


Figure 3: *Left : a projection of a disk. Right : an extrusion of a rectangle.*

that x, y are "given" parameters (they are coordinates in Y, in the notation of section **??**), and the minimization is done on z.

This projection "forgets" the coordinate z ; for sets parametrized with parameters $u_1, u_2, ...u_p$, we want to "forget" all parameters u_i ; the generalization is left to the reader.

4.3. The extrusion

Figure 3 illustrates an extrusion of a 2D rectangle.

Let E be a set defined by the system $E(e,X) = 0$ and the variable e. Let $M(t), 0 \leq t \leq 1$ be the matrix of an affine transformation, at time t. For example, for rotating around the z axis :

$$M(t) = \begin{pmatrix} \cos(2\pi t) & \sin(2\pi t) & 0 & 0 \\ -\sin(2\pi t) & \cos(2\pi t) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

E is extruded, or swept, along $M(t)$. It gives the set E'. The points x', y', z' of E' are defined by $(x', y', z,' 1) = (x, y, z, 1)M(t)$. The variable e' characterizing E' is $\min e$ such that $(x', y', z,' 1) = (x, y, z, 1)M(t)$.

Here E was moved basically along a curve (the time parameter t in $[0, 1]$). It is possible to move E along a surface patch, parametrized by two parameters u and v, or inside a volume parametrized by three parameters u, v, w. It suffices to concatenate the definition of this parameter space, to get the full definition of E'.

5. Implementation

Using SAGE [S$^+$12], we have implemented the proposed modeling of geometric sets. The advantage of SAGE is that it provides a very powerful programming environment through *Python*, with interfaces to various open source scientific libraries. We implemented generation of the corresponding algebraic systems which are then solved (more precisely, a covering of the solution space is computed until a specified resolution is reached) using :

(a) a naive-solver that performs search in $\mathbb{R}^d$

(b) Quimper [CJ09]

(c) an Interval-Newton solver written in C++ [Kub12]

Solver (a) provides a testbed for ideas and experiments and was implemented within SAGE. We implemented four variations : (a_0) naive search in full $\mathbb{R}^d$, (a_1) full search with a trivial contractor for algebraic expressions, (a_2) subdivision in visible space along with the trivial contractor for other dimensions, (a_3) reduced-dimension search exploiting dependencies between variables which are indicated *a priori*.

Solvers (b) and (c) are used as external programs, called within our script. We also added support for graph visualization and analysis in order to perform the DM-decomposition [AaJM93] via the corresponding bipartite graph or partitioning through the variable graph [WBL09] (we aim to investigate the use of such techniques in future work).

We performed benchmarks concerning the following geometric primitives :

set	equation	type	$x \times y$
E_1	$\frac{1}{3}x^2 + y^2 < 1$	elliptic disk	$[-2,2] \times [-2,2]$
E_2	$(x - \frac{3}{4})^2 + (y - \frac{3}{4})^2 < \frac{2}{3}$	disk	$[-2,2] \times [-2,2]$
E_3	$\frac{1}{5}x^2 + y^2 < \frac{2}{5}$	elliptic disk	$[-2,2] \times [-3,1]$
E_4	$x^2 + y^2 < \frac{1}{5}$	disk	$[-2,2] \times [-1,1]$
E_5	$\frac{1}{5}x^2 + (y + \frac{1}{2})^2 < \frac{2}{5}$	disk	$[-2,2] \times [-2,2]$

These primitives were used to describe the following geometric sets :

	set	description	sys. size
(i)	E_1	simple primitive	1×3
(ii)	$\overline{E_1}$	complement	2×4
(iii)	$E_1 \cap E_2$	intersection (Lagrangian)	6×8
(iv)	$E_1 \cap E_2$	intersection (algebraic number)	4×6
(v)	$(E_3 - E_4) \cap E_5$	more complex set	8×10
(vi)	$E_3 \oplus E_4$	Minkowski sum ; x for E_4 is $[-1,1]$	14×16

which are then solved using solvers (a), (b) and (c), as shown in Fig. 4, 5 and 6. The top two rows show the covering of the geometric set with precision 0.1, as computed by solvers a_0, a_1, a_2, a_3, b, c in this order. For convenience, only the visible space (x, y) is displayed (which is often what we are interested in), however the dimension is much bigger, *e.g.* the Minkowski sum of two primitives involves 16 variables, compared to only 3 for a single primitive. Red boxes show rejected boxes, blue boxes show non-rejected boxes, and green boxes show boxes that are guaranteed to contain at least one solution. This is currently implemented only for solver (a) with a simple Miranda test [Vra89], but we aim to improve the test in the future, so that all boxes within the interior of a set are green and only boxes on the boundary are blue. The leftmost figure of the bottom row displays runtimes with varying precision. Bars in each cluster correspond to the six benchmarked solvers. In the middle figure of the bottom row, the coresponding bipartite graph is displayed [AaJM93] : One vertex for each equation and one vertex for each variable is created, with an edge joining an equation with a variable iff the equation contains that variable. The rightmost graph is the variable graph [WBL09] : Each vertex represents a variable and two vertices are connected iff the corresponding variables appear together in some equation.

We set a maximum run-time limit of 10 minutes. From the benchmarks we can observe the following properties.

For sets (i) and (ii) (simple primitives and their complements), solvers (b) and (c) are very efficient. When high precision is requested, (c) appears to be slightly more efficient than (b). Then variation (a_3) follows, which in fact hard-codes the dependency of the slack variable on the visible space ((x,y) in our examples). The difference in runtime is due to the fact that the implementation of (a_3) is in *Python*, while (b) and (c) are in C++.

For (iii) and (iv) (intersections), the number of unknowns increases. For (iv), we notice a remarkable performance of (b) contrary to the competition. For (iii) which is using the lagrangian approach, the runtime of (b) increases by an order of magnitude, and the number of unknowns rises from 6 to 8. We also observe that the naive 8-dimensional search of (a_0) timed out, while (a_2) and (a_3) perform very well, which suggests that focusing on the visible space and/or exploiting dependencies between variables may be the way to go. However, the current version of (a_2) is still experimental, as it may fail to reject empty boxes for Minkowski sum computations (vi).

For the complex set (v), again (b) outperforms all other solvers. Solver (c) timed out for the first time (with 10 unknowns).

Finally, for the Minkowski sum (vi), only (b) managed to run accurately within time, (a_2) performed relatively well, but failed to reject many boxes.

Solver (c) looks promising but needs to be extended so that it exploits the dependencies between variables in order to be efficient in high dimensions.

Finally, we would like to mention that we have not treated yet an equally important problem : testing the emptyness of a set. For this problem, one may follow two approaches :

- Modify existing solvers to stop when a solution is encountered.
- Augment the under-constrained system in a way that it becomes well-constrained and then solve the new system. This can be done for example by computing a point in the solution set that lies closest to the origin, by solving an optimization problem.

6. Conclusion

The DJC method initially applies only to boolean combinations of primitive sets. This new representation of geometric sets permits to extend significantly and easily the scope of DJC : it now applies to any combination of boolean operations, projections, Minkowski sums, extrusions, deformations. The new representation of geometric sets likely permits to extend the scope of other geometric algorithms. Its main feature is to provide a language for specifying constraints about geometric sets, and for interrogating geometric sets. Interval analysis is used to solve the automatically generated systems of equations. Indeed, with this new representation, it seems that numerous geometric problems are reduced to solving a system of equations, or solving a system of optimization problems. Regarding optimization problems, we are still working on open problems such as the number of constraints that have to be taken into account, whether nested optimizations can be avoided, whether bounds for Lagrange multipliers can be computed efficiently and finally, whether it is possible to completely avoid Lagrangians by expressing optimality in other ways.

The implications for interval solvers.

First of all, the examples are in accordance with the theory and confirm that the modeling through the proposed algebraic systems is correct. Current benchmarks suggest that the built-in contractor of Quimper (b) performs amazingly well and at the same time, we might benefit if we exploit dependencies between variables. Thus future work includes :

- Using Bernstein-based solvers in the hope that they will produce more efficient pruning of boxes. In this direction we have already implemented a parser for multivariate polynomials to be used along with [Dje12] (which concerns an efficient LP solver for sparse problems) towards an efficient implementation of [FMF10].
- Applying techniques in [AaJM93] and [WBL09] in order to exploit dependencies between variables and the fact that the generated systems are rather sparse (each equation contains only a few unknowns, compared to the total number of unknowns). In [WBL09] a partitioning technique has been successfully used to attack cryptanalysis problems.
- Implementing the emptiness test. This can be done as mentioned either by stopping at the first solution encountered, or by solving an appropriate well-constrained system (cf. Sec. 5).
- Implementing second derivative tests for generic optimization problems (though not required so far).

Improvements of DJC.

The DJC method provides a simplicial complex homotopic to the studied set. The Hausdorff distance between the studied set and the simplicial complex can be large ; for instance, a simplicial complex made of a single vertex is homotopic to any (topological) ball. It seems possible to modify DJC, to account for some threshold on the Hausdorff distance. Another question is : is it possible to get a simplicial complex isotopic (instead of just homotopic) to the studied set ?

What about the fat set constraint ? For instance a square in the plane, minus a circle (which is not fat set) inside the square, is a fat set, with two components (the inside disk, and the outside). Further, is it possible to account for pieces of

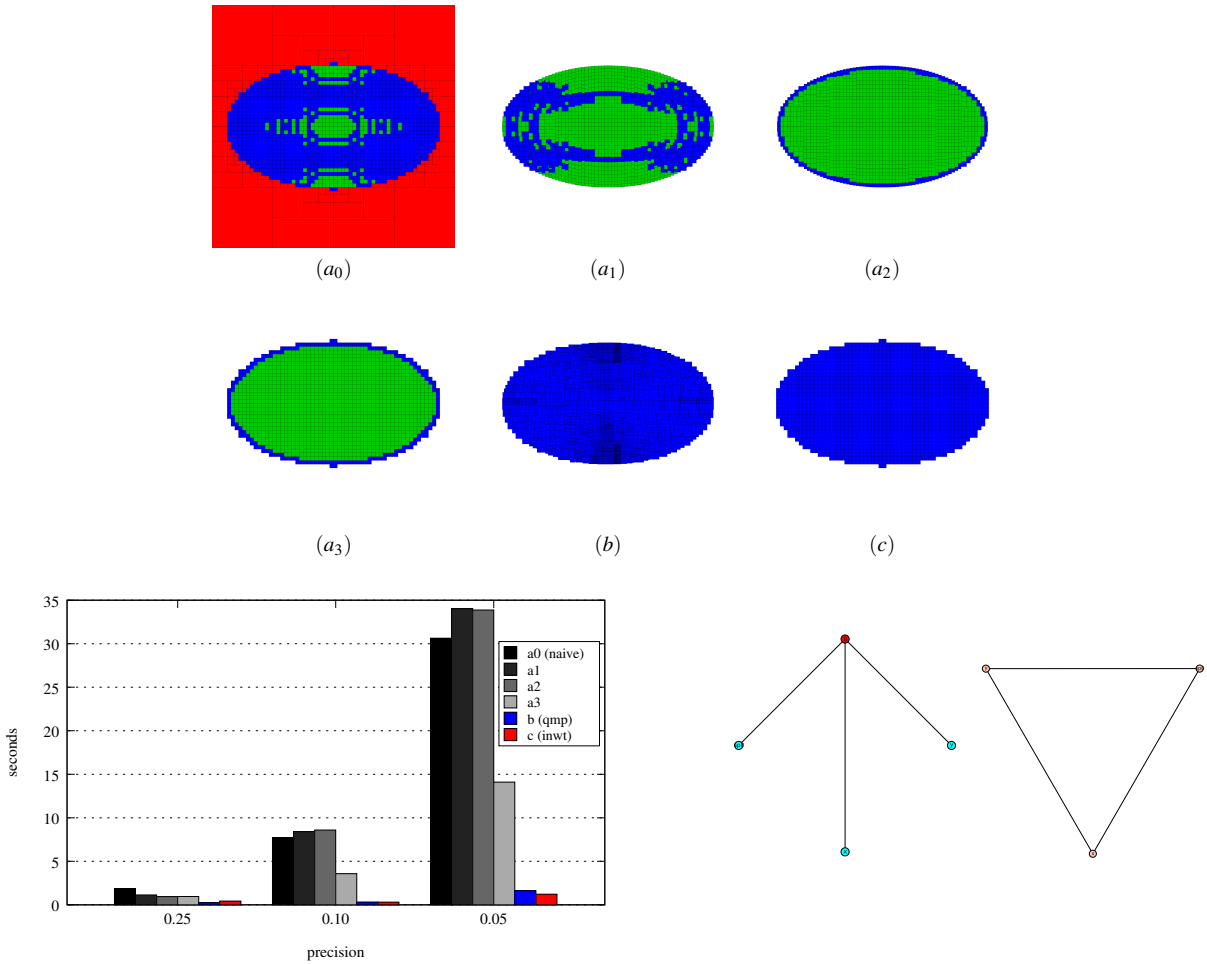


Figure 4: E_1

boundary : parametric arcs of 2D curves in 2D, or parametric surface patches in 3D ? We would like to study the topology imposed by a set of boundary pieces : for instance do they delimit a boundary or not ? It should be used to prove that a given Brep (a Boundary Representation) is valid.

Acknowledgements

The authors want to thank Vishal Donderia for useful discussions during his internship. This research work, especially Vishal Donderia's internship and George Tzoumas' current postdoc in Dijon, have been funded by NPRP grant number NPRP 09-906-1-137 from the Qatar National Research Fund (a member of The Qatar Foundation).

Références

[AaJM93] Samy Ait-aoudia, Roland Jegou, and Dominique Michelucci. Reduction of constraint systems. In *Compugraphics*, pages 83–92, Alvor, Portugal, 1993.

[CJ09] Gilles Chabert and Luc Jaulin. Contractor programming. *Artificial Intelligence*, 173(11) :1079 – 1100, 2009.

[Col75] G. E. Collins. Quantifier elimination for real closed fields by cylindrical algebraic decomposition. In *Proc. 2nd GI Conference on Automata Theory and Formal Languages*, volume 33 of *Lecture Notes Comput. Sci.*, pages 134–183. Springer-Verlag, 1975.

[DJC07] N. Delanoue, L. Jaulin, and Bertrand Cottenceau. Guaranteeing the homotopy type of a set defined by nonlinear inequalities. *Reliable computing*, 13(5) :381–398, 2007.

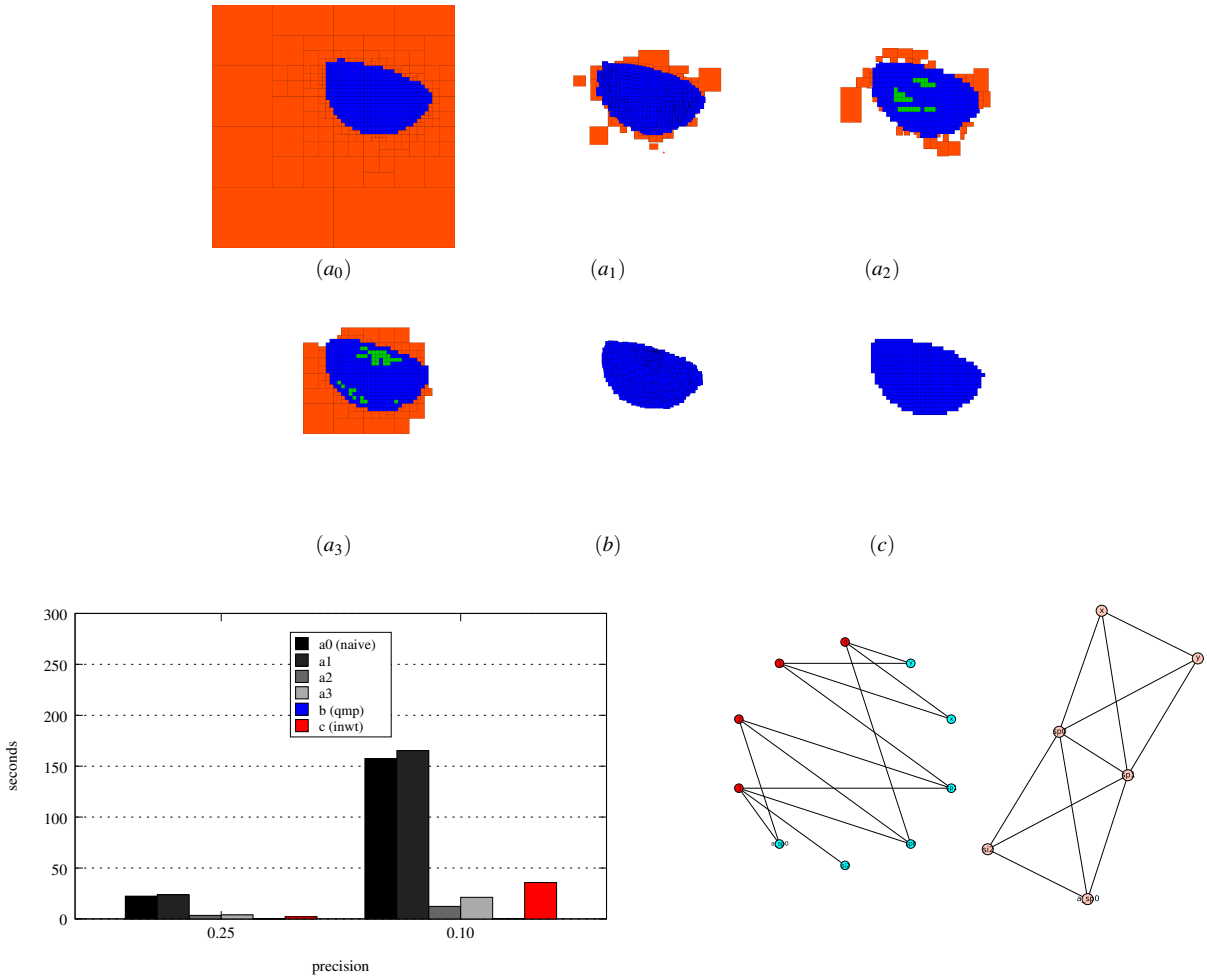


Figure 5: $E_1 \cap E_2$ *(algebraic number system)*

[Dje12] Mahfoud Djedaini. Internship report. Technical report, Quatar University, 2012.

[FMF10] Christoph Fünfzig, Dominique Michelucci, and Sebti Foufou. Optimizations for tensorial Bernstein-based solvers by using polyhedral bounds. *International Journal of Shape Modeling*, 16(01n02) :109–128, 2010.

[Kub12] A. Kubicky. *Interval Newton solver in C++*. University of Burgundy, 2012.

[NDC06] L. Jaulin N. Delanoue and B. Cottenceau. Using interval arithmetic to prove that a set is path-connected. *Theoretical Computer Science, Special issue : Real Numbers and Computers*, 351(1) :119–128, February 2006.

[RT78] Aristides A. G. Requicha and Robert B. Tilove. Mathematical foundations of constructive solid geometry : General topology of closed regular sets. Technical report, UR Research [http ://dspace.lib.rochester.edu/oai/request] (United States), 1978.

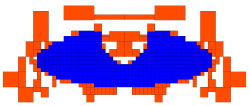
[S+12] W. A. Stein et al. *Sage Mathematics Software (Version 5.0.1)*. The Sage Development Team, 2012. http ://www.sagemath.org.

[SH97] Barton T. Stander and John C. Hart. Guaranteeing the topology of an implicit surface polygonization for interactive modeling. pages 279–286, 1997.

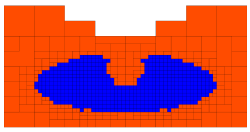
[Vra89] Michael N. Vrahatis. A Short Proof and a Generalization of Miranda's Existence Theorem. *Proceedings of the American Mathematical Society*, 107(3) :pp. 701–703, 1989.

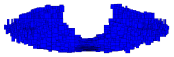
[WBL09] Kenneth Koon-Ho Wong, Gregory V. Bard, and Robert H. Lewis. Partitioning multivariate polynomial equations via vertex separators for algebraic cryptanaly-

(timeout) (timeout)



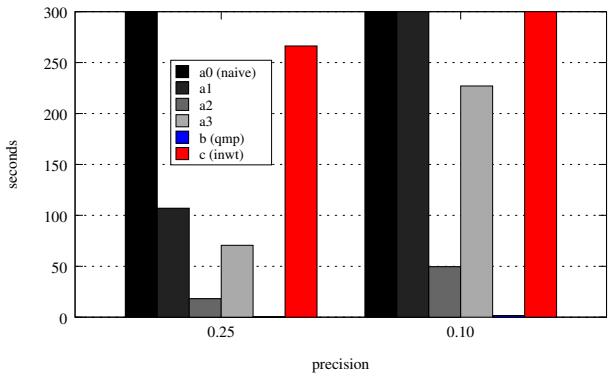
(a_0) (a_1) (a_2)

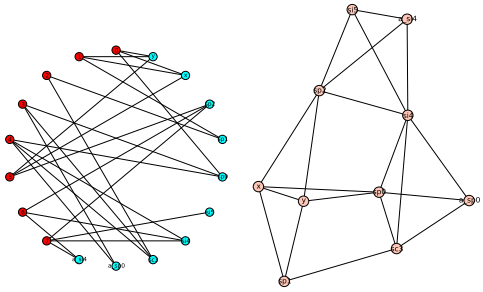




(timeout)

(a_3) (b) (c)





8 equations $\times$ 10 unknowns

Figure 6: $(E_3 - E_4) \cap E_5$

sis and mathematical applications. Cryptology ePrint Archive, Report 2009/343, 2009. http ://eprint.iacr.org/.