– Computational geometry –
From theory to practice,
From linear objects to curved objects.

Monique Teillaud

HABILITATION À DIRIGER DES RECHERCHES
Université de Nice Sophia Antipolis
École doctorale STIC

Defended on September 25, 2007

Reviewers:
Bernard Chazelle
Daniel Lazard
Chee Yap

Members of the Defense Committee:
André Galligo (President)
Jean-Daniel Boissonnat
Daniel Lazard
Jean-Michel Moreau
Günter Rote
Chee Yap

Contents

Introduction

Whereas the international community of computational geometry is often tempted to plunge into essentially theoretical research, in particular of a combinatorial nature, the main originality of the work at INRIA resided, already when I started, in the concern of experimental validation and applicability.
The field globally evolved in this direction, in particular after the *Impact Task Force Report* [C+96, C+99]. But our lead on this topic still stays: our interest for technologic and industrial transfer, and for setting up a platform for research, became even more concrete with our strong implication in the CGAL project [CGA] for which our group is one of the leaders.
This document choses to present the work from the point of view of this practical concern.

The long chain, that leads algorithms to largely distributed and used software

The list of ingredients that allow to obtain a library of the quality and impact of CGAL is long. They are all important, none of them must be minimized.

Nothing can be done without a good **mathematical formalism**, that forms the foundations, essential to the structure. The geometric data structures, such as Delaunay triangulations, already stand on good bases and their mathematical properties are known. A few new studies still must be often performed to help the adaptation of these bases to a rigorous programming.

Then, an **algorithmic and combinatorial** study of the geometric structures to be computed, of the data structures and the complexity of the algorithms is necessary. It is well known that the best algorithms in theory are not always possible to code, and even when they are, they are not always the best in practice. Still, an algorithm that is easy to code and is getting good running times in practice must be validated by a theoretical study that will allow to certify its behavior in the worst case or in an expected case, its capacity of scaling, and sometimes to correctly define its conditions of use.

The choices of **representation** of the objects and the geometric structures are mostly guided by the combinatorial study. They often result from a compromise between acces

time, memory space, richness of offered functionalities. These choices are an important preliminary when settling geometric software.

Robustness issues appear quickly. They can be divided into two main categories. Handling **degenerate cases** cannot be neglected, since real data provided by industrial partners very often contain cases like coplanar or cospherical 3D points. Dealing with **numerical issues** is perhaps even more crucial, since rounding errors can very frequently make programs to crash [KMP+04]. We work here in the model of exact geometric computation, in which geometric predicates, that are the tests on which the algorithms base their decisions, are in particular evaluated in an exact way (see Section 1.4).

Efficiency of programs complete the set. For obvious reasons, a program will never be used in a company if it does not have good performances, whatever its stabibilty may be. Filtering methods, both of arithmetic and geometric natures, allow more and more often the exact geometric computation to get performances comparing with those of non certified computation.

Of course, the quality of the programming itself has a large influence on the quality of the final software. CGAL additionnally demands code genericity, reusability, flexibility. To answer this demand, the software **design** choices must be done in a very careful way.

CGAL

I won't spend time here on the technical aspects of C++ programming, of which I am certainly not the best specialist. basic ideas can be found in the following chapter:

> [44] Generic programming and the CGAL library.
> In Jean-Daniel Boissonnat and Monique Teillaud, editors, *Effective Computational Geometry for Curves and Surfaces*, pages 313–320. Springer-Verlag, Mathematics and Visualization, 2006.
> Avec Efi Fogel.

I only describe here in a few lines the CGAL specifities. The project site[1] gives more information. The CGAL project formally started in 1996 under the initiative of a consortium of a few European teams (including our group) and was supported by the European Community for three years. It evolved to an *Open Source* project, allowing other researchers to participate. An INRIA start-up, GeometryFactory[2], was created in january 2003, and is selling commercial licenses, support and specific developments based on CGAL.
The project goal is to promote research in computational geometry and to translate results into robust programs for industrial applications.

[1] `www.cgal.org`
[2] `http://www.geometryfactory.com/`

At the same period, the *Computational Geometry Impact Task Force Report* coordinated by Bernard Chazelle insisted on a few recommendations [C+96, C+99]. The production of useful and usable geometric software was a key recommendation, and came with the need for creating a rewarding structure for implementations in the academic world. CGAL answers these two recommendations as well as possible.
CGAL shows a strong demand in terms of quality, and quality is ensured by different means. A complete manual (more than 3000 pages now) is available on-line. Tests are running every night on internal versions. Finally, an editorial board was crated in 2001; we are currently 12 persons in the committee (two of them were recently elected, on july 2nd, 2007) in charge of making technical decisions and coordinating CGAL promotion; the committee evaluates new submitted packages and reviews them, in a process that is similar to the submission of papers to journals and conferences. In this way, the committee plays an important role not only in the quality of the library, but also in the reward of the packages authors, who can take advantage of having gone through this deep review process before being able to "publish" their package in CGAL.
CGAL hides neither the merits nor the work on its contributors by attributing them to some mysterious superior entity; on the contrary, the important visibility of CGAL reflects on the project participants. The gratitude of the community for the contributors is also sensitive in many occasions.

CGAL qualifies as a première in a community that is originally mostly theoretical. The impact of CGAL is difficult to measure, due to its principal means of distribution, but all clues (roughly 1000 downloads per month, more than 800 registered people on the public mailing list, and an important activity on this list) let us think that this impact is huge, both in the community and even more outside.
This impact somehow legitimates research in computational geometry, by proving that it does not boil down to an intellectual game of a handful of researchers, and it shows that the small size of the community (the field is sometimes called "pin head" by sceptical colleagues) must be relativized against its impact.

CGAL, Research or not?

If programming, in particular programming in CGAL with its strong demands, does not pay in terms of publications, keeping this question in mind as time goes allows to sharpen guidelines that naturally integrate the CGAL work in a research framework.

CGAL **is useful for research.** many research works are possible because some basic CGAL packages exist, like the 3D triangulation package that is widely used for research in meshing and reconstruction, like for instance in the following two references taken from a large set [DG01, ORY05].

CGAL **raises new questions.** Some questions would obviously never have been raised without the work on libraries such as CGAL and CORE. In particular, many studies on

robustness, on the arithmetic side [Pio99, BEPP97, PY06] as on a more algorithmic side (see Section 1.4) would never have been done. Some work on simpler and more efficient algorithms is also motivated by programming.

CGAL **reconciles C++ concepts and mathematic concepts.** This is the thesis I am supporting in particular. This conceptualization is the aspect that makes the CGAL work really interesting, and objectively infinitely more interesting than just writing programs that work. Of course this work does not come without efforts, efforts both in terms of research (see for instance Section 2.3) and in terms of communication and persuasion.

Document Overview

This document consists of two main chapters: the first one gathers work on triangulations, the second presents work on curved objects. These two chapters are concluded by a set of open directions. The third chapter quickly surveys other results.

Chapter 1

Triangulations Made Practical

A Fundamental Structure

The practical usefulness of triangulations, especially Delaunay triangulations, does not need to be proved, neither does their theoretical interest.
The mathematic foundations of Delaunay triangulations and Voronoi diagrams of point sets are well known, as well as their combinatorial properties and the complexity bounds of their computations in the worst case and for random distributions. The different other links of the chain leading to software were left to study.

1.1 Randomization, the First Link of the Chain

Most of this work was already presented in my PhD thesis [60] and in the monograph that followed [61]. A more complete state of the art is available in the literature [Dev96].

This section summarizes the following papers:

[60] *Vers des algorithmes randomisés dynamiques en géométrie algorithmique.*
Thèse de doctorat en sciences, Université Paris-Sud, Orsay, France, 1991.

[61] *Towards dynamic randomized algorithms in computational geometry*,
volume 758 of *Lecture Notes Comput. Sci.* Springer-Verlag, 1993.

[17] A hierarchical representation of objects: the Delaunay tree.
In *Proc. 2nd Annual ACM Symposium on Computational Geometry*, pages 260–268, 1986.
With Jean-Daniel Boissonnat.

[18] On the randomized construction of the Delaunay tree.
Theoretical Computer Science, 112:339–354, 1993.
With Jean-Daniel Boissonnat.

[15, 14] A semidynamic construction of higher-order Voronoi diagrams and its randomized analysis.
Algorithmica, 9:329–356, 1993.
Preceded by a short version: An on-line construction of higher-order Voronoi diagrams and its randomized analysis.
In *Proc. 2nd Canad. Conf. Comput. Geom.*, pages 278–281, 1990.
With Jean-Daniel Boissonnat and Olivier Devillers.

[34, 33] Fully dynamic Delaunay triangulation in logarithmic expected time per operation.
Computational Geometry: Theory and Applications, 2(2):55–80, 1992.
Preceded by a short version: In *Proc. 2nd Workshop on Algorithms and Data Structures*, volume 519 of *Lecture Notes Comput. Sci.*, pages 42–53. Springer-Verlag, 1991.
With Olivier Devillers and Stefan Meiser.

[13, 12] Applications of random sampling to on-line algorithms in computational geometry.
Discrete and Computational Geometry, 8:51–71, 1992.
Preceded by a short version: On-line geometric algorithms with good expected behaviours.
In *Proc. 13th World Congress on Computation and Applied Math.*, pages 137–139, 1991.
With Jean-Daniel Boissonnat, Olivier Devillers, René Schott and Mariette Yvinec.

as well as to a study performed after the PhD:

[66, 62] Union and split operations on dynamic trapezoidal maps.
Computational Geometry: Theory and Applications, 17:153–163, 2000.
Preceded by a short version: In *Proc. 7th Canad. Conf. Comput. Geom.*, pages 181–186, 1995.

After some studies limited to Delaunay triangulations, we could abstract the essential underlying principle, that contains in maintaining the history of the construction throughout the incremental algorithm. We thus obtained and analyzed a general structure: the *IDAG (Influence Directed Acyclic Graph)*, also known as *History DAG*.
This work became so classic that the *IDAG* is presented in the chapter *Randomization and derandomization* of the famous *Handbook of Discrete and Computational Geometry, Second Edition* [GO04] without any citation!

After the PhD, the structure was enriched with the possibility to perform union and split operations on a trapezoidal map, which was an attempt to adapt the work to parallel machines.

In addition to the exciting chance to participate to research on some "hot" topic, the interest of these advances was in the practical aspect of the randomized algorithms, in

particular the incremental algorithms: they are easy to code, efficient, and their complexity analysis is very realistic, since it does not rely on any hypothesis on the data distribution. Let us remark that this structure later inspired a more efficient one, the Delaunay hierarchy [Dev02], still used in CGAL.

The motivation of the above work has an important practical aspect. This is not true for some other studies. Indeed, the technique also allows to solve other kinds of more theoretical problems:

[20, 19] Splitting a Delaunay triangulation in linear time.
Algorithmica, 34:39–46, 2002.
Preceded by a short version:
In *Proc. 9th. European Symposium on Algorithms*, volume 2161 of *Lecture Notes Comput. Sci.*, pages 312–320. Springer-Verlag, 2001.
With Bernard Chazelle, Olivier Devillers, Ferran Hurtado, Mercè Mora and Vera Sacristán.

Computing the Delaunay triangulation of n points in the plane admits a $\Omega(n \log n)$ lower bound. However, when additional information is available, more efficient algorithms can be obtained. Here, we show that given two sets of points, when the Delaunay triangulation of their union is known, the Delaunay triangulation of each set can be computed by a randomized algorithm of linear size, which is just an apparent contradiction with the lower bound.

1.2 Representation of Triangulations

This section summarizes the following papers:

[63] Three dimensional triangulations in CGAL.
In *Abstracts 15th European Workshop on Computational Geometry*, pages 175–178. INRIA Sophia-Antipolis, 1999.

[6] Programming with CGAL: The example of triangulations.
In *Proc. 15th Annual ACM Symposium on Computational Geometry*, pages 421–423, 1999.
With Jean-Daniel Boissonnat, Frédéric Cazals, Frank Da, Olivier Devillers, Sylvain Pion, François Rebufat and Mariette Yvinec.

[11, 16] Triangulations in CGAL.
Computational Geometry: Theory and Applications, 22:5–19, 2002.
With Jean-Daniel Boissonnat, Olivier Devillers, Sylvain Pion and Mariette Yvinec.

Preceded by a short version: In *Proc. 16th Annual ACM Symposium on Computational Geometry*, pages 11–18, 2000.
With Jean-Daniel Boissonnat, Olivier Devillers and Mariette Yvinec.

I started to work on the CGAL 3D triangulation package in 1997. Global design choices for the packages and classes had already been made by the first CGAL developers. A package of 2D triangulations already existed [Yvi06] and was a solid source of inspiration for the 3D. However, going to 3D raised new problems, and some of the preliminary choices made for the 2D had to be questioned. Very lively discussions happened within the PRISME group, yielding choices that are still valid today.

Let us summarize these decisions:

- Adding an infinite vertex to $\mathbb{R}^2$ and $\mathbb{R}^3$, which solves the problem of handling the unbounded face/cell of the triangulation. The triangulation is thus a combinatorial triangulation of the sphere $\mathbb{S}^2$ in $\mathbb{R}^3$ (resp. $\mathbb{S}^3$ in $\mathbb{R}^4$).
- Reinforcing the separation between the combinatorial triangulation and its geometric embedding.[1]

Also, CGAL handles all degenerate configurations. In particular in 3D, the changes of dimension when dealing when degenerate dimensions (when points are all collinear, or all coplanar) are performed in a mathematically sound way [63].
The first 3D triangulations package was released in 2000 [65, 64].
The later improvements in this package, consisting in optimizations or new functionalities, raised new questions presented in the next sections.

1.3 Location in a Triangulation

This section summarizes the following papers:

[38, 37] Walking in a triangulation.
International Journal on Foundations of Computer Science, 13:181-199, 2002.
Preceded by a short version: In *Proc. 17th Annu. ACM Symposium on Computational Geometry*, pages 106-114, 2001.
With Olivier Devillers and Sylvain Pion.

The point location problem is a fundamental question. Several sophisticated structures can answer the question in optimal time [Pre90, Kir83] but they are often difficult to code. It is

[1] in particular under the impulsion of Lutz Kettner, one of the CGAL "dinosaurs", who developed the 3D Polyhedron package and its underlying combinatorial Half edge data structure.

often judicious to replace them by simpler techniques such as traversals using adjacencies between triangles.
We have studied the performances of different walking techniques:
- walking along a line,
- walking in the directions of coordinate axes,
- walking using visibility relations,
- a variant of the visibility walk, using random decisions in the case of cyclic triangulations for which the visibility relation has cycles,
from both theoretical and practical points of view. We counted not only the number of visited simplices but also the cost of visiting a simplex. Based on this study, we finally decided to implement the last strategy, since it has a little bit better performance than the others (up to 10%), and because it is easy to code and does not suffer from degenerate cases.
Let us remark that, whereas the complexiy of the line walk in a 2D Delaunay triangulation is known for a random distribution of points,[2] the question is still open for visibility walks.

1.4 The Famous Degenerate Cases

---

[39] Perturbations and Vertex Removal in a 3D Delaunay Triangulation.
In *Proc. 14th ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 313-319, 2003.
With Olivier Devillers.

[40] Perturbations and vertex removal in Delaunay and regular 3D triangulations.
Research Report INRIA 5968, 2006.
With Olivier Devillers.

Classical literature of computational geometry assumes that
- numerical computations are exact,
- input data are in general position,
two hypotheses in general not satisfied in practice.
This work lies in the framework of the exact geometric computation paradigm pioneered by C. Yap [YD95]. This model consists in ensuring that geometric predicates (the tests on which algorithms base their decisions) are evaluated exactly, and allows to guarantee the correctness of the results of the algorithms.
Note that exact geometric computation should not be assimilated to exact arithmetic. Filtering techniques allow to use fast approximate algorithms most of the time and to combine exactness and efficiency [BBP01, DP03].

[2] the number of visited triangles to reach q from p is $O(|pq|\sqrt{n})$ [BD98]

In the exact geometric computation framework, degenerate cases can be detected: they correspond to cases where some predicates vanish. CGAL handles degenerate cases explicitely.
Perturbation techniques can be divided into two categories: Symbolic perturbations [Yap90, EM90, Sei98] and controlled perturbations [HS98, FKMS05]. We use symbolic perturbations.

In fact, when some points are cospherical, the Delaunay triangulation is not uniquely defined. The problem consists in giving a unique definition.
Note that using perturbation techniques that perturb the input data [ADS00] may have major drawbacks: tetrahedra can become flat , which is unacceptable both from a practical point of view for users and from a theoretical point of view, since the sphere circumscribing a flat tetrahedron is not even defined.
Edelsbrunner and Mücke write that using their technique for Delaunay triangulations is *"a real pain"* and suggest to use the transformation into a convex hull in higher dimension and to perturb the computation of this convex hull [EM90].
We propose a variant, that perturbs only the fourth coordinate of the 3D points projected on the unit paraboloid in 4D. The 3D points themselves are not perturbed, which avoids creating flat tetrahedra since the orientation predicate is not modified. The perturbed cosphericity predicate amounts to computing the sign of the determinant

$$Det_\varepsilon(s_i,s_j,s_k,s_l,s_m) = \begin{vmatrix} 1 & 1 & 1 & 1 & 1 \\ x_i & x_j & x_k & x_l & x_m \\ y_i & y_j & y_k & y_l & y_m \\ z_i & z_j & z_k & z_l & z_m \\ t_i+\varepsilon^{n-i} & t_j+\varepsilon^{n-j} & t_k+\varepsilon^{n-k} & t_l+\varepsilon^{n-l} & t_m+\varepsilon^{n-m} \end{vmatrix}$$

where

$$t_\star = x_\star^2 + y_\star^2 + z_\star^2 \text{ pour } \star = i,j,k,l,m$$

Det_ε is a polynomial in ε, and it can be shown that it never vanishes as soon as the points are not all coplanar.
The method was recently generalized to the case of the additively weighted Delaunay triangulation (or regular triangulation). The proof is more tricky since new kinds of degeneracies must be considered.

This method, that does not seem to be a special case of any general scheme known up to now, is extremely simple to code, and has the advantage of not introducing any other predicate. Also, it can be observed that the result only depends on an indexing of the points, and that this choice is free. The indexing was initially in CGAL the insertion order of the points, it was later replaced by the lexicographic ordering, which is intrinsecal.
This work allowed to deterministically define and compute a unique Delaunay triangulation in all cases, which is crucial in particular for removing vertices. To our knowledge, CGAL is the only publicly avalaible software offering this functionality for the 3D Delaunay and weighted Delaunay triangulations.

1.5 CGAL Packages

CGAL is the natural output of our research, and also raises new research questions: before trying to code vertex removal in a 3D Delaunay triangulation, I had not thought that there was any new question to solve, and I thought I would only have to achieve a tedious programming work. This continuous exchange between programming and research, one feeding the other and reciprocally, is a remarquable feature of CGAL.

The work described in this chapter led to the following CGAL packages:

---

[65] 3D triangulations.
In CGAL Editorial Board, editor, CGAL *User and Reference Manual*. 2.1 and 2.2 edition, 2000.

[64] 3D triangulation data structure.
In CGAL Editorial Board, editor, CGAL *User and Reference Manual*. 2.1 and 2.2 edition, 2000.

[51] 3D triangulations.
In CGAL Editorial Board, editor, CGAL *User and Reference Manual*. 2.3, 2.4, 3.0, 3.1, 3.2 and 3.3 edition, 2001, 2002, 2003, 2004, 2006 and 2007.
With Sylvain Pion, whose skills in C++ are precious, and who made the code more efficient in particular by his important work on arithmetic filtering, and an improvement of memory management.

[50] 3D triangulation data structure.
In CGAL Editorial Board, editor, CGAL *User and Reference Manual*. 2.3, 2.4, 3.0, 3.1, 3.2 and 3.3 edition, 2001, 2002, 2003, 2004, 2006 and 2007.
With Sylvain Pion.

The impact of this work is difficult to precisely evaluate, since CGAL is distributed under an Open Source license.[3] and can be downloaded on the projet web site (that registers roughly 14000 downloads per release). Still, the volumes of discussions on the public mailing list (about 800 subscribers) and users sending feedback (sometimes to ask for new functionalities) allow us to understand the real importance of the impact in particular in the academic world, in computational geometry (surface reconstruction, meshing,...) and outside (computer graphics, fluid mechanics, structural biology, topology,...).[4] Note that the 3D triangulation packages are the foundation of the CGAL meshing packages [RY06] and are used for instance by Tamal Dey for his reconstruction software[5] [DG01] as well as Deok-Soo Kim[6]. Also, commercial licenses were sold via GeometryFactory to the

[3] QPL for triangulations and more generally almost all combinatorial structures, whereas the basic functionalities (the "kernels") are under LGPL

[4] The page *Projects using* CGAL `http://www.cgal.org/projects.html` tries to list users.

[5] `http://www.cse.ohio-state.edu/~tamaldey/cocone.html`

[6] `http://voronoi.hanyang.ac.kr/`

petroleum companies Midland Valley Exploration (United Kingdom) and Total (France), as well as to BSAP (Switzerlan) for tunnel drilling and France Télécom and British Telecom for antenna placement.

1.6 Towards Other Geometries

1.6.1 Space of Spheres and Hyperbolic Geometry

The general motivation of this work was to generalize to the case of non-parallel planes the algorithm computing the Delaunay triangulation of a set $\mathcal{S}$ of points measured in two parallel sections of an object. This algorithm is optimal and consists in superimposing the two respective Voronoi diagrams of the sets [Boi88].
The algorithm generalizes in fact by replacing the Euclidean Voronoi diagrams by Voronoi diagrams for an hyperbolic metric.

[8, 7] Output-sensitive construction of the Delaunay triangulation of points lying in two planes.
International Journal of Computational Geometry and Applications, 6(1):1–14, 1996.
Preceded by a short version: Output-sensitive construction of the 3-d Delaunay triangulation of constrained sets of points.
In *Proc. 3rd Canad. Conf. Comput. Geom.*, pages 110–113, 1991.
With Jean-Daniel Boissonnat, André Cérézo and Olivier Devillers.

We introduced the space of spheres, that is not a revolutionary notion in mathematics since it is already present in the reference books by Berger [Ber77, Ber87], but allowed a new vision in computational geometry. In the Euclidean plane, a circle of radius r centered at (x, y) is associated in the space of circles to the point $(x, y, x^2 + y^2 - r^2)$. in this space, pencils of circles are lines, and points in the plane (circles of radius 0) are associated with points of the unit paraboloid.

[35, 36] The space of spheres, a geometric tool to unify duality results on Voronoi diagrams.
In *Proc. 4th Canad. Conf. Comput. Geom.*, pages 263–268, 1992.
Long version: Research Report INRIA 1620, 1992.
With Olivier Devillers and Stefan Meiser.

This mathematic vision allowed us to give a unified framework to all known results on the various generalizations of Voronoi diagrams and the projections of levels in arrangements of higher dimensions [ES86].

We also showed that the Voronoi diagram in the hyperbolic plane could be easily deduced from a Voronoi diagram in the Euclidean plane.
In the Poincaré half-plane model, the hyperbolic plane is represented by the half-plane $\{z \in \mathbb{C}, \mathrm{Im}z > 0\}$ ($\mathrm{Im}z$ denotes the imaginary part of the complex z) limited by the infinite line $\mathrm{Im}z = 0$. In this model, the hyperbolic lines are half-circles (in the Euclidean sense) orthogonal to the infinite line.
In the space of spheres, a line parallel to a given direction represents a pencil of circles of given radical axis. This is the essential argument for proving that a Euclidean Voronoi diagram is the vertical projection of the lower envelope of planes that are tangent to the unit paraboloid. The same argument shows that, if this lower envelope is projected onto the paraboloid in the direction corresponding to the infinite line of the Poincaré half-plane, then vertically onto the plane, then the hyperbolic Voronoi diagram is obtained.

This work was the opportunity (this is too rare) to dream on hyperbolic geometry in front of a blackboard. These dreams were written by André Cérézo in a pedagogic publication [Cér91] (section IV-E: "beyond the mirror"), with his thin writing that we don't forget.
The construction of a bisecting line can in fact continue outside the Poincaré half-plane (thus, "beyond the mirror"). The curved is then an arc of an hyperbola, that is the imaginary extension of the bisecting line of the two points.

These works on the space of spheres and hyperbolic geometry ware unfortunately never published, but they would certainly deserve to be exploited in CGAL.

1.6.2 New Questions

Apart from the previous section, all work presented in this chapter takes place in the Euclidean space $\mathbb{R}^3$. This is in fact the case in roughly all the literature in computational geometry. Still, other spaces are necessary for applications. For instance, the case of a periodic parameter space like $[0,1] \times [0,1] \times [0,1]$ is very important for simulations in fluid mechanisms and astrophysics, among other fields.
In these application fields, it is usual to duplicate a certain number of points (it seems that roughly 20% of the points must be duplicated in general) around the boundaries of the reference domain, in order to simulate periodicity. This results in overheads both in running times and memory.
Developing data structures and algorithms able to represent and compute directly triangulations in a space having the topology of the torus $\mathbb{T}^3$ in $\mathbb{R}^4$ is thus a very important issue.
Let us notice here that triangulations in such spaces are not always cellular complexes, since a cell can have several times the same vertex and the same neighbor. Sampling conditions can guarantee that the triangulation is a cellular complex.

This work will lead to a complete reorganization of the architecture of the CGAL packages and to some publications. It was started during the 6 months[7] post-doctoral stay of Nico

[7] Nico was also completing his work on *Skin surfaces*

Kruithof in 2006 and is currently being studied further by Manuel Caroli who is starting his PhD thesis.
Let us also mention that we also started, during Mridul Aanjaneya's internship, to work on computing triangulations in the projective plane [1], while studies in computational geometry were limited to the oriented projective plane [Sto91].
No more detail is given in this document.

Chapter 2

Curved Objects

Geometric algorithms traditionally limit their framework to manipulating linear objects (points, segments, triangles) in affine Euclidean space, curved objects being discretized by linear elements.
Handling directly curved objects, which would allow not to increase tremendoussly the number of manipulated objects, is one of the important challenges in computational geometry.
Its is still rather usual to find in the best literature short sentences quickly claiming that the presented algorithm can easily be adapted to the case of curves, followed by a theoretical complexity analysis. The crucial problem is then ignored: how are the geometric primitives used by the algorithm evaluated? let us recall here that the algorithms we are interested in assume primitives to be evaluated *exactly*.

As will be seen in this chapter, a large part of this work up to now focuses on the study of circles, spheres and circular arcs in dimensions two and three.
These cases did not seem to be a goal at the beginning. However, as the work progressed (see Section 2.3), several applications appeared, justifying to consider them not only as a step to design an test methods before studying more general cases, but as an important topic in itself. Indeed, robust and efficient manipulations of circular arcs in the plane are of capital importance in some industrial domains like manufacturing and printed or integrated circuits design. Manipulations of spheres, and circular arcs in 3D, are central in geometric aspects of structural biology.
The recent increasing interest in the community for approximationss of curves by circular arcs [DRS06, AAH+07] also confirms that this research direction was well chosen, which was not so obvious for everybody at the beginning.
This chapter will also mention more prospective work on more general objects.

The work on arrangements of quadrics, algebraic and geometric primitives, and CGAL kernels, is mentioned in the following book chapter:

[42] Arrangements.
In Jean-Daniel Boissonnat and Monique Teillaud, editors, *Effective Computational*

Geometry for Curves and Surfaces, pages 1–66. Springer-Verlag, Mathematics and Visualization, 2006.
With Efi Fogel, Dan Halperin, Lutz Kettner, Ron Wein and Nicola Wolpert.

2.1 Circular Arcs

Great challenge, but humble start. Already in 1997, in the Prisme project-team at that time, I started a working group gathering a few members of Fiable[1] (INRIA *incentive action*). We worked on the evaluation of primitives on circular arcs, and more precisely, as a start, the comparision of abscissae of two points each defined as the intersection of two circular arcs, which is one of the most interesting predicates for computing an arrangement of circular arcs (the predicates for computing Voronoi diagrams of circles being far from reachable at that time).

This section summarizes the following publications:

[32, 30, 31] Algebraic methods and arithmetic filtering for exact predicates on circle arcs.
Computational Geometry: Theory and Applications, 22:119–142, 2002.
Preceded by short versions:
- In *Abstracts 16th European Workshop on Computational Geometry*, pages 117–120. Ben-Gurion University of the Negev, 2000.
- Exact predicates for circle arcs arrangements. In *Proc. 16th Annual ACM Symposium on Computational Geometry*, pages 139–147, 2000.
With Olivier Devillers, Alexandra Fronville ans Bernard Mourrain.

The restriction of the problem to the case of circular arcs may seem to make it trivial: it reduces to comparisons of chosen roots of two degree two equations. A manual (still, careful) elimination of radicals in the roots expressions, allowing rational computations, seems to solve the question. However, some interesting questions still arise.

The first question is the representation of circles which has an important impact on the size and the degree of algebraic expressions, and on the length of numbers that are manipulated. Also, our goal was to draw methods that could be generalized to higher degree objects in the future. We used several algebraic ingredients [Yap93]: resultants, invariants, Descartes rule, which allowed us to reduce the problem to the evaluation of signs of polynomial expressions of maximal degree 12. Then usual filtering methods can be used, to get an efficient exact evaluation of these signs.

[1] `http://www-sop.inria.fr/prisme/fiable/`

An interesting aspect of this work is also the geometric interpretation of these polynomial expressions, even if it is not concretely used in the computations. Quantities appear, like differences of abscissae between the circles centers, widths of the circular arcs, powers of some points with respect to the circles, a cross-ratio between the projections of intersection points.
This interpretation confirms, if necessary, that algebra and geometry are intimately related, but I also think that going deeper would allow to better understand this kind of questions, and to establish mechanisms for geometric reasoning that could complete algebraic methods. This direction was not explored.

Our goal of drawing general methods that could apply to higher degree curves was mentioned earlier. Unfortunately, we have not been able to do so even for conics, that lead to manipulations of algebraic numbers of degree four. Sturm sequences were shown later to be more adapted to these questions [KE03, EK06, ET04].
Still, in the case of circles, the methods based on Sturm sequences yield to the same polynomial expressions as ours in the case of circles. This is not surprising at all since these expressions do have an intrinsic geometric meaning.
The results of this work are the basis of the CGAL implementation presented in Section 2.3.

2.2 Arrangements of Quadrics in the Space

This section summarizes the following publications:

[47, 46] On the computation of an arrangement of quadrics in 3D.
Computational Geometry: Theory and Applications, 30:145–164, 2005.
Preceded by a short version: Sweeping an Arrangement of Quadrics in 3D. In *Proc. 19th European Workshop on Computational Geometry*, pages 31–34, 2003.
With Bernard Mourrain and Jean-Pierre Técourt.

While implementations of arrangements of 2D curves are now starting to be rather well mastered [BEH+05, WFZH07] (for easy curves, regarding the algebraic questions), it is not the case for surfaces in 3D. The topology of the arrangement is then very complex. Some work define and analyze spatial decompositions, like the cylindrical algebraic decomposition [Col75], a stratification [CEGS91], or the vertical decomposition [SA95], but they do not always come with algorithms for construction. More concrete results were given, even reaching an experimental study for triangles in 3D, with a complexity analysis still valid for general well-behaved algebraic surfaces [SH02].

A complete implementation of arrangements of quadrics can only be considered in the long term. Most work in this domain chose surfacic representation of the arrangements, i.e. a

description by its faces of dimension no more than two [GHS01]. A near-optimal parameterization, in terms of radicals, of the intersections of two quadrics was proposed [DLLP03] and the computation of these intersections was implemented [LPP04, qi]. A complete classification of the intersections of quadrics was recently given [DLLP05a, DLLP05b, DLLP05c]. We proposed a sweeping algorithm that follows a volumic approach instead, since the 3D cells are described. More precisely, like in [SH02], the algorithm computes the vertical decomposition of the arrangement. We took a particular care of the algebraic aspects, which is the deadlock for this construction. The algorithm requires comparisons of algebraic numbers of degree up to 16, so, it requires computations in an extension of degree 256... In the case of spheres, the algebraic numbers are only of degree 4, which allows to be more optimistic about an implementation.

Moreover, in an on-going work with Daniel Russel, we propose a new decomposition of the space, using only algebraic numbers of degree two. This result allows an implementation, which is in progress.
Though particular, the case of spheres is important since it allows to hope future uses in structural biology: knowing the cells of the arrangement, an additional computation of their volume allows to estimate energies in molecules [EM86].

2.3 A Curved Kernel for CGAL

This section summarizes the research on designing a curved kernel for CGAL.

[41, 52, 67] Towards an Open Curved Kernel.
In *Proc. 20th Annual ACM Symposium on Computational Geometry*, pages 438–446, 2004.
With Ioannis Z. Emiris, Athanasios Kakargias, Sylvain Pion and Elias P. Tsigaridas.
Preceded by preliminary studies:
- Towards a CGAL-like kernel for curves.
Research Report ECG-TR-302206-01, 2003.
With Sylvain Pion.
- First prototype of a CGAL geometric kernel with circular arcs.
Research Report ECG-TR-182203-01, 2002.

This research yielded packages:

[53] 2D Circular Kernel.
In CGAL Editorial Board, editor, CGAL *User and Reference Manual*. 3.2 and 3.3 edition, 2006 and 2007.
With Sylvain Pion.

[69] 3D Circular Kernel.
In CGAL Editorial Board, editor, CGAL *User and Reference Manual*. Submitted.

This package was also the topic of a report:
[68] Specifications of the CGAL 3D Circular Kernel.
Research Report ACS-TR-243302-01, 2007.

The following paper contains recent complements:
[25] Design of the CGAL 3D spherical kernel and application to arrangements of circles on a sphere.
Submitted.
With Pedro Machado Manhães de Castro, Frédéric Cazals ans Sébastien Loriot.

The CGAL kernel gathers elementary objects with rational coordinates and that are mostly linear. Circles are defined in this kernel but hardly any functionality is offered. The work mentioned above settled the bases for programming new types of objects in CGAL: circular arcs, together with points having algebraic coordinates (of degree two).
Introducing computations on circular arcs showed the limits of the current design of the CGAL kernel, since for instance it is not allowed to manipulate different point types with different coordinate types. But it offers the possibility of beeing extended [HHK+01], which allowed us to design a kernel offering computations on circular arcs.
The key idea in this design is to clearly separate the geometric and the algebraic aspects. This separation can be seen as a separation between polynomials or algebraic curves (circles, lines) on one hand, and semi-algebraic curves (arcs, segments) on the other hand.
This separation algebra/geometry is made concrete in the code by the introduction of a template parameter:

```
template < class LinearKernel, class AlgebraicKernel >
struct Circular_kernel_2
```

The implementation led to the sumbission of a new package to the CGAL editorial board, and its integration in release 3.2 of the library [53].

As said earlier, the goal was first more general, but interesting applications showed up, and justified a deeper study to improve the implementation. Indeed, a printed circuit company, in contact with the GeometryFactory start-up, was interested by this work, since industrial software fail on some data sets because of robustness problems. We could use some industrial data sets for our experiments. Moreover, Dassault-Systèmes bought a research license of this package.

An interface with the CGAL arrangement package developed in Tel-Aviv [WFZH06] allowed us to experiment our kernel on these industrial data.

---

[45] An Empirical Comparison of Software for Constructing Arrangements of Curved Arcs (preliminary version).
Research Report ECG-TR-361200-01, 2004.
With Efraim Fogel, Dan Halperin, Ron Wein, Sylvain Pion, Ioannis Emiris, Athanasios Kakargias, Elias Tsigaridas, Eric Berberich, Arno Eigenwillig, Michael Hemmer, Lutz Kettner, Kurt Mehlhorn, Elmar Schömer and Nicola Wolpert.

[48, 49] Benchmarking of different arrangement traits. Research Report ACS-TR-123110-01, 2006.
On the evaluation of 2D curved kernels. Research Report ACS-TR-123104-01, 2006.
With Sylvain Pion and Ilya Suslov.

[54] Geometric filtering of primitives on circular arcs.
Research Report ACS-TR-121105-01, 2006.
With Sylvain Pion and Constantinos P. Tsirogiannis.

[27, 26] Exact and efficient computations on circles in CGAL and applications to VLSI design.
In *Abstracts 23rd European Workshop on Computational Geometry*, pages 219–222. Graz University, 2007.
Full version: Research Report INRIA 6091, 2007.
With Pedro Machado Manhães de Castro and Sylvain Pion.

This works also raised two reports:
[28] Benchmarks and evaluation of algebraic kernels for circles. Research Report ACS-TR-243306-01, 2007.
[29] CGAL package for 2d filtered circular kernel. Research Report ACS-TR-243404-02, 2007.
With Pedro Machado Manhães de Castro and Sylvain Pion.

Several methods were explored, in particular different representations of algebraic numbers, and led to important improvements in both running times and memory, which are available in the release 3.3 of CGAL.
The new representation of algebraic numbers also allowed us to code without new difficulties a kernel for manipulating circular arcs in 3D [69], which will naturally be used with the code of arrangements of spheres (Section 2.2).

This work was done with two objectives: handle circular arcs, and draw methods and techniques generalizing to higher degree. In this way, the main algebraic concepts[2] drawn for circular arcs are still central for any algebraic curves: comparisons of roots of systems, evaluation of the sign of a polynomial at the roots of a system.
This basis being given, precise specifications must still be written, which is in progress as a collaboration with our European ACS partners.

---

[3] Interface specification of algebraic kernel.
Research Report ACS-TR-123101-01, 2006.
With Eric Berberich, Michael Hemmer, Menelaos Karavelas, Sylvain Pion and Elias Tsigaridas.

[2] in the computerese *C++ Standard Template Library*: set of properties anfd funtionalities that a type must provide

[4] Prototype implementation of the algebraic kernel.
Research Report ACS-TR-123101-01, 2006.
With Eric Berberich, Michael Hemmer, Menelaos Karavelas, Sylvain Pion and Elias Tsigaridas.

[5] Revision of Interface specification of algebraic kernel.
Research Report ACS-TR-243300-01, 2007.
With Eric Berberich, Michael Hemmer and Menelaos Karavelas.

Let us emphasize that in fact the central concepts are still valid for non algebraic implicit functions, which opens many new directions in the future...

In parallel, I cooperate with Fabrice Rouillier, Sylvain Petitjean and Sylvain Lazard about an implementation of an algebraic kernel based on FGb/RS [fgb], whose main author will be Luis Mariano Peñaranda.

2.4 Open Questions on Predicates

Underlying questions, which seem fundamental to me, are left open. They are quickly raised in Sections 2.4.2 and 2.4.3.

2.4.1 Degree of geometric objects

Before opening these questions, let us quickly mention some work on the degree of offsets of plane algebraic curves. This question is well posed, which is not the case of the questions quickly brushed in the following sections.

---

[2] The offset to an algebraic curve and an application to conics.
In *Proc. International Conference on Computational Science and its Applications*, volume 3480 of *Lecture Notes Comput. Sci.*, pages 683–696. Springer-Verlag, 2005.
With François Anton, Ioannis Emiris and Bernard Mourrain.

We obtained an exact expression of the degree of the generalized offset[3] of an algebraic curve, using the multiplicities of infinite components and singular locii. In the simple case of conics, the offset has degree 8 for ellipses and hyperbolas, 6 for parabolas, and 4 for circles and pairs of lines, and an implicit equation of the offset can be computed.

[3] locus of the centers of the circles of given radius and tangent to the curve

2.4.2 Minimality of the Set of Predicates

While it is well known that predicates *orientation* and *in_sphere* are necessary and sufficient to compute a Delaunay triangulation, the question is not solved for some other geometric structures.
Two questions arise: predicates required to define and compute a structure, and predicates required for a given algorithm.

The answers would have a practical importance, especially for generic implementations like in CGAL [44]. Indeed each generic class computing a structure is templated by a *Traits_class* providing the algorithms with elementary geometric computations. Every user may implement his own model of the traits class concept. It is important not to require useless requirements from users.
Moreover, minimizing the complexity of the predicates is essential for efficiency. That's what I insisted on in a cooperation with Tel-Aviv, in order to remove some expensive predicates from the list of predicates required by CGAL arrangements. The authors write "Reducing the requirements from the geometric traits-class and minimizing the number of invocations of traits-class functions has a dramatic effect on the performance of our arrangement operations" [WFZH07].

---

[44] Generic programming and the CGAL library.
In Jean-Daniel Boissonnat and Monique Teillaud, editors, *Effective Computational Geometry for Curves and Surfaces*. Springer-Verlag, Mathematics and Visualization, 2006.
With Efi Fogel.

[43] Specification of the traits classes for CGAL arrangements of curves.
Research Report ECG-TR-241200-01, 2003.
With Efi Fogel, Dan Halperin, Ron Wein, Eric Berberich, Arno Eigenwillig, Susan Hert and Lutz Kettner.

2.4.3 Algebraic Degree of the Predicates

As seen in section 2.2 or the previous section, the degree of predicates is an important measure of complexity. It is also a measure of the precision of the computations, or of the length of multiprecision numbers used, if the computations are performed exactly on integers. Still, this question is "fluid". It can be declined in several versions:
- Degree of a given predicate. This is the simplest question since the predicate often reduce to the sign of a polynomial, more precisely a resultant. The resultant is the polynomial of minimal degree that expresses a necessary and sufficient condition for a system to have a solution, so, we could imagine that the degree of the resultant is the degree of the predicate. However this would be wrong: first, knowing the degree of the resultant is not trivial, since methods computing it in fact often compute a multiple; then, the resultant can sometimes

be factorized, and the degree of the predicates would more likely be the maximum degree of its factors; finally, since a factor can have the shape of $P^2 + Q^2$, which has constant sign, the degree of polynomials does not have a clear meaning.
- Degree of an algorithm. Thos notion is extremely badly defined, because it depends on the way geometric predicates required by the algorithms are translated into algebraic terms. A recent example shows that these expressions are far from being canonical: for the Voronoi diagram of circles, predicates supposed to have degree 16 [KE03, EK06] were translated into rational expressions [DM].
- Degree of a problem. This sounds even more complicated. However, the degree necessary to describe a structure is a lower bound for its computation. Still, this bounds remains difficult to obtain.
Furthermore, let us remark that this measure is not necessarily related to the running time that can be expected in practice: indeed, the running time depends more on the number of arithmetic expressions to evaluate the predicate. The CGAL implementation of Voronoi diagram of circles [KY06] uses predicates of degree 20, because predicates of degree 16 (first considered to be optimal, see above) require 100 times as many arithmetic operations.
To summarize, everything is still to be done regarding a formalization of this question and a result of an definition intrinsic to the geometric nature of the computed objects.[4].

[4]I tried to vehicle this idea at the *European Workshop on Computational Geometry* in march 2006 `http://cgi.di.uoa.gr/~ewcg06/` in my invited talk `http://www-sop.inria.fr/geometrica/team/Monique.Teillaud/talks/EWCG.pdf`

Chapter 3

Other Geometric Works and Applications

This chapter just presents shorts abstracts of the cited papers.

3.1 Minkowski Sums and Satellite Layout

[10, 9] Slicing Minkowski sums for satellite antenna layout.
Computer-Aided Design, 30(4):255–265, 1998. Special Issue on Computational Geometry and Computer-Aided Design and Manufacturing.
Preceded by a short version: Minkowski operations for satellite antenna layout.
In *Proc. 13th Annual ACM Symposium on Computational Geometry*, pages 67–76, 1997.
With Jean-Daniel Boissonnat and Eelco de Lange.

This study was performed in the framework of an industrial contract with Matra Marconi Space. Eelco de Lange worked during his PhD 3/4th of his time at Matra Marconi Space in Toulouse, communicating by e-mail and visiting us regularly [dL98].
Satellite layout is a very hard task because available space for the instruments is small and the physical constraints on the layout are strong. In this article, we show how some physical layout constraints can be modeled geometrically and how Minkowski operations can be used to place instruments, and in particular antennas.
The crucial step consists in placing an antenna by translation onto a satellite wall. The admissible space is therefore two-dimensional, although the instrument is three-dimensional. We propose an algorithm to efficiently compute the admissible space and describe its implementation. Experimental results on realistic examples are given.

3.2 Matching Polygonal Objects

[21, 22] Computing the maximum overlap of two convex polygons under translations.
Theory of Computing Systems, 31:613–628, 1998.
Preceded by a short version: In *Proc. 7th Annu. Internat. Sympos. Algorithms Comput.*, volume 1178 of *Lecture Notes Comput. Sci.*, pages 126–135. Springer-Verlag, 1996.
With Mark de Berg, Otfried Cheong, Olivier Devillers and Marc van Kreveld.

Let P be a convex polygon in the plane with n vertices and let Q be a convex polygon with m vertices. We prove that the maximum number of combinatorially distinct placements of Q with respect to P under translations is $O(n^2 + m^2 + \min(nm^2 + n^2m))$, and we give an example showing that this bound is tight in the worst case. Second, we present an $O((n + m)\log(n + m))$ algorithm for determining a translation of Q that maximizes the area of overlap of P and Q.
We also prove that the placement of Q that makes the centroids of Q and P coincide realizes an overlap of at least $9/25$ of the maximum possible overlap. As an upper bound, we show an example where the overlap in this placement is $4/9$ of the maximum possible overlap.

3.3 Cones and Motion Planning

[24, 23] Reaching a goal with directional uncertainty.
Theoretical Computer Science, 140:301–317, 1995.
Preceded by a short version: In *Proc. 4th Annu. Internat. Sympos. Algorithms Comput.*, volume 762 of *Lecture Notes Comput. Sci.*, pages 1–10. Springer-Verlag, 1993.
With Mark de Berg, Leonidas Guibas, Dan Halperin, Mark Overmars, Otfried Schwarzkopf (/Cheong) and Micha Sharir.

We study two problems related to planar motion planning for robots with imperfect control, where, if the robot starts a linear movement in a certain commanded direction, we only know that its actual movement will be confined in a cone of angle α centered around the specified direction.
First, we consider a single goal region, namely the "region at infinity", and a set of polygonal obstacles, modeled as a set S of n line segments. We are interested in the region $\mathcal{R}_\alpha(S)$ from where we can reach infinity with a directional uncertainty of α. We prove

that the maximum complexity of $\mathcal{R}_\alpha(S)$ is $O(n/\alpha^5)$. Second, we consider a collection of k polygonal goal regions of total complexity m, but without any obstacles. Here we prove an $O(k^3m)$ bound on the complexity of the region from where we can reach a goal region with a directional uncertainty of α. For both situations we also prove lower bounds on the maximum complexity, and we give efficient algorithms for computing the regions.

3.4 More Spheres, and Robots

---

[59] Application de la géométrie synthétique au problème de modélisation géométrique directe des robots parallèles.
Mechanism and Machine Theory, 34:255–269, 1999.
With Luc Tancredi.

[58] Forward kinematics of a parallel manipulator with additional rotary sensors measuring the position of platform joints.
In J-P. Merlet and B. Ravani, editors, *Computational Kinematics*, pages 261–270. Kluwer Academic Publishers, 1995.
With Luc Tancredi and Jean-Pierre Merlet.

[57] Extra sensors for solving the forward kinematics problem of parallel manipulators.
In *9th World Congress on the Theory of Machines and Mechanisms*, volume 3, pages 2122–2126, Milan, 1995. IFToMM.
With Luc Tancredi and Jean-Pierre Merlet.

[56] Symbolic elimination for parallel manipulators.
Communication at 4th Internat. Sympos. on Effective Methods in Algebraic Geometry (MEGA), 1996.
With Luc Tancredi and Olivier Devillers.

The forward kinematics problem of parallel manipulators consists in determining the position and the orientation of the platform when the values of the articular variables, which are used to command the robot, are known. When lengths of legs are known, the problem relates to intersections of spheres.
We considered several methods: the so-called *synthetic* geometry (which consists in a few words in computing the algebraic degrees of the geometric locii of points), and symbolic elimination. Measures of additional sensors are also used. This work allowed us to obtain bounds on the numbers of solutions of several variants of the problem.

3.5 Projective Geometry and Camera Calibration

---

[55] On the absolute quadratic complex and its application to autocalibration.
In *Proc. IEEE International Conference on Computer Vision and Pattern Recognition*, volume 1, pages 780 – 787, 2005.
With Jean Ponce, Kenton McHenry, Théo Papadopoulo and Bill Triggs.

I worked on this paper during a visit at Urbana-Champaign. It introduces the absolute quadratic complex formed by all lines that intersect the absolute conic. If ω denotes the 3×3 symmetric matrix representing the image of that conic under the action of a camera with projection matrix P, it is shown that $\omega \approx \overline{\mathcal{P}}\underline{\Omega}\overline{\mathcal{P}}^T$, where $\overline{\mathcal{P}}$ is the 3×6 line projection matrix associated with $\mathcal{P}$, and $\underline{\Omega}$ is a 6×6 symmetric matrix of rank 3 representing the absolute quadratic complex. This simple relation between a camera's intrinsic parameters, its projection matrix expressed in a projective coordinate frame, and the metric upgrade separating this frame from a metric one —as respectively captured by the matrices ω, $\mathcal{P}$, and $\underline{\Omega}$— provides a new framework for autocalibration, particularly well suited to typical digital cameras with rectangular or square pixels since the skew and aspect ratio are decoupled from the other intrinsic parameters in ω.

Perspectives

Par ce texte élémentaire, l'auteur espère faire partager sa joie de la découverte d'une théorie mathématique dont la beauté l'émeut, la richesse le ravit, la profondeur l'impressionne, et le pouvoir de description de l'univers le déconcerte.

André Cérézo [Cér91]

Of course, far be it from me the idea of comparing this work to a beautiful mathematical theory... Still, in due proportion, I can see some "beauty" in some axes of this research.
Working for the CGAL library, or in the CGAL project, is interesting by several ways. While for me, programming is not a pleasure in itself, CGAL still provides some satisfactions: the demand for quality that can look like a constraint is in fact a profit, the construction of something that actually works is gratifying, as well as the practical usefulness for the numerous users (writing a non-used code would look like a pure waste of time to me). And of course, the visibility and the impact that the CGAL project gives to its participants must not be neglected.
Even more, the interest of a certain form of abstraction appears, lying every day in the definition of concepts (in the sense of the STL *(standard template library)*), but that goes closer ro revealing underlying mathematical concepts when studied deeper. I am far from opposing them, like some communities of programmers unfortunately use to, on the contrary I insist to show their convergences in the CGAL community.
As a matter of interest, I will mention a conversation with Daniel Lazard in may 2006. A caricaturist would show us, me talking about programs (programming elementary functionalities on circular arcs), and him of mathematics; but in a short time, I saw with pleasure that, even though our starting points were different, our conversations were perfectly converging: *solve* and *sign_at* are the heart of the solution![1] Such dialogues are not only reassuring but also very rewarding and constructive, much more than claims sometimes heard ("The requirements for a type to be model of Kernel::Point_2 is basically to be CopyConstructible. Nothing more."), that seem to me to be on the wrong track in occulting the mathematical semantics of manipulated objects.

[1] See Section 2.3

The more I code, and the more I am convinced that choosing the right mathematical concepts is crucial to obtain good code, in particular code aiming at some perennity. Programming mathematical concepts sounds to me like the source of a lot of good research topics.
The interest of this topic is far from being only related with the taste of a researcher for a nice formalism. The needs are currently important in many fields: the computation of triangulations in periodic domains (3D torii or cylinders in $\mathbb{R}^4$) are necessary for simulations[2] (Section 1.6.2), computations in hyperbolic spaces are used for studying cristalline structures[3], projective geometry gives a natural framework to many computer vision problems (multi-camera geometry and calibration for instance), etc
Various mathematical and computer science fields must intervene in this wide topic. We talked in this document of algebraic geometry, combinatorics, projective geometry, programming, computer arithmetic, hyperbolic geometry, algorithmic, algebraic topology... This multi-disciplinarity makes the topic particularly attractive, both by its scientific interest and the multiple collaborations it implies.
The algorithmic and programming techniques are now ready to allow important advances in providing useful and efficient mathematical tools.

[2]see for instance `http://xxx.lanl.gov/abs/cond-mat/0301378`
[3]see for instance `http://wwwrsphysse.anu.edu.au/~vbr110/index.php`

Publications

[1] Mridul Aanjaneya and Monique Teillaud. Triangulating the real projective plane. Research Report 6296, INRIA, 09 2007.

[2] François Anton, Ioannis Emiris, Bernard Mourrain, and Monique Teillaud. The offset to an algebraic curve and an application to conics. In *Proc. International Conference on Computational Science and its Applications*, volume 3480 of *Lecture Notes Comput. Sci.*, pages 683–696. Springer-Verlag, 2005.

[3] Eric Berberich, Michael Hemmer, Menelaos Karavelas, Sylvain Pion, Monique Teillaud, and Elias Tsigaridas. Interface specification of algebraic kernel. Research Report ACS-TR-123101-01, INRIA, NUA, MPI, 2006.

[4] Eric Berberich, Michael Hemmer, Menelaos Karavelas, Sylvain Pion, Monique Teillaud, and Elias Tsigaridas. Prototype implementation of the algebraic kernel. Research Report ACS-TR-121202-01, INRIA, NUA, MPI, 2006.

[5] Eric Berberich, Michael Hemmer, Menelaos Karavelas, and Monique Teillaud. Revision of interface specification of algebraic kernel. Research Report ACS-TR-243300-01, INRIA, NUA, MPI, 2007.

[6] Jean-Daniel Boissonnat, Frédéric Cazals, Frank Da, Olivier Devillers, Sylvain Pion, François Rebufat, Monique Teillaud, and Mariette Yvinec. Programming with CGAL: The example of triangulations. In *Proc. 15th Annual ACM Symposium on Computational Geometry (Short communication)*, pages 421–423, 1999.

[7] Jean-Daniel Boissonnat, André Cérézo, Olivier Devillers, and Monique Teillaud. Output-sensitive construction of the 3-d Delaunay triangulation of constrained sets of points. In *Proc. 3rd Canad. Conf. Comput. Geom.*, pages 110–113, 1991.

[8] Jean-Daniel Boissonnat, André Cérézo, Olivier Devillers, and Monique Teillaud. Output-sensitive construction of the Delaunay triangulation of points lying in two planes. *International Journal of Computational Geometry and Applications*, 6(1):1–14, 1996.

[9] Jean-Daniel Boissonnat, Eelco de Lange, and Monique Teillaud. Minkowski operations for satellite antenna layout. In *Proc. 13th Annual ACM Symposium on Computational Geometry*, pages 67–76, 1997.

[10] Jean-Daniel Boissonnat, Eelco de Lange, and Monique Teillaud. Slicing Minkowski sums for satellite antenna layout. *Computer-Aided Design*, 30(4):255–265, 1998. Special Issue on Computational Geometry and Computer-Aided Design and Manufacturing.

[11] Jean-Daniel Boissonnat, Olivier Devillers, Sylvain Pion, Monique Teillaud, and Mariette Yvinec. Triangulations in CGAL. *Computational Geometry: Theory and Applications*, 22:5–19, 2002.

[12] Jean-Daniel Boissonnat, Olivier Devillers, René Schott, Monique Teillaud, and Mariette Yvinec. On-line geometric algorithms with good expected behaviours. In *Proc. 13th World Congress on Computation and Applied Math.*, pages 137–139, 1991.

[13] Jean-Daniel Boissonnat, Olivier Devillers, René Schott, Monique Teillaud, and Mariette Yvinec. Applications of random sampling to on-line algorithms in computational geometry. *Discrete and Computational Geometry*, 8:51–71, 1992.

[14] Jean-Daniel Boissonnat, Olivier Devillers, and Monique Teillaud. An on-line construction of higher-order Voronoi diagrams and its randomized analysis. In *Proc. 2nd Canad. Conf. Comput. Geom.*, pages 278–281, 1990.

[15] Jean-Daniel Boissonnat, Olivier Devillers, and Monique Teillaud. A semidynamic construction of higher-order Voronoi diagrams and its randomized analysis. *Algorithmica*, 9:329–356, 1993.

[16] Jean-Daniel Boissonnat, Olivier Devillers, Monique Teillaud, and Mariette Yvinec. Triangulations in CGAL. In *Proc. 16th Annual ACM Symposium on Computational Geometry*, pages 11–18, 2000.

[17] Jean-Daniel Boissonnat and Monique Teillaud. A hierarchical representation of objects: the Delaunay tree. In *Proc. 2nd Annual ACM Symposium on Computational Geometry*, pages 260–268, 1986.

[18] Jean-Daniel Boissonnat and Monique Teillaud. On the randomized construction of the Delaunay tree. *Theoretical Computer Science*, 112:339–354, 1993.

[19] Bernard Chazelle, Olivier Devillers, Ferran Hurtado, Mercè Mora, Vera Sacristán, and Monique Teillaud. Splitting a Delaunay triangulation in linear time. In *Proc. 9th. European Symposium on Algorithms*, volume 2161 of *Lecture Notes Comput. Sci.*, pages 312–320. Springer-Verlag, 2001.

[20] Bernard Chazelle, Olivier Devillers, Ferran Hurtado, Mercè Mora, Vera Sacristán, and Monique Teillaud. Splitting a Delaunay triangulation in linear time. *Algorithmica*, 34:39–46, 2002.

[21] Mark de Berg, Otfried Cheong, Olivier Devillers, Mark van Kreveld, and Monique Teillaud. Computing the maximum overlap of two convex polygons under translations. *Theory of Computing Systems*, 31:613–628, 1998.

[22] Mark de Berg, Olivier Devillers, Marc van Kreveld, Otfried Schwarzkopf, and Monique Teillaud. Computing the maximum overlap of two convex polygons under translations. In *Proc. 7th Annu. Internat. Sympos. Algorithms Comput.*, volume 1178 of *Lecture Notes Comput. Sci.*, pages 126–135. Springer-Verlag, 1996.

[23] Mark de Berg, Leonidas Guibas, Dan Halperin, Mark Overmars, Otfried Schwarzkopf, Micha Sharir, and Monique Teillaud. Reaching a goal with directional uncertainty. In *Proc. 4th Annu. Internat. Sympos. Algorithms Comput.*, volume 762 of *Lecture Notes Comput. Sci.*, pages 1–10. Springer-Verlag, 1993.

[24] Mark de Berg, Leonidas Guibas, Dan Halperin, Mark Overmars, Otfried Schwarzkopf, Micha Sharir, and Monique Teillaud. Reaching a goal with directional uncertainty. *Theoretical Computer Science*, 140:301–317, 1995.

[25] Pedro M. M. de Castro, Frédéric Cazals, Sébastien Loriot, and Monique Teillaud. Design of the cgal 3D spherical kernel and application to arrangements of circles on a sphere, 2007. Submitted.

[26] Pedro M. M. de Castro, Sylvain Pion, and Monique Teillaud. Exact and efficient computations on circles in CGAL. In *Abstracts 23rd. European Workshop on Computational Geometry*, pages 219–222. Technische Universität Graz, Austria, 2007.

[27] Pedro M. M. de Castro, Sylvain Pion, and Monique Teillaud. Exact and efficient computations on circles in CGAL and applications to VLSI design. Research Report 6091, INRIA, 01 2007.

[28] Pedro Machado Manhães de Castro, Sylvain Pion, and Monique Teillaud. Benchmarks and evaluation of algebraic kernels for circles. Technical Report ACS-TR-243306-01, INRIA, 2007.

[29] Pedro Machado Manhães de Castro, Sylvain Pion, and Monique Teillaud. CGAL package for 2d filtered circular kernel. Technical Report ACS-TR-243404-02, INRIA, 2007.

[30] Olivier Devillers, Alexandra Fronville, Bernard Mourrain, and Monique Teillaud. Algebraic methods and arithmetic filtering for exact predicates on circle arcs. In *Abstracts 16th European Workshop on Computational Geometry*, pages 117–120. Ben-Gurion University of the Negev, 2000.

[31] Olivier Devillers, Alexandra Fronville, Bernard Mourrain, and Monique Teillaud. Exact predicates for circle arcs arrangements. In *Proc. 16th Annual ACM Symposium on Computational Geometry*, pages 139–147, 2000.

[32] Olivier Devillers, Alexandra Fronville, Bernard Mourrain, and Monique Teillaud. Algebraic methods and arithmetic filtering for exact predicates on circle arcs. *Computational Geometry: Theory and Applications*, 22:119–142, 2002.

[33] Olivier Devillers, Stefan Meiser, and Monique Teillaud. Fully dynamic Delaunay triangulation in logarithmic expected time per operation. In *Proc. 2nd Workshop on Algorithms and Data Structures*, volume 519 of *Lecture Notes Comput. Sci.*, pages 42–53. Springer-Verlag, 1991.

[34] Olivier Devillers, Stefan Meiser, and Monique Teillaud. Fully dynamic Delaunay triangulation in logarithmic expected time per operation. *Computational Geometry: Theory and Applications*, 2(2):55–80, 1992.

[35] Olivier Devillers, Stefan Meiser, and Monique Teillaud. The space of spheres, a geometric tool to unify duality results on Voronoi diagrams. In *Abstracts 8th European Workshop on Computational Geometry*, pages 45–49. Utrecht University, 1992.

[36] Olivier Devillers, Stefan Meiser, and Monique Teillaud. The space of spheres, a geometric tool to unify duality results on Voronoi diagrams. Research Report 1620, INRIA, Valbonne, France, 1992.

[37] Olivier Devillers, Sylvain Pion, and Monique Teillaud. Walking in a triangulation. In *Proc. 17th Annu. ACM Symposium on Computational Geometry*, pages 106–114, 2001.

[38] Olivier Devillers, Sylvain Pion, and Monique Teillaud. Walking in a triangulation. *International Journal on Foundations of Computer Science*, 13:181–199, 2002. Special issue on triangulations.

[39] Olivier Devillers and Monique Teillaud. Perturbations and vertex removal in a 3D Delaunay triangulation. In *Proc. 14th ACM-SIAM Symposium on Discrete Algorithms*, pages 313–319, 2003.

[40] Olivier Devillers and Monique Teillaud. Perturbations and vertex removal in Delaunay and regular 3D triangulations. Research Report 5968, INRIA, 2006.

[41] Ioannis Z. Emiris, Athanasios Kakargias, Sylvain Pion, Monique Teillaud, and Elias P. Tsigaridas. Towards an open curved kernel. In *Proc. 20th Annual ACM Symposium on Computational Geometry*, pages 438–446, 2004.

[42] Efi Fogel, Dan Halperin, Lutz Kettner, Monique Teillaud, Ron Wein, and Nicola Wolpert. Arrangements. In Jean-Daniel Boissonnat and Monique Teillaud, editors, *Effective Computational Geometry for Curves and Surfaces*, pages 1–66. Springer-Verlag, Mathematics and Visualization, 2006.

[43] Efi Fogel, Dan Halperin, Ron Wein, Monique Teillaud, Eric Berberich, Arno Eigenwillig, Susan Hert, and Lutz Kettner. Specification of the traits classes for CGAL arrangements of curves. Research Report ECG-TR-241200-01, MPI Saarbrücken, INRIA Sophia-Antipolis, Tel-Aviv University, 2003.

[44] Efi Fogel and Monique Teillaud. Generic programming and the CGAL library. In Jean-Daniel Boissonnat and Monique Teillaud, editors, *Effective Computational Geometry for Curves and Surfaces*, pages 313–320. Springer-Verlag, Mathematics and Visualization, 2006.

[45] Efraim Fogel, Dan Halperin, Ron Wein, Sylvain Pion, Monique Teillaud, Ioannis Z. Emiris, Athanasios Kakargias, Elias P. Tsigaridas, Eric Berberich, Arno Eigenwillig, Michael Hemmer, Lutz Kettner, Kurt Mehlhorn, Elmar Schömer, and Nicola Wolpert. An empirical comparison of software for constructing arrangements of curved arcs (preliminary version). Research Report ECG-TR-361200-01, Tel-Aviv University, INRIA Sophia-Antipolis, MPI Saarbrücken, 2004.

[46] Bernard Mourrain, Jean-Pierre Técourt, and Monique Teillaud. Sweeping an arrangement of quadrics in 3d. In *Proc. 19th European Workshop on Computational Geometry*, pages 31–34, 2003.

[47] Bernard Mourrain, Jean-Pierre Técourt, and Monique Teillaud. On the computation of an arrangement of quadrics in 3d. *Computational Geometry: Theory and Applications*, 30:145–164, 2005. Special issue, 19th European Workshop on Computational Geometry.

[48] Sylvain Pion, Ilya Suslov, and Monique Teillaud. Benchmarking of different arrangement traits. Research Report ACS-TR-123110-01, INRIA, 2006.

[49] Sylvain Pion, Ilya Suslov, and Monique Teillaud. On the evaluation of 2D curved kernels. Research Report ACS-TR-123104-01, INRIA, 2006.

[50] Sylvain Pion and Monique Teillaud. 3D triangulation data structure. In CGAL Editorial Board, editor, *CGAL User and Reference Manual*. 2.3, 2.4, 3.0, 3.1, 3.2 and 3.3 edition, 2001, 2002, 2003, 2004, 2006 and 2007.

[51] Sylvain Pion and Monique Teillaud. 3D triangulations. In CGAL Editorial Board, editor, *CGAL User and Reference Manual*. 2.3, 2.4, 3.0, 3.1, 3.2 and 3.3 edition, 2001, 2002, 2003, 2004, 2006 and 2007.

[52] Sylvain Pion and Monique Teillaud. Towards a CGAL-like kernel for curves. Research Report ECG-TR-302206-01, MPI Saarbrücken, INRIA Sophia-Antipolis, 2003.

[53] Sylvain Pion and Monique Teillaud. 2D circular kernel. In CGAL Editorial Board, editor, *CGAL User and Reference Manual*. 3.2 and 3.3 edition, 2006 and 2007.

[54] Sylvain Pion, Monique Teillaud, and Constantinos P. Tsirogiannis. Geometric filtering of primitives on circular arcs. Research Report ACS-TR-121105-01, INRIA, 2006.

[55] Jean Ponce, Kenton McHenry, Théo Papadopoulo, Monique Teillaud, and Bill Triggs. On the absolute quadratic complex and its application to autocalibration. In *Proc. IEEE International Conference on Computer Vision and Pattern Recognition*, volume 1, pages 780 – 787, 2005.

[56] L. Tancredi, M. Teillaud, and O. Devillers. Symbolic elimination for parallel manipulators. Communication at 4th Internat. Sympos. on Effective Methods in Algebraic Geometry, 1996.

[57] L. Tancredi, M. Teillaud, and J-P. Merlet. Extra sensors for solving the forward kinematics problem of parallel manipulators. In *9th World Congress on the Theory of Machines and Mechanisms*, volume 3, pages 2122–2126, Milan, 1995. IFToMM.

[58] L. Tancredi, M. Teillaud, and J-P. Merlet. Forward kinematics of a parallel manipulator with additional rotary sensors measuring the position of platform joints. In J-P. Merlet and B. Ravani, editors, *Computational Kinematics*, pages 261–270. Kluwer Academic Publishers, 1995.

[59] Luc Tancredi and Monique Teillaud. Application de la géométrie synthétique au problème de modélisation géométrique directe des robots parallèles. *Mechanism and Machine Theory*, 34:255–269, 1999.

[60] Monique Teillaud. *Vers des algorithmes randomisés dynamiques en géométrie algorithmique*. Thèse de doctorat en sciences, Université Paris-Sud, Orsay, France, 1991.

[61] Monique Teillaud. *Towards dynamic randomized algorithms in computational geometry*, volume 758 of *Lecture Notes Comput. Sci.* Springer-Verlag, 1993.

[62] Monique Teillaud. Union and split operations on dynamic trapezoidal maps. In *Proc. 7th Canad. Conf. Comput. Geom.*, pages 181–186, 1995.

[63] Monique Teillaud. Three dimensional triangulations in CGAL. In *Abstracts 15th European Workshop on Computational Geometry*, pages 175–178. INRIA Sophia-Antipolis, 1999.

[64] Monique Teillaud. 3D triangulation data structure. In CGAL Editorial Board, editor, *CGAL User and Reference Manual*. 2.1 and 2.2 edition, 2000.

[65] Monique Teillaud. 3D triangulations. In CGAL Editorial Board, editor, *CGAL User and Reference Manual*. 2.1 and 2.2 edition, 2000.

[66] Monique Teillaud. Union and split operations on dynamic trapezoidal maps. *Computational Geometry: Theory and Applications*, 17:153–163, 2000.

[67] Monique Teillaud. First prototype of a CGAL geometric kernel with circular arcs. Research Report ECG-TR-182203-01, INRIA, 2002.

[68] Monique Teillaud. Specifications of the CGAL 3D circular kernel. Research Report ACS-TR-243302-01, INRIA, 2007.

[69] Monique Teillaud. 3D circular kernel. In CGAL Editorial Board, editor, *CGAL User and Reference Manual*. Submitted.

Bibliographie

[AAH+07] Oswin Aichholzer, Franz Aurenhammer, Thomas Hackl, Bert Jüttler, Margot Oberneder, and Zbynek Sir. Computational and structural advantages of circular boundary representation. In *Proc. 10th International Workshop on Algorithms and Data Structures (WADS)*, 2007. To appear.

[ADS00] Pierre Alliez, Olivier Devillers, and Jack Snoeyink. Removing degeneracies by perturbing the problem or the world. *Reliable Computing*, 6:61–79, 2000.

[BBP01] H. Brönnimann, C. Burnikel, and S. Pion. Interval arithmetic yields efficient dynamic filters for computational geometry. *Discrete Applied Mathematics*, 109:25–47, 2001.

[BD98] Prosenjit Bose and Luc Devroye. Intersections with random geometric objects. *Comput. Geom. Theory Appl.*, 10:139–154, 1998.

[BEH+05] E. Berberich, A. Eigenwillig, M. Hemmer, S. Hert, L. Kettner, K. Mehlhorn, J. Reichel, S. Schmitt, E. Schömer, and N. Wolpert. Exacus–efficient and exact algorithms for curves and surfaces. In *Proc. 13th European Symposium on Algorithms*, volume 3669 of *Lecture Notes Comput. Sci.*, pages 155–166, 2005.

[BEPP97] H. Brönnimann, I. Emiris, V. Pan, and S. Pion. Computing exact geometric predicates using modular arithmetic with single precision. In *Proc. 13th Annual ACM Symposium on Computational Geometry*, pages 174–182, 1997.

[Ber77] M. Berger. *Géométrie (vols. 1-5)*. Fernand Nathan, Paris, 1977.

[Ber87] M. Berger. *Geometry (vols. 1-2)*. Springer-Verlag, 1987.

[Boi88] Jean-Daniel Boissonnat. Shape reconstruction from planar cross-sections. *Comput. Vision Graph. Image Process.*, 44(1):1–29, October 1988.

[C+96] Bernard Chazelle et al. Application challenges to computational geometry: CG impact task force report. Technical Report TR-521-96, Princeton Univ., April 1996.

[C+99] Bernard Chazelle et al. Application challenges to computational geometry: CG impact task force report. In B. Chazelle, J. E. Goodman, and R. Pollack, editors, *Advances in Discrete and Computational Geometry*, volume 223 of *Contemporary Mathematics*, pages 407–463. American Mathematical Society, Providence, 1999.

[CEGS91] Bernard Chazelle, H. Edelsbrunner, Leonidas J. Guibas, and Micha Sharir. A singly-exponential stratification scheme for real semi-algebraic varieties and its applications. *Theoret. Comput. Sci.*, 84:77–105, 1991.

[Cér91] André Cérézo. Le plan hyperbolique à pied, puis un bond dans l'espace. Publications pédagogiques, Prépublication No 10, Département de Mathématiques, Université de Nice, France, 1991.

[CGA] CGAL, Computational Geometry Algorithms Library. `http://www.cgal.org`.

[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.

[Dev96] Olivier Devillers. An introduction to randomization in computational geometry. *Theoret. Comput. Sci.*, 157:35–52, 1996.

[Dev02] Olivier Devillers. The Delaunay hierarchy. *Internat. J. Found. Comput. Sci.*, 13:163–180, 2002.

[DG01] Tamal K. Dey and Joachim Giesen. Detecting undersampling in surface reconstruction. In *Proc. 17th Annu. ACM Sympos. Comput. Geom.*, pages 257–263, 2001.

[dL98] Eelco de Lange. *Aide géométrique à l'aménagement de satellites*. Thèse de doctorat en sciences, École Nationale Supérieure des Mines de Paris, France, 1998.

[DLLP03] Laurent Dupont, Daniel Lazard, Sylvain Lazard, and Sylvain Petitjean. Near-optimal parameterization of the intersection of quadrics. In *Proc. 19th Annu. ACM Sympos. Comput. Geom.*, pages 246–255, 2003.

[DLLP05a] L. Dupont, D. Lazard, S. Lazard, and S. Petitjean. Near-optimal parameterization of the intersection of quadrics: I. The generic algorithm. Research Report n° 5667, INRIA, Sept. 2005. 36 pages.

[DLLP05b] L. Dupont, D. Lazard, S. Lazard, and S. Petitjean. Near-optimal parameterization of the intersection of quadrics: II. A classification of pencils. Research Report n° 5668, INRIA, Sept. 2005. 37 pages.

[DLLP05c] L. Dupont, D. Lazard, S. Lazard, and S. Petitjean. Near-optimal parameterization of the intersection of quadrics: III. Parameterizing singular intersections. Research Report n° 5669, INRIA, Sept. 2005. 34 pages.

[DM] Christophe Delage and Dave Millman. Improved predicates for the Apollonius diagram. Personal communication.

[DP03] Olivier Devillers and Sylvain Pion. Efficient exact geometric predicates for Delaunay triangulations. In *Proc. 5th Workshop Algorithm Eng. Exper.*, pages 37–44, 2003.

[DRS06] Scot Drysdale, Günter Rote, and Astrid Sturm. Approximation of an open polygonal curve with a minimum number of circular arcs. In *Proceedings of the 22nd European Workshop on Computational Geometry (EWCG)*, pages 25–28, 2006.

[EK06] I. Z. Emiris and M. I. Karavelas. The predicates of the Apollonius diagram: algorithmic analysis and implementation. *Computational Geometry: Theory and Applications, Special Issue on Robust Geometric Algorithms and their Implementations*, 33(1-2):18–57, 2006.

[EM86] D. Eisenberg and A.D. McLachlan. Solvation energy in protein folding and binding. *Nature*, 319:199–203, 1986.

[EM90] H. Edelsbrunner and E. P. Mücke. Simulation of simplicity: A technique to cope with degenerate cases in geometric algorithms. *ACM Trans. Graph.*, 9(1):66–104, 1990.

[ES86] H. Edelsbrunner and R. Seidel. Voronoi diagrams and arrangements. *Discrete Comput. Geom.*, 1:25–44, 1986.

[ET04] I. Emiris and E. P. Tsigaridas. Computing with real algebraic numbers of small degree. In *Proc. 12th European Symposium on Algorithms*, volume 3221 of *Lecture Notes Comput. Sci.*, pages 652–663. Springer-Verlag, 2004.

[fgb] FGB/RS.
`http://fgbrs.lip6.fr/salsa/Software/`.

[FKMS05] S. Funke, C. Klein, K. Mehlhorn, and S. Schmitt. Controlled perturbation for Delaunay triangulations. In *Proc. Symposium on Discrete Algorithms*, pages 1047–1056, 2005.

[GHS01] N. Geismann, M. Hemmer, and E. Schömer. Computing a 3-dimensional cell in an arrangement of quadrics: Exactly and actually! In *Proc. 17th Annu. ACM Sympos. Comput. Geom.*, pages 264–273, 2001.

[GO04] J. E. Goodman and J. O'Rourke, editors. *Handbook of Discrete and Computational Geometry, Second Edition*. CRC Press LLC, Boca Raton, FL, 2004.

[HHK+01] Susan Hert, Michael Hoffmann, Lutz Kettner, Sylvain Pion, and Michael Seel. An adaptable and extensible geometry kernel. In *Proc. Workshop on Algorithm Engineering*, volume 2141 of *Lecture Notes Comput. Sci.*, pages 79–90. Springer-Verlag, 2001.

[HS98] Dan Halperin and Christian R. Shelton. A perturbation scheme for spherical arrangements with application to molecular modeling. *Comput. Geom. Theory Appl.*, 10:273–287, 1998.

[KE03] Menelaos I. Karavelas and Ioannis Z. Emiris. Root comparison techniques applied to computing the additively weighted Voronoi diagram. In *Proc. 14th ACM-SIAM Sympos. Discrete Algorithms (SODA)*, pages 320–329, 2003.

[Kir83] D. G. Kirkpatrick. Optimal search in planar subdivisions. *SIAM J. Comput.*, 12(1):28–35, 1983.

[KMP+04] Lutz Kettner, Kurt Mehlhorn, Sylvain Pion, Stefan Schirra, and Chee Yap. Classroom examples of robustness problems in geometric computations. In *Proc. 12th European Symposium on Algorithms*, volume 3221 of *Lecture Notes Comput. Sci.*, pages 702–713. Springer-Verlag, 2004.

[KY06] Menelaos Karavelas and Mariette Yvinec. 2d apollonius graphs (delaunay graphs of disks). In CGAL Editorial Board, editor, *CGAL User and Reference Manual*. 3.2 edition, 2006.

[LPP04] Sylvain Lazard, Luis Mariano Peñaranda, and Sylvain Petitjean. Intersecting quadrics: An efficient and exact implementation. In *Proc. 20th Annu. ACM Sympos. Comput. Geom.*, pages 419–428, 2004.

[ORY05] Steve Oudot, Laurent Rineau, and Mariette Yvinec. Meshing volumes bounded by smooth surfaces. In *Proc. 14th International Meshing Roundtable*, pages 203–219, 2005.

[Pio99] Sylvain Pion. *De la géométrie algorithmique au calcul géométrique*. Thèse de doctorat en sciences, université de Nice-Sophia Antipolis, France, 1999.

[Pre90] F. P. Preparata. Planar point location revisited. *Internat. J. Found. Comput. Sci.*, 1(1):71–86, 1990.

[PY06] Sylvain Pion and Chee K. Yap. Constructive root bound for k-ary rational input numbers. *Theoret. Comput. Sci.*, 369:361–376, 2006.

[qi] QI: Quadrics intersection. `http://www.loria.fr/equipes/isa/qi`.

[RY06] Laurent Rineau and Mariette Yvinec. *3D Surface Mesher*, 3.2 edition, 2006. CGAL User and Reference Manual.

[SA95] Micha Sharir and P. K. Agarwal. *Davenport-Schinzel Sequences and Their Geometric Applications*. Cambridge University Press, New York, 1995.

[Sei98] R. Seidel. The nature and meaning of perturbations in geometric computing. *Discrete Comput. Geom.*, 19:1–17, 1998.

[SH02] H. Shaul and D. Halperin. Improved construction of vertical decompositions of three-dimensional arrangements. In *Proc. 18th Annu. ACM Sympos. Comput. Geom.*, pages 283–292, 2002.

[Sto91] J. Stolfi. *Oriented Projective Geometry: A Framework for Geometric Computations*. Academic Press, New York, NY, 1991.

[WFZH06] Ron Wein, Efi Fogel, Baruch Zukerman, and Dan Halperin. 2d arrangements. In CGAL Editorial Board, editor, *CGAL User and Reference Manual*. 3.2 edition, 2006.

[WFZH07] Ron Wein, Efi Fogel, Baruch Zukerman, and Dan Halperin. Advanced programming techniques applied to cgal's arrangement package. *Computational Geometry - Theory and Applications (CGTA), Special issue on CGAL*, 38(1-2):37–63, 2007.

[Yap90] C. K. Yap. A geometric consistency theorem for a symbolic perturbation scheme. *J. Comput. Syst. Sci.*, 40(1):2–18, 1990.

[Yap93] C. K. Yap. *Fundamental Problems in Algorithmic Algebra*. Princeton University Press, 1993.

[YD95] C. K. Yap and T. Dubé. The exact computation paradigm. In D.-Z. Du and F. K. Hwang, editors, *Computing in Euclidean Geometry*, volume 4 of *Lecture Notes Series on Computing*, pages 452–492. World Scientific, Singapore, 2nd edition, 1995.

[Yvi06] Mariette Yvinec. *2D Triangulations*, 3.2 edition, 2006. CGAL User and Reference Manual.