– Géométrie algorithmique –De la théorie à la pratique,Des objets linéaires aux objets courbes.

Monique Teillaud

– Géométrie algorithmique –
De la théorie à la pratique,
Des objets linéaires aux objets courbes.

Monique Teillaud

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

Soutenue le 25 septembre 2007

Rapporteurs :

Bernard Chazelle

Daniel Lazard

Chee Yap

Membres du jury :
André Galligo (Président)
Jean-Daniel Boissonnat
Daniel Lazard
Jean-Michel Moreau
Günter Rote
Chee Yap

Table des matières

Introduction

Si la communauté internationale de géométrie algorithmique a souvent la tentation de s'engouffrer dans des recherches essentiellement théoriques, et en particulier combinatoires, la grande originalité des travaux à l'INRIA résidait déjà à l'époque de mes débuts dans le souci de leur validation expérimentale et de leur applicabilité.
Le domaine a suivi globalement une évolution dans cette direction, en particulier grâce à l'*Impact Task Force Report* [C+96, C+99]. Mais notre avance sur ce point reste néanmoins certaine : notre intérêt pour le transfert technologique et industriel, ainsi que pour l'établissement d'une plateforme pour la recherche, a pris pendant ce temps une tournure encore plus concrète avec notre implication très forte dans le projet CGAL [CGA] dont notre équipe est l'un des moteurs.
Ce document prend le parti de présenter les travaux sous l'angle de cette préoccupation pratique.

La longue chaîne, entre les algorithmes et du code largement diffusé et utilisé

La liste des ingrédients qui permettent d'obtenir une bibliothèque ayant la qualité et l'impact de CGAL est longue. Ils sont tous importants, aucun ne doit être minimisé.

Rien ne peut se faire sans un bon **formalisme mathématique**, qui constitue les fondations indispensables à l'édifice. Les structures de données géométriques telles que les triangulations de Delaunay reposent en général déjà sur de bonnes bases et leurs propriétés mathématiques sont connues. Il reste cependant souvent quelques nouvelles études à effectuer pour favoriser l'adaptation de ces bases à une programmation rigoureuse.

Ensuite, une étude **algorithmique et combinatoire** des structures géométriques à calculer, des structures de données et de la complexité des algorithmes est nécessaire. Comme chacun sait, les algorithmes les meilleurs en théorie ne seront pas nécessairement programmables, ni, même lorsqu'ils le sont, les meilleurs en pratique. Un algorithme simple à programmer et aboutissant a de bons temps de calcul en pratique doit néanmoins être validé par une étude théorique qui permettra de garantir son comportement dans le pire des cas ou en moyenne, sa capacité de passage à l'échelle, et le cas échéant de définir correctement ses bonnes conditions d'utilisation.

Les choix de **représentation** des objets et structures géométriques sont en grande part guidés par l'étude combinatoire. Ils résultent souvent d'un compromis entre les temps d'accès, la place mémoire utilisée, la richesse des fonctionnalités offertes. Ces choix sont un préalable important lors de la mise en place de code géométrique.

Rapidement, apparaissent les problèmes de **robustesse**, qu'on peut ranger sous deux catégories principales. La gestion des **cas dégénérés** ne peut être négligée, car les jeux de données réels fournis par des partenaires industriels contiennent très souvent des cas tels que des points coplanaires ou cosphériques dans l'espace. La gestion des **questions numériques** est peut-être encore plus cruciale, les erreurs d'arrondis dans les calculs pouvant provoquer très fréquemment des interruptions brutales dans l'exécution des programmes [KMP+04]. Nous nous plaçons dans le modèle du calcul géométrique exact, dans lequel les prédicats géométriques, tests sur lesquels les décisions prises par les algorithmes sont basées, sont en particulier évalués de manière exacte (voir Section 1.4).

La question de l'**efficacité** des programmes complète l'ensemble. Pour des raisons évidentes, sans de bonnes performances, quelle que soit sa stabilité, un programme ne pourra jamais être utilisé en production dans une entreprise. Des méthodes de filtrage, à la fois arithmétique et géométrique, permettent de plus en plus souvent au calcul géométrique exact d'obtenir des performances comparables à celles obtenues par du calcul non certifié.

Bien sûr, la qualité de la programmation elle-même influe beaucoup sur la qualité du logiciel final. CGAL a une exigence supplémentaire, qui est la généricité, la réutilisabilité, la flexibilité du code. Pour remplir cette exigence, les choix d'**architecture** *("design")* du code doivent être effectués de façon particulièrement soigneuse.

CGAL

Je ne m'attarderai pas ici sur des aspects techniques de la programmation C++ et de CGAL, dont je ne suis certainement pas la meilleure spécialiste. On peut en trouver les idées de base dans le chapitre suivant :

---

[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.

Je décris simplement ici en quelques lignes les particularités de CGAL. Le site du projet[1] donne de plus amples informations. Le projet CGAL a formellement démarré en 1996 à l'initiative d'un consortium de quelques équipes européennes (dont notre équipe à l'INRIA)

[1] `www.cgal.org`

et a été soutenu par la Communauté européenne pendant trois ans. Il a évolué en un projet *Open Source*, autorisant une participation d'autres chercheurs. Une *start-up* INRIA, GeometryFactory[2], a été créé en janvier 2003, et vend des licenses commerciales, du support et des développements spécifiques basés sur CGAL.
Le but de ce projet est de promouvoir la recherche en géométrie algorithmique et de traduire les résultats en des programmes robustes pour des applications industrielles.
À la même époque, le *Computational Geometry Impact Task Force Report* coordonné par Bernard Chazelle insistait sur quelques recommandations [C+96, C+99]. La production de code géométrique utile et utilisable était une recommendation clef, et s'accompagnait du besoin de création d'une structure de reconnaissance pour les implantations dans le monde académique. CGAL répond du mieux possible à ces deux recommandations.
CGAL présente une forte exigence en termes de qualité, qualité qui est assurée de plusieurs manières. Une documentation complète (plus de 3000 pages actuellement) est disponible en ligne. Des tests tournent toutes les nuits sur des versions internes. Enfin, un comité éditorial a été mis en place en 2001 ; nous sommes actuellement 12 personnes dans le comité, chargé de prendre les décisions techniques, de coordonner la promotion de CGAL ; le comité évalue les nouveaux modules soumis et en effectue des relectures, dans un processus comparable au processus de soumission d'articles à des journaux ou congrès scientifiques. Ainsi, le comité joue un rôle important non seulement dans la qualité de la bibliothèque, mais aussi dans la reconnaissance des auteurs de modules, qui peuvent se prévaloir d'avoir subi ce processus de relecture approfondie avant de pouvoir «publier» leur module dans CGAL.
CGAL ne cache pas les mérites ni le travail de ses contributeurs en les attribuant à on ne sait quelle entité abstraite supérieure, au contraire la visibilité importante de CGAL rejaillit sur les participants du projet. La gratitude de la communauté envers les contributeurs est également sensible en de nombreuses occasions.

CGAL peut-être qualifié de grande première dans une communauté à l'origine plutôt théorique. L'impact de CGAL est difficilement mesurable vu le mode principal de distribution, mais tous les indices (de l'ordre de 1000 téléchargements par mois, plus de 800 abonnés sur la liste des utilisateurs et une activité importante sur la liste) tendent à faire penser que cet impact est énorme, à la fois dans la communauté mais surtout en dehors.
Cet impact légitime d'une certaine façon la recherche en géométrie algorithmique, en démontrant qu'elle ne se résume pas à un jeu intellectuel d'une poignée de chercheurs, et permet de relativiser la taille petite de la communauté (le domaine est parfois qualifié de «tête d'épingle» par des collègues sceptiques) au vu de son impact.

CGAL, recherche ou pas ?

Si la programmation, et en particulier la programmation dans CGAL avec ses exigences fortes, est extrèmement peu rentable en terme de publications, garder en tête au fil du temps la question posée dans cette section permet d'affiner une ligne directrice qui intègre naturellement le travail CGAL dans un cadre de recherche.

[2]`http ://www.geometryfactory.com/`

CGAL **est utile pour la recherche.** De nombreux travaux de recherche sont rendus possibles par l'existence de modules de base de CGAL, tels que le module de triangulation 3D largement utilisé pour les recherches en maillages et en reconstruction, parmi lesquels on peut citer deux références parmi d'autres [DG01, ORY05].

CGAL **pose de nouveaux problèmes.** Il est évident que certaines questions n'auraient jamais été posées sans la mise en place de bibliothèques telles que CGAL et CORE. En particulier, de nombreux travaux sur la robustesse, du côté arithmétique [Pio99, BEPP97, PY06] comme d'un côté plus algorithmique (voir Section 1.4) n'auraient pas vu le jour. Des travaux portant sur la recherche d'algorithmes plus simples et plus efficaces sont aussi motivés par la programmation.

CGAL **réconcilie concepts C++ et concepts mathématiques.** C'est cette thèse que je m'attache à défendre en particulier. C'est cet aspect de conceptualisation qui rend le travail CGAL réellement intéressant, et objectivement infiniment plus intéressant que le fait d'écrire des programmes qui fonctionnent. Bien sûr ce travail ne se fait pas sans efforts, efforts à la fois en termes de travail de recherche (voir par exemple Section 2.3) et en termes de travail de communication et de persuasion.

Plan du mémoire

Ce document comporte deux chapitres principaux : le premier rassemble des travaux sur les triangulations, le second présente des travaux sur les objets courbes. Ces deux chapitres se concluent par un ensemble de directions ouvertes. Le troisième chapitre survole rapidement d'autres résultats.

Chapitre 1

Des triangulations devenues pratiques

Une structure fondamentale

L'utilité pratique des triangulations, en particulier des triangulations de Delaunay, n'est plus à démontrer, pas plus que ne l'est leur intérêt théorique.
Les fondements mathématiques des triangulations de Delaunay et diagrammes de Voronoï de points sont bien connus, ainsi que leurs propriétés combinatoires, et les bornes de complexité de leur calcul dans le cas le pire et pour des distributions aléatoires. Il nous restait donc à étudier les différents autres maillons.

1.1 La randomisation, le premier maillon de la chaîne

Cette section est volontairement très courte par rapport aux années de travail qu'elle représente, car ces travaux ont déjà été présentés pour la plupart dans ma thèse de doctorat [60] ainsi que dans l'ouvrage qui en a été tiré [61]. Un état de l'art plus complet est disponible dans la littérature [Dev96].

Les publications ci-dessous sont évoquées dans cette section :

[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.
Avec Jean-Daniel Boissonnat.

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

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

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

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

ainsi qu'un travail effectué après la thèse :

[66, 62] Union and split operations on dynamic trapezoidal maps.
Computational Geometry : Theory and Applications, 17 :153–163, 2000.
Article précédé d'une version courte : In *Proc. 7th Canad. Conf. Comput. Geom.*, pages 181–186, 1995.

Alors que les analyses d'algorithmes semblaient suivre un chemin bien tracé, une nouvelle technique d'analyse d'algorithmes a été apppliquée à la géométrie à la fin des années 80 : la randomisation.
Les algorithmes concernés étaient au départ «statiques» (toutes les données devant être connues au début de l'exécution), avec le *graphe de conflits* de Clarkson [Cla87]. Les méthodes d'analyse ont rapidement permis d'évaluer des algorithmes incrémentaux «semi-dynamiques» (autorisant l'acquisition de nouvelles données au cours de l'exécution) [GKS92] et de donner (enfin) un cadre rigoureux à l'analyse de notre *arbre de Delaunay* [17, 18] pour le calcul des triangulations de Delaunay, puis d'une structure dérivée pour les diagrammes de Voronoï d'ordres supérieurs [14, 15]. Finalement, des algorithmes totalement

«dynamiques» (la suppression de données étant elle aussi permise) ont pu être développés [33, 34].
Nous avons pu abstraire du cas des triangulations le principe essentiel sous-jacent, qui consiste à maintenir l'historique de la construction au cours de l'algorithme incrémental. Nous avons ainsi obtenu et analysé une structure générale : le *IDAG (Influence Directed Acyclic Graph)*, aussi connu sous le nom de *History DAG* [12, 13].

Ces travaux sont devenus tellement classiques que le *IDAG* est présenté dans le chapitre *Randomization and derandomization* du fameux *Handbook of Discrete and Computational Geometry, Second Edition* [GO04] sans aucune citation !

Après la thèse de doctorat, notre structure de données a été enrichie d'une nouvelle possibilité : effectuer des opérations d'union et d'éclatement dans le cas de la carte des trapèzes d'un arrangement de segments, ce qui allait dans la direction d'une adaptation à des machines parallèles [62, 66].

Outre le caractère passionnant de participer à des recherches sur un sujet «chaud», l'intérêt de ces avancées tenait en l'aspect pratique des algorithmes randomisés, en particulier des algorithmes incrémentaux : ils sont simples à programmer, efficaces, et leur analyse est très réaliste, car elle ne repose sur aucune hypothèse de distribution des données.
Notons que notre structure a eu depuis une filiation, offrant de meilleures performances en pratique, en particulier la «hiérarchie de Delaunay» [Dev02], structure randomisée elle aussi, toujours utilisée dans CGAL.

Si la motivation des travaux sur la randomisation mentionnés jusqu'ici comporte un important volet pratique, il n'en est pas de même pour d'autres études. En effet, la technique de randomisation permet également de résoudre d'autres types de problèmes, plus théoriques. C'est le cas du travail suivant effectué plus récemment :

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

La randomisation est en effet réapparue fugitivement dans un travail initié lors d'un séjour à Barcelone. Le calcul de la triangulation de Delaunay de n points du plan admet une borne inférieure en $\Omega(n \log n)$. Cependant, lorsque des informations supplémentaires sont connues, on peut obtenir des algorithmes plus rapides. C'est le cas ici : étant donné deux ensembles de points, nous montrons que si la triangulation de Delaunay de leur union est connue, on peut calculer la triangulation de Delaunay de chaque ensemble par un algorithme randomisé de complexité linéaire, contrairement à ce que pouvait laisser penser la borne inférieure du cas général.

1.2 Représentation des triangulations

Cette section résume les publications suivantes :

[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.
Avec Jean-Daniel Boissonnat, Frédéric Cazals, Frank Da, Olivier Devillers, Sylvain Pion, François Rebufat et Mariette Yvinec.

[11, 16] Triangulations in CGAL.
Computational Geometry : Theory and Applications, 22 :5–19, 2002.
Avec Jean-Daniel Boissonnat, Olivier Devillers, Sylvain Pion et Mariette Yvinec.
Article précédé d'une version courte : In *Proc. 16th Annual ACM Symposium on Computational Geometry*, pages 11–18, 2000.
Avec Jean-Daniel Boissonnat, Olivier Devillers et Mariette Yvinec.

J'ai commencé à travailler sur le module CGAL de triangulations 3D en 1997. Des choix globaux d'architecture des modules et des classes avaient déjà été effectués par les développeurs CGAL de la première heure, mon travail sur ces choix a donc été minimal.
Un module de triangulations dans le plan existait déjà [Yvi06] et était une source d'inspiration solide pour le module 3D. Néanmoins, le passage du plan à l'espace a posé de nouveaux problèmes, et à cette occasion, quelques choix préliminaires faits dans le cas du plan ont été remis en question. Des discussions (forcément animées !) ont eu lieu au sein de l'équipe PRISME, aboutissant à des choix toujours valides aujourd'hui.

La décision d'ajouter un sommet a l'infini à $\mathbb{R}^2$ et $\mathbb{R}^3$ a été prise [16, 11], la triangulation calculée est donc une triangulation combinatoire de la sphère $\mathbb{S}^2$ de $\mathbb{R}^3$ (resp. $\mathbb{S}^3$ de $\mathbb{R}^4$). Cette décision a permis de résoudre définitivement le problème de gestion de la face non bornée dans le plan, dont la solution était peu satisfaisante auparavant. (Nous reviendrons sur la gestion de la topologie de l'espace triangulé dans la section 1.6.2.)
La séparation entre la triangulation combinatoire et son plongement géométrique a été renforcée à cette époque, suivant un mouvement général des modules CGAL[1].

[1]sous l'impulsion en particulier de Lutz Kettner, l'un des «dinosaures» de CGAL, qui développait le module *3D Polyhedron* et sa structure combinatoire sous-jacente *Half edge data structure*

Cette triangulation combinatoire sous-jacente est stockée en dimension 3 de manière similaire à la dimension 2, sous la forme d'un graphe dont les noeuds sont des cellules (finies ou infinies), chaque cellule ayant des pointeurs vers ses quatre sommets (finis ou infini) et ses quatre cellules voisines.
Par ailleurs, CGAL s'attache à traiter toutes les configurations sans échec, il fallait donc en particulier pouvoir traiter le cas de points tous coplanaires dans le module 3D, en calculant une triangulation dans le plan des points. Un mécanisme permettant de réutiliser les cellules et les sommets dans le cas des dimensions dégénérées, et de gérer de façon mathématiquement correcte les changements de dimension lors d'insertions ou de suppressions de points, a été mis en place [63].
Un premier module de triangulations dans $\mathbb{R}^3$ a pu être intégré à CGAL en 2000 [65, 64]. Les améliorations apportées ensuite à ce module, qu'elles soient des optimisations, ou l'ajout de nouvelles fonctionnalités, ont conduit à résoudre de nouvelles questions de recherche présentées dans les sections suivantes.

1.3 Localisation dans une triangulation

Cette section résume les publications suivantes :

[38, 37] Walking in a triangulation.
International Journal on Foundations of Computer Science, 13 :181-199, 2002.
Article précédé d'une version courte : In *Proc. 17th Annu. ACM Symposium on Computational Geometry*, pages 106-114, 2001.
Avec Olivier Devillers et Sylvain Pion.

Le problème de localisation d'un point p dans une triangulation donnée du plan est une question fondamentale en géométrie algorithmique. Un certain nombre de structures sophistiquées existent pour répondre a ces requêtes en temps optimal [Pre90, Kir83] mais elles sont souvent compliquées à coder en pratique. Il est souvent judicieux de les remplacer par des techniques plus simples, telles que des parcours utilisant les adjacences entre triangles.
Nous avons étudié plusieurs stratégies de marche dans des triangulations dans $\mathbb{R}^2$ et $\mathbb{R}^3$:

- une première stratégie consiste à visiter tous les simplexes le long d'une droite joignant un point arbitraire à la requête p [MN00], c'est la méthode employée dans le module de triangulations 2D de CGAL [Yvi06],
- une seconde stratégie suit un chemin tour à tour parallèle à chacun des axes,
- finalement, la marche par visibilité consiste à se déplacer d'un simplexe t vers l'un de ses voisins en traversant leur face commune f si l'hyperplan support de f sépare t de p.

Cette dernière stratégie est populaire pour la triangulation de Delaunay, car il a été démontré qu'elle permettait toujours d'atteindre le but [Ede90, DFNP91, GB97]. Cependant, pour une triangulation quelconque, cette marche peut entrer dans une boucle infinie.

Nous avons donc considéré une variante, la marche stochastique, dans laquelle on décide que si deux faces séparent t de p, on choisit de façon aléatoire celle que l'on traverse.
Nous avons étudié les performances de ces différentes stratégies, à la fois d'un point de vue théorique et pratique. Nous nous sommes intéressés à compter non seulement le nombre de simplexes visités par une marche, mais aussi au coût de la visite d'un simplexe, et avons considéré les problèmes de robustesse liés à ces méthodes. Cette étude nous a conduits à décider d'utiliser la marche stochastique dans le module de triangulations 3D de CGAL, car elle se comporte en pratique un peu mieux que les deux autres marches (jusqu'à 10%), et parce qu'elle est facile à coder et ne souffre d'aucun problème de cas dégénérés.
Remarquons que si la complexité d'une marche en ligne droite dans une triangulation de Delaunay 2D est connue, lorsque les points sont distribués de manière aléatoire[2], la question reste ouverte pour les marches par visibilité.

1.4 Les fameux cas dégénérés

---

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

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

Dans la littérature «classique» de géométrie algorithmique, on suppose que
- les calculs numériques sont exacts,
- les données sont en position générale,
deux hypothèses en général non vérifiées en pratique.

Nos travaux se situent dans le cadre du *calcul géométrique exact* dont C. Yap a été le pionnier [YD95]. En quelques mots, ce modèle consiste à faire en sorte que les prédicats géométriques (les tests sur lesquels les algorithmes basent leurs décisions, par exemple «ce point est-il situé à gauche ou à droite de la droite orientée définie par deux autres points ?») soient évalués de façon exacte. Ce cadre permet aux algorithmes de s'affranchir des problèmes numériques et garantit la correction de leur résultat.
Ce terme de *calcul géométrique exact* ne doit pas être assimilé à l'*arithmétique exacte* : des techniques de *filtrage* permettent, en certifiant la plupart du temps des résultats obtenus par des calculs approchés, de prendre des décisions exactes ; l'arithmétique exacte, coûteuse, n'est utilisée que dans les rares cas difficiles où la certification échoue. Ces techniques de

[2] le nombre de triangles visités pour aller de p à q est alors $O(|pq|\sqrt{n})$ [BD98]

filtre permettent donc de combiner calcul exact et efficacité. Elles ne sont pas détaillées ici [BBP01, DP03].

Dans ce cadre du calcul géométrique exact, les *cas dégénérés* peuvent être détectés : le cas de points alignés en 2D (coplanaires en 3D), si particulier pour le calcul de triangulations, correspond à l'annulation du déterminant permettant d'évaluer le prédicat d'orientation, et le cas de points cocycliques en 2D (cosphériques en 3D) à l'annulation d'un autre déterminant. CGAL traite ces cas dégénérés explicitement.
Les techniques de perturbations peuvent être classées en deux catégories : perturbations symboliques [Yap90, EM90, Sei98] et perturbations contrôlées [HS98, FKMS05]. Notre travail s'inscrit dans la première catégorie. L'idée générale consiste à rendre le problème dépendant d'un paramètre ε de manière à ce que :
- il existe $\varepsilon_0 > 0$ tel que le problème paramétré soit en position générale pour $\varepsilon \in (0, \varepsilon_0]$,
- si le problème original est en position générale, la solution du problème parametré tende vers la solution du problème original quand ε tend vers 0,
- si le problème original n'est pas en position générale, la solution du problème paramétré tende vers un résultat correct quand ε tend vers 0 par valeurs positives.

La première question consiste en fait pour nous à définir le problème à résoudre : en effet, lorsque plus de 4 points sont cosphériques en 3D, la triangulation de Delaunay n'est pas uniquement définie. Il s'agit en fait avant toute chose d'en donner une définition unique.
Différentes méthodes de perturbations pourraient être utilisées : soit les données peuvent être perturbées, soit la définition du problème peut être légèrement modifiée. Perturber les données peut avoir des inconvénients sérieux : si les points bougent de ε [ADS00], alors un tétraèdre non plat peut devenir plat à la limite (c'est-à-dire que ses quatre sommets sont exactement coplanaires). Ceci serait inacceptable autant pour les utilisateurs que d'un point de vue plus théorique, la sphère circonscrite à un tel tetraèdre n'étant même pas définie.
Edelsbrunner et Mücke écrivent qu'utiliser leur technique de perturbation dans le cas des triangulations de Delaunay est *“a real pain”* et suggèrent d'utiliser la transformation en enveloppe convexe en dimension supérieure puis de perturber le calcul de cette enveloppe convexe [EM90].
Nous proposons une variante qui consiste à perturber simplement les points relevés en 4D dans la direction verticale, les points 3D eux-mêmes ne bougeant donc pas, ce qui interdit la formation de tétraèdres plats puisque le prédicat d'orientation n'est pas modifié. Le calcul du prédicat de cosphéricité de cinq sites s_i, s_j, s_k, s_l, s_m consiste alors principalement à évaluer le signe du déterminant

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

où

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

Développer par rapport à la dernière ligne fournit un polynôme en ε dont le terme constant est le déterminant de cosphéricité non perturbé, et les coefficients sont les déterminants calculant l'orientation des points pris quatre par quatre. On peut alors démontrer par des considérations de géométrie élémentaire que ce déterminant n'est jamais nul.
La méthode a été récemment généralisée au cas de la triangulation de Delaunay à poids additif, duale du diagramme de puissance. Ceci présentait de nouvelles difficultés, car de nouveaux types de cas dégénérés ont du être considérés, et ont rendu les démonstrations de correction plus délicates.

Cette méthode, qui ne semble être un cas particulier d'aucun schéma plus général connu à ce jour, a l'avantage d'être extrêmement simple à programmer, et de n'introduire aucun nouveau type de prédicat. De plus, on remarque que tout repose sur un choix d'indexation des points, et que ce choix est quelconque. On peut donc en pratique choisir celui qui convient le mieux. L'ordre d'insertion des points était initialement utilisé dans le module CGAL, mais la version actuelle utilise l'ordre lexicographique sur les points, qui est intrinsèque.
Ce travail a permis de définir de manière déterministe une triangulation de Delaunay unique dans tous les cas, point crucial en particulier pour la fonctionnalité de suppression d'un sommet. Ainsi, CGAL est à notre connaissance le seul code disponible publiquement de triangulation de Delaunay et de triangulation de Delaunay pondérée 3D offrant cette fonctionnalité.

1.5 Modules CGAL

CGAL est le débouché naturel de nos recherches, et met également en évidence de nouveaux problèmes de recherche : avant d'essayer d'ajouter la fonctionnalité de suppression d'un sommet dans une triangulation de Delaunay 3D, je n'avais pas pensé qu'il restait une quelconque question à résoudre à ce propos et pensais n'avoir qu'un fastidieux travail de programmation à effectuer. C'est cet échange constant entre la programmation et la recherche, l'un se nourrissant de l'autre et réciproquement, qui donne à CGAL son caractère remarquable.

Les travaux résumés précédemment dans ce chapitre ont abouti aux modules CGAL suivants :

---

[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.
Avec Sylvain Pion, dont les compétences en C++ sont précieuses, et qui a rendu le code plus efficace grâce en particulier à son important travail sur les filtres arithmétiques, et une amélioration de la gestion de la mémoire.

[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.
Avec Sylvain Pion.

L'impact de ces travaux est difficile à évaluer précisément, puisque la bibliothèque CGAL est distribuée sous license *Open Source*[3] et téléchargeable sur le site du projet (qui enregistre environ 14000 téléchargements par release). Néanmoins, le volume des échanges sur la liste de discussion publique (environ 800 abonnés au total) et les utilisateurs envoyant des retours (parfois pour demander de nouvelles fonctionnalités) permettent de se rendre compte de l'importance réelle de cet impact en particulier dans la communauté académique, que ce soit en géométrie algorithmique (reconstruction de surfaces, maillages,...) ou en dehors de cette communauté (graphique, mécanique des fluides, biologie structurale, topologie,...)[4]. Notons que ces modules sont la fondation des modules de maillages de CGAL [RY06] et sont utilisés entre autres par Tamal Dey pour ses logiciels de reconstruction[5] [DG01] ainsi que par Deok-Soo Kim[6]. Par ailleurs, des licenses commerciales ont été vendues via GEOMETRYFACTORY aux compagnies pétrolières Midland Valley Exploration (Royaume Uni) et Total (France), ainsi qu'à BSAP (Suisse) pour le forage de tunnels et France Télécom et British Telecom pour le placement d'antennes.

1.6 Vers d'autres géométries

1.6.1 Espace des sphères et géométrie hyperbolique

La motivation initiale de ce travail était de généraliser au cas de coupes non parallèles l'algorithme qui permettait de calculer la triangulation de Delaunay d'un ensemble $\mathcal{S}$ de points mesurés dans deux coupes parallèles d'un objet, avec une complexité optimale $O(n \log n + t)$ où t est la taille du résultat, c'est-à-dire le nombre de tétraèdres calculés. Ceci était obtenu en superposant les diagrammes de Voronoï respectifs des deux ensembles [Boi88].

[8, 7] Output-sensitive construction of the Delaunay triangulation of points lying in two planes.

[3] QPL pour les triangulations et plus généralement la plupart des structures combinatoires, les fonctionnalités de base (le «noyau») étant elles disponibles sous LGPL

[4] La page *Projects using* CGAL `http://www.cgal.org/projects.html` tente de répertorier des utilisateurs

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

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

International Journal of Computational Geometry and Applications, 6(1) :1–14, 1996.
Article précédé d'une version courte :
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.
Avec Jean-Daniel Boissonnat, André Cérézo et Olivier Devillers.

Étant donné deux plans quelconques P et Q, construisons des sphères passant par trois points p_1, p_2 et p_3 formant un triangle de Delaunay de $\mathcal{S} \cap P$. Une telle sphère est centrée sur une droite orthogonale à P, et coupe P selon le cercle circonscrit aux trois points. Tant que le centre est suffisamment loin de Q, la sphère ne coupe pas Q, puis lorsqu'on le rapproche, elle devient tangente à Q, ensuite son intersection avec le plan Q décrit un faisceau de cercles non cocentriques, et on s'arrête dès que le centre de la sphère est tel que le cercle passe par un point q de $\mathcal{S} \cap Q$, ensuite la sphère n'est plus vide. Le faisceau de cercles est à points base si le cercle circonscrit à p_1, p_2 et p_3 coupe la droite $P \cap Q$, il est à points limites sinon. Dans les deux cas, c'est un faisceau de cercles d'axe radical $P \cap Q$. Ce point q est en fait le plus proche voisin du point de tangence, mais pour une métrique hyperbolique. L'algorithme pour les plans parallèles se généralise alors de façon immédiate en remplaçant les diagrammes de Voronoï euclidiens dans P et Q par des diagrammes de Voronoï pour une métrique hyperbolique [8].

On en déduit également un algorithme de construction de la triangulation de Delaunay de sites répartis dans k plans non parallèles : pour un triangle de Delaunay donné dans un plan, on trouve, dans chacun des k plans, un point candidat pour former un tétraèdre de Delaunay, et il ne reste plus qu'à choisir le bon point parmi ces candidats. L'algorithme obtenu a une complexité $O(kt \log n)$.

Nous avons introduit l'espace des sphères qui, sans être une notion révolutionnaire en mathématiques puisqu'on la trouve déjà dans les ouvrages de référence en géométrie de Berger [Ber77, Ber87], a permis une vision nouvelle en géométrie algorithmique. Si l'on s'intéresse au cas du plan euclidien, un cercle centré en un point (x, y) et de rayon r est associé dans l'espace des cercles au point $(x, y, x^2 + y^2 - r^2)$. Dans cet espace, les faisceaux de cercles sont des droites. Les points du plan (cercles de rayon nul) sont associés à des points du paraboloïde unité.

[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.
Version longue :
Research Report INRIA 1620, 1992.
Avec Olivier Devillers et Stefan Meiser.

Cette vision mathématique nous a permis de donner un cadre unifié à tous les résultats connus sur les diverses généralisations des diagrammes de Voronoï et la projection de niveaux dans des arrangements de dimension supérieure [ES86].

Nous avons aussi montré qu'un diagramme de Voronoï dans le plan hyperbolique pouvait facilement se déduire d'un diagramme de Voronoï dans le plan euclidien.
Plaçons-nous dans le demi-plan de Poincaré, l'un des modèles du plan hyperbolique. Dans ce modèle, le plan hyperbolique est représenté par le demi-plan $\{z \in \mathbb{C}, \mathrm{Im} z > 0\}$ ($\mathrm{Im} z$ représentant la partie imaginaire du complexe z) délimité par la droite à l'infini $\mathrm{Im} z = 0$. La médiatrice de deux points p_1 et p_2 du plan hyperbolique est une droite hyperbolique. Dans le demi-plan de Poincaré, les droites hyperboliques sont les demi-cercles (au sens euclidien) orthogonaux à la droite à l'infini.
Dans l'espace des sphères, une droite parallèle à une direction donnée représente un faisceau de cercles d'axe radical donné. Cet argument est essentiel dans la démonstration du fait qu'un diagramme de Voronoï euclidien est la projection verticale d'une enveloppe inférieure de plans tangents au paraboloïde unité. Le même argument permet de voir que, si l'on projette cette même enveloppe inférieure sur le paraboloïde dans la direction correspondant à la droite infinie du demi-plan de Poincaré, puis verticalement sur le plan, alors on obtient le diagramme de Voronoï dans le demi-plan de Poincaré.

Ce travail a également été l'occasion (trop rare) de faire des rêves sur la géométrie hyperbolique devant un tableau. Ces rêves ont été couchés sur le papier par André Cérézo dans une publication pédagogique [Cér91] (section IV-E : «au delà du miroir»), de cette écriture fine et régulière que nous n'oublions pas.
On peut en fait poursuivre la construction de la médiatrice en dehors du demi-plan de Poincaré (donc «au delà du miroir»). La courbe tracée est alors un arc d'hyperbole, prolongement imaginaire de la médiatrice des deux points p_1 et p_2.

Ces travaux sur l'espace des sphères et la géométrie hyperbolique, malheureusement jamais publiés, mériteraient très probablement d'être exploités dans CGAL.

1.6.2 De nouvelles questions

Si l'on excepte la section précédente, les travaux présentés dans ce chapitre se sont toujours situés dans le cadre de l'espace euclidien $\mathbb{R}^3$. C'est aussi le cas de quasiment toute la littérature en géométrie algorithmique. Malgré cela, d'autres espaces sont nécessaires aux applications. Par exemple, le cas d'un espace de paramètres périodique du type $[0,1] \times [0,1] \times [0,1]$ est très important pour les simulations en écoulement de fluides et en astrophysique entre autres.
Dans ces domaines d'applications, il est de coutume de dupliquer un certain nombre de points (il semble qu'approximativement 20% des points doivent être en général dupliqués) autour des bornes du domaine de référence, afin de simuler la périodicité. Ceci résulte en des surcoûts pénalisants à la fois en temps de calcul et en mémoire.
La développement de structures de données et d'algorithmes capables de représenter et de calculer directement les triangulations dans un espace ayant la topologie du tore $\mathbb{T}^3$ de $\mathbb{R}^4$

est donc un enjeu important.
Il faut noter que les triangulations de tels espaces ne sont pas toujours des complexes cellulaires, une cellule pouvant avoir plusieurs fois le même sommet et le même voisin. Des conditions d'échantillonnage permettent d'assurer que la triangulation est un complexe cellulaire.

Ces travaux aboutiront à une réorganisation complète de l'architecture des modules CGAL de triangulation 3D et à des articles. Ils ont fait l'objet du séjour post-doctoral de 6 mois[7] de Nico Kruithof en 2006 et sont en cours de finalisation avec Manuel Caroli qui commence actuellement sa thèse de doctorat.
Signalons aussi que nous avons démarré, lors du stage de Mridul Aanjaneya, une étude du calcul de triangulations dans le plan projectif [1], alors que les travaux en géométrie algorithmique se sont jusqu'ici limités au plan projectif orienté [Sto91].
Ces travaux ne sont pas plus détaillés dans ce document.

[7]Nico terminait en même temps son travail sur les *Skin surfaces*

Chapitre 2

Des courbes

Les algorithmes géométriques limitent traditionnellement leur domaine d'étude à la manipulation d'objets linéaires (points, segments, triangles) dans l'espace affine euclidien, les objets courbes étant quant à eux discrétisés par des éléments linéaires.
Traiter directement les objets courbes en tant que tels, ce qui permettrait entre autres de ne pas faire exploser l'ordre de grandeur du nombre d'objets manipulés, est l'un des grands défis de la géométrie algorithmique.
Il n'est toujours pas rare de trouver dans la meilleure littérature de géométrie algorithmique des petites phrases affirmant rapidement que l'algorithme présenté se généralise facilement au cas des courbes, phrases suivies d'une mesure de complexité théorique. Le problème crucial est alors ignoré : comment évaluer les primitives géométriques utilisées par l'algorithme ? Rappelons ici que les algorithmes auxquels nous nous intéressons supposent que les primitives soient évaluées de façon *exacte*.

Comme on le verra dans ce chapitre, une grande partie des travaux effectués jusqu'ici a porté sur l'étude des cercles, des sphères et des arcs de cercles en dimensions deux et trois. Ces cas ne semblaient pas être une finalité en soi au départ. Néanmoins, au fil de ce travail (voir Section 2.3), plusieurs applications se sont révélées, justifiant de les considérer non seulement comme une étape pour mettre en évidence et tester des méthodes avant d'étudier des courbes plus générales, mais comme un sujet important en soi. En effet, les manipulations robustes et efficaces d'arcs de cercles dans le plan sont capitales dans certains domaines indutriels tels que l'usinage et la conception de circuits imprimés ou intégrés. Les manipulations de sphères, et d'arcs de cercles dans l'espace sont quant à elles centrales dans les aspects géométriques en biologie structurale.
L'intérêt croissant récemment dans la communauté pour les approximations de courbes par des arcs de cercles [DRS06, AAH$^+$07] confirme également que cette direction de recherche était bien choisie, ce qui n'était pas nécessairement une évidence pour tout le monde au départ.

Ce chapitre évoque également des travaux plus prospectifs sur des objets plus généraux.

Les travaux sur les arrangements de quadriques, les primitives géométriques et algébriques, et le noyau CGAL, sont mentionnés dans le chapitre de livre suivant :

[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.
Avec Efi Fogel, Dan Halperin, Lutz Kettner, Ron Wein et Nicola Wolpert.

2.1 Arcs de cercles

À grand défi, attaque humble. Dès 1997, dans le projet PRISME à l'epoque, j'ai mis en place un groupe de travail «Objets courbes» réunissant quelques membres de FIABLE[1] (*action incitative* INRIA). Nous avons travaillé sur l'évaluation de primitives sur les arcs de cercles, et plus particulièrement, pour commencer, à la comparaison des abscisses de deux points définis chacun comme intersection de deux arcs de cercles, qui est l'un des prédicats les plus intéressants pour le calcul d'un arrangement d'arcs de cercles (les prédicats pour le calcul de diagrammes de Voronoï de cercles semblant largement hors de portée à l'époque).

Cette section résume les publications suivantes :

[32, 30, 31] Algebraic methods and arithmetic filtering for exact predicates on circle arcs.
Computational Geometry : Theory and Applications, 22 :119–142, 2002.
Article précédé de versions courtes :
- 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.
Avec Olivier Devillers, Alexandra Fronville et Bernard Mourrain.

La restriction du problème au cas des arcs de cercles peut sembler le rendre évident : il se réduit alors à la comparaison de racines particulières de deux équations du second degré. Une élimination manuelle (soigneuse, tout de même) des radicaux dans les expressions des racines, permettant ensuite des calculs rationnels, doit pouvoir faire l'affaire. Un certain nombre de questions intéressantes se posent néanmoins.

La première question à se poser est celle de la représentation des cercles (centre et rayon ? centre et carré du rayon ? trois points ? etc) des arcs de cercles, et des points d'intersection. Ce choix a des conséquences importantes en particulier sur la taille et le degré des expressions algébriques manipulées, et sur la longueur des nombres manipulés.

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

D'autre part, le but de notre travail était de dégager des méthodes généralisables dans le futur à des courbes de degré plus élevé.

Pour comparer les nombres algébriques, nous avons utilisé divers ingrédients algébriques [Yap93] : un résultant (plus précisément, un résultant Bézoutien) qui nous a permis de ramener le problème a l'étude du signe d'une racine d'un polynôme univarié, de degré quatre, avec pour coefficients des expressions de degrés 8 à 12 en les données (centres et rayons des cercles, sachant que les rayons ne sont connus que par leur carré) pour un degré total égal à 12 et 659 monômes en version développée ; l'introduction d'invariants nous a permis d'obtenir une expression simplifiée de ce polynôme ; finalement, nous avons utilisé la règle de Descartes pour établir une stratégie d'évaluation de signes d'expressions polynomiales en les données qui permettait de déduire la réponse. Les méthodes «habituelles» de calcul exact filtré, évoquées dans le cadre des triangulations (Section 1.4), peuvent alors être utilisées pour l'évaluation exacte et efficace de ces signes.

Un aspect intéressant de ce travail, il me semble, est l'interprétation géométrique de ces expressions polynomiales en les centres et carrés des rayons des cercles, même si cette interprétation n'est pas concrètement utilisée dans les calculs. Nous avons pu faire apparaître les différences entre les abscisses des centres des cercles, les largeurs des arcs de cercles, les puissances de certains points par rapport aux cercles, un birapport entre les projections de points d'intersection.

Cette interprétation est non seulement intéressante en elle-même, confirmant s'il en était besoin qu'algèbre et géométrie sont intimement liées, mais il me semble que de plus, une étude approfondie pourrait permettre d'améliorer la compréhension de ce type de questions, et d'établir des mécanismes de raisonnement géométrique qui pourraient venir en complément des méthodes de calcul algébrique pour résoudre ce type de problèmes. Nous n'avons cependant pas exploré cette direction.

J'évoquais plus haut notre souci de proposer des méthodes générales même pour le cas simple des arcs de cercles, afin de les appliquer ensuite à des courbes de degré plus élevé. Il faut bien avouer que nous n'avons pas réussi à le faire même pour des coniques, qui bien qu'ayant le même degré que les cercles, font intervenir des nombres algébriques de degré quatre.[2] Il s'est avéré par la suite que des méthodes basées sur des suites de Sturm plutôt que sur des résultants étaient en fait mieux adaptées à ces questions [KE03, EK06, ET04].

Cependant, ces méthodes basées sur les suites de Sturm aboutissent dans le cas des cercles à l'évaluation des signes des mêmes expressions polynomiales que les nôtres. Ceci n'a rien d'étonnant puisque, encore une fois, ces expressions ont un sens géométrique intrinsèque. Les résultats de ce travail sont au cœur de l'implantation CGAL présentée dans la Section 2.3.

[2] Deux cercles se coupent, comme des coniques quelconques, en quatre points, et la borne de Bézout est atteinte, mais deux de ces points d'intersection sont imaginaires à l'infini et l'équation polynomiale de degré quatre donnant ces points d'intersection se factorise en deux polynômes de degré deux, dont $x^2 + y^2 = 0$, qui n'a pas de racine réelle.

2.2 Arrangements de quadriques dans l'espace

Cette section résume les publications suivantes :

[47, 46] On the computation of an arrangement of quadrics in 3D.
Computational Geometry : Theory and Applications, 30 :145–164, 2005.
Article précédé d'une version courte : Sweeping an Arrangement of Quadrics in 3D.
In *Proc. 19th European Workshop on Computational Geometry*, pages 31–34, 2003.
Avec Bernard Mourrain et Jean-Pierre Técourt.

Si les implantations de calcul d'arrangements de courbes commencent à être relativement bien maîtrisées [BEH+05, WFZH07] (pour des courbes simples, en ce qui concerne la partie algébrique), il n'en est pas de même pour le cas des surfaces. La topologie de l'arrangement est alors très complexe. Des travaux définissent et analysent des décompositions spatiales, comme la décomposition cylindrique [Col75], une stratification [CEGS91], ou la décomposition verticale [SA95], mais ils ne sont pas nécessairement accompagnés d'algorithmes de construction. Des résultats plus concrets ont été fournis, allant jusqu'à une étude expérimentale pour des triangles dans l'espace, avec une analyse de complexité valable pour des surfaces algébriques générales se comportant suffisamment bien [SH02].

Une implantation complète d'arrangements de quadriques ne peut être envisagée qu'à relativement long terme. La plupart des travaux dans ce domaine se tournent vers des représentations surfaciques des arrangements, c'est-à-dire une description de l'arrangement par ses faces de dimensions inférieures ou égales à deux [GHS01]. Une paramétrisation presque optimale, en termes de radicaux, des intersections de deux quadriques a été proposée [DLLP03] et le calcul de ces intersections est implanté [LPP04, qi]. Une classification complète des intersections de quadriques a même été récemment apportée [DLLP05a, DLLP05b, DLLP05c].
Nous avons proposé un algorithme de balayage pour le calcul d'arrangements de quadriques qui, lui, adopte une approche volumique, les cellules de dimension trois étant décrites. Plus précisément, comme dans [SH02], l'algorithme de balayage calcule la décomposition verticale de l'arrangement. Nous avons porté une attention particulière aux aspects algébriques, qui est le point bloquant pour effectuer cette construction, en examinant précisement les prédicats nécessaires au balayage et le degré des nombres algébriques manipulés. Nous avons montré que cette approche nécessitait des comparaisons de nombres algébriques dont le degré peut atteindre 16, donc des calculs dans une extension de degré 256. . .
Si on restreint cette étude au cas des sphères, les nombres algébriques manipulés par cet algorithme sont de degré quatre seulement, ce qui rend déjà plus optimiste sur la possibilité d'une implantation.

De plus, dans un travail en cours avec Daniel Russel, nous proposons une nouvelle décomposition de l'espace, qui ne nécessite que des manipulations de nombres algébriques de degré deux.

Grâce à ce résultat, il est tout à fait possible d'envisager une implantation efficace, et la programmation est d'ailleurs en cours.
Ce cas des sphères, quoique particulier, est important car il permet d'espérer des utilisations en biologie structurale : la connaissance des faces de l'arrangement et le calcul ultérieur du volume de ces faces permet d'estimer certaines énergies au sein de molécules [EM86].

2.3 Un noyau courbe pour CGAL

Cette section résume les recherches sur la conception d'un noyau courbe pour CGAL.

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

Ces recherches ont abouti à l'écriture des modules suivants :

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

[69] 3D Circular Kernel.
In CGAL Editorial Board, editor, CGAL *User and Reference Manual*. Submitted.
Ce module a donné lieu à l'écriture d'un rapport :
[68] Specifications of the CGAL 3D Circular Kernel.
Research Report ACS-TR-243302-01, 2007.

L'article suivant contient des compléments récents :
[25] Design of the CGAL 3D spherical kernel and application to arrangements of circles on a sphere.
Soumis.
Avec Pedro Machado Manhães de Castro, Frédéric Cazals et Sébastien Loriot.

Le noyau CGAL regroupe des objets géométriques élémentaires à coordonnées rationnelles et principalement linéaires (points, segments, vecteurs, droites,. . .) ainsi que des calculs sur ces objets (prédicats d'intersection, d'orientation,. . .et constructions de points d'intersection, de centre de cercle circonscrit,. . .). Les cercles sont définis dans ce noyau mais n'offrent

quasiment aucune fonctionnalité. Les travaux sur les arcs de cercles mentionnés plus haut ont posé les bases pour la programmation de nouveaux types d'objets dans CGAL : des arcs de cercles, donc des points à coordonnées algébriques (de degré deux).
L'introduction de calculs sur les arcs de cercles, et donc de nombres algébriques, a montré les limites de la conception actuelle du noyau de CGAL, dans lequel il n'est par exemple pas prévu de manipuler différents types de points ayant des types de coordonnées différents. Mais la possibilité qu'il offre de l'étendre à des objets et des fonctionnalités nouvelles, sans perdre l'accès à l'existant [HHK+01] nous a permis de concevoir un noyau offrant des calculs sur des arcs de cercles, des segments de droites à extrémités non rationnelles et des points à coordonnées non rationnelles.
L'idée centrale de cette conception est la séparation entre les aspects géométriques et les aspects algébriques, de façon un peu similaire à ce qui existe dans d'autres modules séparant la structure combinatoire de son plongement géométrique. Algèbre et géométrie sont intimement liées et une simple traduction permet souvent de passer de l'une à l'autre. La séparation peut être vue ici comme la séparation entre travail sur des polynômes ou courbes algébriques (droites et cercles) et travail sur des courbes semi-algébriques (segments et arcs de cercles).
Cette séparation algèbre/géométrie est matérialisée dans le code par l'introduction d'un paramètre *template*, le noyau algébrique, dans le noyau géométrique :

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

(le paramètre `LinearKernel` étant destiné à être instancié par le noyau CGAL et en permettant l'extension grâce à une dérivation). Le paramètre `AlgebraicKernel` est chargé de fournir les opérations sur les polynômes et les nombres algébriques.
Ce travail d'implantation a mené à la soumission d'un nouveau module au comité éditorial de CGAL et à son intégration dans la version 3.2 de la bibliothèque [53].

Comme dit en introduction, si au départ le but était plus général, des applications intéressantes se sont présentées, et justifient le fait d'approfondir et d'améliorer ce résultat. En effet, une entreprise de circuits imprimés, en contact avec la start-up GeometryFactory qui distribue la license commerciale de CGAL, s'est montrée intéressée par ces manipulations robustes d'arcs de cercles, les codes industriels échouant sur une partie des jeux de données en raison de problèmes arithmétiques, et nous a fourni pour nos expérimentations quelques jeux de données réelles difficiles à traiter. Par ailleurs, Dassault-Systèmes a fait l'acquisition d'une license de recherche pour ce module.

Une interface avec le module d'arrangements de CGAL, développé à Tel-Aviv [WFZH06], nous permet d'expérimenter ce noyau sur des données synthétiques et sur les données industrielles à notre disposition. Ce travail expérimental a fait l'objet de plusieurs rapports de recherche :

---

[45] An Empirical Comparison of Software for Constructing Arrangements of Curved Arcs (preliminary version).
Research Report ECG-TR-361200-01, 2004.

Avec 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 et 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.
Avec Sylvain Pion et Ilya Suslov.

[54] Geometric filtering of primitives on circular arcs.
Research Report ACS-TR-121105-01, 2006.
Avec Sylvain Pion et 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.
Avec Pedro Machado Manhães de Castro et Sylvain Pion.

Ce travail a également fait l'objet de deux rapports
[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.
Avec Pedro Machado Manhães de Castro et Sylvain Pion.

Au cours de ce travail expérimental, diverses méthodes ont été explorées pour améliorer l'efficacité du code. Le dernier travail en particulier montre l'impact de techniques variées sur les temps de calcul obtenus. Modifier la représentation des nombres algébriques a permis de manipuler des polynômes dont les coefficients (des rationnels multi-précision) sont de taille inférieure, ce qui est une économie non seulement en mémoire mais en temps lors des opérations arithmétiques effectuées. Certaines informations ont pu être cachées dans les types de base afin d'éviter d'effectuer plusieurs fois les mêmes calculs. Enfin, des techniques de filtrage arithmétique ont été introduites dans les nombres algébriques, et des techniques de filtrage géométrique par boîtes englobantes ont été utilisées pour l'évaluation des prédicats. En résumé, ce travail a permis de diviser les temps de calculs par un facteur allant jusqu'à six. Ces changements sont disponibles dans la version 3.3 de CGAL.
Par ailleurs, la nouvelle représentation des nombres algébriques nous a permis de coder sans difficulté supplémentaire un noyau de manipulations d'arcs de cercles dans l'espace [69], qui s'interfacera tout naturellement avec le code d'arrangements de sphères en cours de programmation (Section 2.2).

Le travail effectué sur les arcs de cercles a été conduit avec un double objectif : traiter ce cas, et mettre en évidence des méthodes et des techniques se généralisant à des courbes de plus haut degré. C'est ainsi que les concepts[3] algébriques principaux dégagés pour les arcs de cercles restent centraux pour toute courbe algébrique : il s'agit de la comparaison de racines de systèmes, et de l'évaluation du signe d'un polynôme en les racines d'un système. Cette base étant posée, il faut encore écrire très précisement les spécifications du(/des) futur(s) noyaux algébriques de CGAL. Ce travail se poursuit actuellement en collaboration avec nos partenaires européens du projet ACS.

---

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

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

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

Remarquons qu'en fait ces concepts centraux sont également valables pour des fonctions implicites non algébriques, ce qui permet d'envisager des extensions encore plus générales de ces travaux à l'avenir...

En parallèle, je collabore avec Fabrice Rouillier, Sylvain Petitjean et Sylvain Lazard à une implantation en cours de noyau algébrique basée sur le code FGb/RS [fgb], dont l'auteur principal sera Luis Mariano Peñaranda.

2.4 Questions ouvertes sur les prédicats

Les travaux précédemment relatés dans ce chapitre laissent non résolues des questions sous-jacentes qui me paraissent fondamentales. Elles sont rapidement posées dans les sections 2.4.2 et 2.4.3.

2.4.1 Degré d'objets géométriques

Avant d'ouvrir ces questions, évoquons rapidement en préalable un travail sur l'évaluation du degré des offsets de courbes algébriques planes, question bien posée, contrairement à celles qui sont effleurées dans les sections suivantes.

---

[3] au sens de la *C++ Standard Template Library* : ensemble de propriétés et de fonctionnalités qu'un type doit fournir

[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.
Avec François Anton, Ioannis Emiris et Bernard Mourrain.

Nous avons obtenu une expression exacte du degré de l'offset généralisé[4] d'une courbe algébrique, en faisant intervenir les multiplicités des composantes à l'infini et des lieux singuliers. Dans le cas simple des coniques, l'offset est de degré 8 pour les ellipses et les hyperboles, 6 pour les paraboles, et 4 pour les cercles ou les paires de droites, et on peut calculer une équation implicite de cet offset.

2.4.2 Minimalité de l'ensemble des prédicats

S'il est à présent bien connu que les deux prédicats *orientation* et *in_sphere* sont nécessaires et suffisants au calcul d'une triangulation de Delaunay, la question n'est pas résolue pour certaines structures de données géométriques.
Deux questions dérivées se posent quant à l'ensemble minimal de prédicats nécessaires et suffisants pour une structure : prédicats pour définir et calculer la structure, prédicats pour un algorithme donné.

Il serait important en pratique de savoir répondre à ces questions, en particulier dans des implantations génériques telles que celles de CGAL [44]. En effet, chaque classe générique permettant de calculer une structure géométrique est paramétrée par une *Traits_class* fournissant les calculs géométriques élémentaires nécessaires à ce calcul. Une *Traits_class* est fournie par défaut, mais un utilisateur désirant fournir sa propre implantation peut le faire, en fournissant son propre modèle du concept (rappelons qu'au sens de la *C++ Standard Template Library*, un concept est l'ensemble de propriétés et de fonctionnalités qu'un type doit fournir). Il est donc important de ne pas imposer de prérequis inutilement lourds à l'utilisateur.
Par ailleurs, outre la taille de la liste des prédicats, minimiser leur complexité est primordiale pour l'efficacité du calcul. C'est ce sur quoi j'ai insisté dans une collaboration avec Tel-Aviv afin de supprimer certains prédicats coûteux, car de degré algébrique trop élevé, de la liste des prédicats demandés dans le concept de *Traits_class* pour le module d'arrangements. Les auteurs de ce module écrivent : "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,

[4]lieu des centres de cercles de rayon donné et tangents à la courbe

2006.
Avec Efi Fogel.

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

2.4.3 Degré algébrique des prédicats

Comme on l'a vu dans la section 2.2 sur l'exemple des arrangements de quadriques, ou dans la section précédente sur l'exemple des arrangements de courbes, la question du degré des prédicats associés à une structure géométrique est une mesure de complexité importante. Le degré est aussi une mesure de la précision des calculs, ou de la longueur des nombres multi-précision qui doivent être utilisés pourque ces calculs, s'ils sont effectués sur des entiers, soient exacts. Ceci dit, cette question est assez «mouvante». Elle se décline en plusieurs versions :
- Degré d'un prédicat donné. Cette question est la plus simple, car souvent, le prédicat se ramène au signe d'un polynôme, et même d'un résultant. Le résultant étant le polynôme de degré minimal exprimant une condition nécessaire et suffisante pour qu'un système ait une solution, on pourrait imaginer en allant un peu vite que le degré de ce résultant donne le degré du prédicat. Néanmoins ce raisonnement n'est pas correct, pour plusieurs raisons : d'abord, connaître le degré du résultant n'est pas immédiat, les méthodes pour le calculer calculant en fait souvent plutôt un multiple ; ensuite, le résultant peut parfois être factorisé, le degré du prédicat serait alors plutôt le maximum des degrés des facteurs ; enfin, puisqu'un facteur peut être de la forme $P^2 + Q^2$, qui a un signe constant, le degré des polynômes n'a pas une signification claire.
- Degré d'un algorithme. Cette notion est extrêmement mal définie, car elle dépend de la façon de traduire en termes algébriques les prédicats géométriques nécessaires à l'algorithme. Un exemple récent montre que ces expressions sont loin d'être uniques : dans le cas du diagramme de Voronoï de cercles, les prédicats a priori de degré 16 [KE03, EK06] ont en fait pu être traduits en expresssions complètement rationnelles [DM].
- Degré d'un problème. Ceci semble encore plus compliqué. Ceci dit, le degré nécessaire à la description de la structure est une borne inférieure pour le calcul de cette structure. Il reste que cette borne est loin d'être facile à mettre en évidence.
Par ailleurs, remarquons que cette mesure à elle seule n'est pas linéairement liée au temps de calcul que l'on peut attendre en pratique : en effet, le temps de calcul est plutôt fonction du nombre d'opérations arithmétiques nécessaires à l'évaluation du prédicat. L'implantation dans CGAL du diagramme de Voronoï de cercles [KY06] utilise ainsi des prédicats de degré 20, car les prédicats de degré 16 (degré considéré jusqu'à il y a peu comme optimal, voir ci-dessus) nécessitent 100 fois plus d'opérations arithmétiques.
Pour résumer, tout est à faire en ce qui concerne une formalisation de cette question

et l'apport, si possible, d'une définition intrinsèque à la nature géométrique des objets calculés[5].

[5] J'ai essayé de faire passer ce message au *European Workshop on Computational Geometry* en mars 2006 http://cgi.di.uoa.gr/~ewcg06/ dans mon exposé invité http://www-sop.inria.fr/geometrica/team/Monique.Teillaud/talks/EWCG.pdf

Chapitre 3

D'autres travaux géométriques et applications

Ce chapitre résume d'autres travaux sans les détailler.

3.1 Sommes de Minkowski et aménagement de satellites

[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.
Article précédé d'une version courte : Minkowski operations for satellite antenna layout.
In *Proc. 13th Annual ACM Symposium on Computational Geometry*, pages 67–76, 1997.
Avec Jean-Daniel Boissonnat et Eelco de Lange.

Ce travail est issu d'un problème industriel, et a fait l'objet d'un contrat dans lequel Matra Marconi Space nous avait chargés de proposer un logiciel d'aide a l'aménagement de satellites. La thèse d'Eelco de Lange s'est inscrite dans ce cadre, Eelco passant les trois quarts de son temps dans les locaux de Matra Marconi Space à Toulouse, communiquant constamment avec moi électroniquement et nous rendant des visites régulières à Sophia [dL98].

Le placement d'antennes et d'instruments d'observation sur les parois extérieures d'un satellite est un problème extrêmement contraint, car les satellites doivent être aussi compacts que possible. En général, une antenne est composée d'un réflecteur parabolique et d'une source qui émet ou reçoit un signal, ainsi que de leurs structures supports. Elle a également

un champ de vision, qui a grossièrement la forme d'un cône. Il ne doit pas être occulté, même partiellement.
L'aménagement est effectué en ajoutant les instruments un par un sur la structure. A chaque étape, le problème consiste à placer un instrument dans un environnement polyédrique. Nous étudions des antennes fixes. La posture d'une antenne est déterminée par un vecteur tri-dimensionnel indiquant la position d'un point de référence de l'antenne, et par un angle donnant la rotation autour de l'axe du réflecteur parabolique, qui est la direction de vision et est nécessairement fixe. Le placement a donc quatre degrés de liberté.
Cependant la rotation ainsi que la composante du vecteur translation dans la direction orthogonale au mur sur lequel on place les instruments sont moins importants, et en général une discrétisation grossière de ces deux degrés de liberté suffit. A chaque étape, nous considérons donc le placement d'un instrument en translation dans un plan.
L'ensemble des translations autorisées dans le plan (l'espace admissible) qui placent un instrument sans collision avec l'environnement est une section plane de la différence de Minkowski tri-dimensionnelle de l'environnement et de l'objet.
Bien que la taille de la section puisse être quadratique, tout comme celle de la différence elle-même, elle est couramment beaucoup plus petite. Elle peut être linéaire même lorsque la différence de Minkowski dans l'espace est quadratique.
L'environnement et l'instrument sont tous deux décomposés en unions de polyèdres convexes. La section de la différence de Minkowski de deux polyèdres convexes correspond à un chemin dans la superposition des diagrammes gaussiens des deux polyèdres sur la sphère, et ce chemin est monotone en l'azimuth. L'algorithme calcule le chemin en utilisant cette propriété.
De cette façon, la section plane d'une différence de Minkowski peut être calculée sans calculer la différence de Minkowski en entier. Nous obtenons ainsi un algorithme optimal dont la complexité dépend de la taille du résultat et qui calcule l'espace admissible.

3.2 Appariement d'objets polygonaux

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

Le problème d'appariement joue un rôle important en vision par ordinateur. Nous nous sommes intéressés à un aspect combinatoire de ce problème. Nous avons montré que le nombre de placements combinatoirement distincts d'un polygone Q à m arêtes par rapport à un polygone P à n arêtes est $O(n^2 + m^2 + \min(nm^2 + n^2m))$ et nous avons donné un

exemple montrant que cette borne peut être atteinte. Nous avons proposé un algorithme en $O((n+m)\log(n+m))$ pour calculer une translation qui maximise l'aire de la superposition entre P et Q.
Nous avons également montré que le placement de Q qui fait coïncider les centres de gravité des deux polygones réalise une superposition d'au moins 9/25èmes de la superposition maximale, et nous avons donné un exemple où cette superposition atteint 4/9èmes de la superposition maximale.

3.3 Cones et planification de trajectoires

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

Nous avons étudié deux problèmes liés à la planification de trajectoire dans le plan, pour des robots au contrôle imparfait : si le robot commence un déplacement linéaire dans une certaine direction, on sait seulement que son mouvement sera confiné dans un cône d'angle α autour de cette direction.
Pour un objectif constitué d'une seule région à l'infini, et un ensemble d'obstacles polygonaux, nous avons montré que la complexité maximale de la région à partir de laquelle l'objectif peut être atteint est $O(n/\alpha^5)$ où n est la complexité totale des obstacles. Pour une collection de k objectifs de complexité totale m, et sans obstacle, nous montrons que la taille de la région à partir de laquelle l'objectif peut être atteint est $O(k^3m)$.

3.4 Encore des sphères, et des robots

---

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

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

Le modèle géométrique direct des robots parallèles consiste à déterminer la position et l'orientation de la plate-forme lorsque sont connues les valeurs des variables articulaires servant à commander le robot. Les longueurs des segments du robot étant connues, le problème fait intervenir des intersections de sphères.
Nous avons étudié ce problème en utilisant plusieurs méthodes : géométrie dite *synthétique* (qui consiste en quelques mots à calculer les degrés algébriques des lieux géométriques des objets), et élimination symbolique. Des mesures de capteurs additionnels, qui permettent d'obtenir des informations supplémentaires, sont utilisées. Ces travaux nous ont permis d'obtenir des bornes sur le nombre de solutions de diverses variantes du problème.

3.5 Géométrie projective et étalonnage de caméras

---

[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.
Avec Jean Ponce, Kenton McHenry, Théo Papadopoulo et Bill Triggs.

Cet article, sur lequel j'ai travaillé lors d'un séjour a Urbana-Champaign, introduit le complexe quadratique absolu. Il est formé par toutes les droites coupant la conique absolue (aussi appelée *ombilic*), qui est l'ensemble des points (imaginaires) à l'infini de l'espace projectif complexe vérifiant $x^2 + y^2 + z^2 = 0$.
En utilisant l'image de la conique absolue par une caméra, on peut mettre en évidence une relation simple entre les paramètres intrinsèques de la caméra, sa matrice de projection exprimée dans une repère projectif, et la transformation reliant ce repère à un repère métrique (apparaissant dans le complexe quadratique absolu). Ceci fournit un nouveau cadre à l'auto-étalonnage.

Perspectives

> *Par ce texte élémentaire, l'auteur espère faire partager sa joie de la découverte d'une théorie mathé-matique 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]

Loin de moi, bien évidemment, l'idée de comparer ces travaux à une belle théorie mathématique... Cependant, toutes proportions gardées, je vois une certaine «beauté» dans certains axes de ces recherches.

Travailler pour la bibliothèque CGAL, ou dans le projet CGAL, est intéressant par de nombreux aspects. Si pour moi le fait de programmer n'est pas un plaisir en soi, il n'en est pas moins que CGAL apporte un certain nombre de satisfactions ; l'exigence de qualité qui peut paraître une contrainte est en fait un gain, la construction de quelque chose qui fonctionne réellement est gratifiante, ainsi que l'utilité de ce travail pour les nombreux utilisateurs (écrire du code non utilisé me paraitrait une perte de temps). Et bien sûr, la visibilité et l'impact certains que le projet CGAL apporte à ses participants n'est pas à négliger.

Plus encore, apparaît l'intérêt d'une certaine forme d'abstraction, résidant au jour le jour dans la définition de concepts au sens de la STL *(standard template library)*, mais qui à force d'approfondisssements se rapproche de la mise en évidence de concepts mathématiques sous-jacents. Bien loin de les opposer comme il est parfois malheureusement de coutume dans des communautés de programmeurs, j'insiste souvent au sein de la communauté CGAL pour en montrer les convergences.

Pour l'anecdote, j'évoquerai une conversation avec Daniel Lazard en mai 2006. Un caricaturiste nous décrirait : moi parlant de code (programmation des fonctionnalités de base sur les arcs de cercle) et lui de mathématiques ; mais en peu de temps, j'ai constaté avec bonheur que, malgré nos points de départ différents, nos discours convergeaient parfaitement : *solve* et *sign_at* sont le cœur de la solution ![1] De tels dialogues sont non seulement rassurants mais enrichissants et constructifs, plus que les affirmations sans appel parfois entendues ("The requirements for a type to be model of Kernel::Point_2 is basically to be CopyConstructible. Nothing more."), qui me semblent faire fausse route en occultant la sémantique mathématique des objets manipulés.

[1] Voir Section 2.3

Plus je programme, et plus je suis convaincue que le choix des bons concepts mathématiques est crucial pour obtenir du bon code, surtout du code visant à une certaine perennité. La programmation de concepts géométriques me semble être la source de nombreux bons sujets de recherche pour plusieurs années.

L'intérêt de ce sujet est loin d'être uniquement lié à un goût de chercheur pour un joli formalisme. Les besoins sont actuellement importants dans de nombreux domaines : les calculs de triangulations sur des domaines périodiques (tores ou cylindres tri-dimensionnels dans $\mathbb{R}^4$) sont nécessaires à la simulation[2] (Section 1.6.2), les calculs dans des espaces hyperboliques sont utilisés pour l'étude de structures cristallines[3], la géométrie projective fournit un cadre naturel à de nombreux problèmes de vision artificielle (géométrie multi-caméras et étalonnage par exemple), etc

Ce sujet large fait intervenir des domaines mathématiques et informatiques variés. On a parlé dans cet ouvrage de géométrie algébrique, de combinatoire, de géométrie projective, de programmation, d'arithmétique des ordinateurs, de géométrie hyperbolique, d'algorithmique, de topologie algébrique...[4] Cette pluri-disciplinarité rend le sujet particulièrement attractif à la fois par son intérêt scientifique et les multiples collaborations qu'il implique.

Les techniques d'algorithmique et de programmation sont à présent suffisamment au point pour permettre de faire des avancées notables dans la fourniture d'outils mathématiques efficaces et utiles.

[2]voir par exemple `http://xxx.lanl.gov/abs/cond-mat/0301378`

[3]voir par exemple `http://wwwrsphysse.anu.edu.au/~vbr110/index.php`

[4]La géométrie différentielle n'est pas apparue dans mes travaux mais est bien sûr d'une grande importance

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. Spe-

cial 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`.

[Cla87] K. L. Clarkson. New applications of random sampling in computational geometry. *Discrete Comput. Geom.*, 2 :195–222, 1987.

[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.

[DFNP91] L. De Floriani, B. Falcidieno, G. Nagy, and C. Pienovi. On sorting triangles in a Delaunay tessellation. *Algorithmica*, 6 :522–532, 1991.

[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.

[Ede90] H. Edelsbrunner. An acyclicity theorem for cell complexes in d dimensions. *Combinatorica*, 10(3) :251–260, 1990.

[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.

[GB97] Paul-Louis George and Houman Borouchaki. *Triangulation de Delaunay et maillage. Applications aux éléments finis.* Hermes, Paris, France, 1997.

[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.

[GKS92] Leonidas J. Guibas, D. E. Knuth, and Micha Sharir. Randomized incremental construction of Delaunay and Voronoi diagrams. *Algorithmica*, 7 :381–413, 1992.

[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.

[MN00] Kurt Mehlhorn and Stefan Näher. *LEDA : A Platform for Combinatorial and Geometric Computing*. Cambridge University Press, Cambridge, UK, 2000.

[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.