Exact geometric and algebraic computations in CGAL[⋆]

Menelaos I. Karavelas[†,‡]

[†]Dept. Applied Mathematics, University of Crete, GR-714 09 Heraklion, Greece
mkaravel@tem.uoc.gr
http://www.tem.uoc.gr/~mkaravel

[‡]Institute of Applied and Computational Mathematics,
Foundation for Research and Technology - Hellas,
P.O. Box 1385, GR-711 10 Heraklion, Greece

Abstract. We summarize recent progress and on-going developments for exact geometric and algebraic computations within the Computational Geometry Algorithms Library (CGAL). We detail the existing machinery used in efficient, exact and robust implementations of various geometric entities.

1 Introduction

Implementing geometric algorithms is far from being a trivial and straightforward process. Not only do we have to accommodate potentially complicated data structures, but also account for computations involving numerical data. These computations, called *predicates*, are functions of a constant number of geometric objects (e.g., points) returning a small number of values (e.g., a sign), and affect the choices made by the algorithm. The correct or consistent evaluation of predicates is vital in order for an implementation to be robust and for the outcome of the algorithm to be usable. However, even the simplest possible calculations may lead to inconsistent or wrong results [16], especially when the geometric input data are in *degenerate* or *almost degenerate* configurations.

This phenomenon that has given rise to a variety of methodologies for ensuring robustness of the implementation, while providing some guaranties for the output. These methodologies include adaptive precision floating-point arithmetic, the "topology-oriented approach", symbolic perturbations, controlled perturbations, and exact geometric computation — see [19] for a nice overview. The exact geometric computation paradigm, along with the continued evolution of software libraries offering exact number types [9,12,18], provides a platform for implementing correct and robust geometric algorithms without the need for arithmetic considerations. When combined with arithmetic and geometric filtering techniques, as well as lazy geometric evaluation considerations the overhead

⋆ Partially supported by the FP7-REGPOT-2009-1 project "Archimedes Center for Modeling, Analysis and Computation".

of exact geometric computation can turn out to be negligible, not only when taking into account the easiness of development and the correctness guarantees that it provides, but also in absolute terms.

2 The Computational Geometry Algorithms Library

The Computational Geometry Algorithms Library (CGAL) [8] is a software library for computational geometry algorithms and data structures. It started in 1995 as a European project, and currently it is a mature Open Source project. It has gone through more than 15 public releases and its latest version, version 3.6 [23], is comprised of more than 600,000 lines of C++ code. The goal of the CGAL Open Source project is "to provide easy access to efficient and reliable geometric algorithms in the form of a C++ library", as it is stated in the project web page. Due to the dual nature of geometric algorithms, namely combinatorics/data structures and computations on geometric objects, the design paradigm for the algorithms in CGAL follows a clear-cut separation between the combinatorial aspects of these algorithms and the numerical computations. This separation is evident in the template parameters CGAL classes. For example, algorithms for computing triangulated Delaunay graphs have two template parameters called respectively *(triangulation) traits* and *(triangulation) data structure*. All combinatorial operations, as well as the data structures representing the Delaunay graphs are provided by the data structure template parameter, whereas all numerical issues are abstracted and encapsulated in the traits template parameter.

Currently, the CGAL library offers algorithms for tackling a variety of geometric problems — see the Package Overview section of the latest CGAL manual [23] for a complete listing of the packages offered. Algorithms in CGAL rely on *concepts*; the library typically provides at least one *model* for each concept. This design allows for modularity and exchangeability of components, making CGAL an ideal platform for benchmarking different ways of implementing the same concept. Flexibility is gained by means of template parameters and has proven to go quite far with respect to the kind and complexity of the functionality provided to the end-user; see, e.g., the case of 2D and 3D triangulations [3], as well as that of 2D arrangements [25]. Robustness is ensured via exact predicates provided by either a kernel or a geometric traits class. The use of various arithmetic/geometric filtering techniques [5,20,24,15], and symbolic perturbations (e.g., [10]) has counter-balanced the additional cost of performing exact arithmetic in degenerate or almost degenerate configurations of the input data.

3 Algebraic computations in CGAL

During the last decade a considerable amount of work in Computational Geometry has shifted towards computing geometric entities involving non-linear geometric objects [4]. If we are to categorize these efforts, they should be split into two categories: (1) problems/implementations involving computations on algebraic numbers of small/bounded algebraic degree (2) problems/implementations involving computations on algebraic numbers of large/arbitrary algebraic degree What is common in both cases, however, is that, either at the analysis or at the implementation level, the need for support for algebraic tools became apparent.

CGAL started as a project offering geometric algorithms for simple objects, such as points, linear segments and circles; as a result the provided algorithms required support for the exact computation of quantities involving only rational operations (or limited support for square roots). CGAL's evolution over the years reflects the shift in interest towards curvilinear objects. As of release 3.0, CGAL offers packages for computing the 2D additively weighted Voronoi diagram (it requires support for algebraic numbers of degree 2) [11], and for computing arrangements of conic curves [24]. A package for computing the 2D Euclidean segment Voronoi diagram was introduced in release 3.1 (it requires support for algebraic number of algebraic degree 4) [15], while in release 3.2 we have the introduction of the 2D Circular Kernel, the first kernel designed for non-linear geometric objects [7], accompanied by a specialized, and with limited functionality, algebraic kernel for degree 2 algebraic numbers. In the same release a framework for Kinetic Data Structures was introduced [22], while the CGAL arrangements' package extended its applicability arbitrary degree Bézier curves [13]: in both cases, support for operations on large/arbitrary degree polynomials and large/arbitrary degree algebraic numbers, namely root isolation and root comparison, has been the computational core, and relied primarily on either internal (to the package) code for algebraic computations, or on algebraic numbers provided by external libraries, such as CORE [9].

A parallel effort had been underway by the EXACUS project [1]; as of CGAL's release 3.3 the two projects started to merge, and CGAL changed its underlying structure: the library moved from being number-type centric to relying on a concrete and well-designed platform of algebraic foundations [14]. In release 3.4 we have the formal introduction on polynomials and modular arithmetic, whereas on the geometry side the 2D Circular Kernel got a three-dimensional counterpart: the 3D Spherical Kernel [6]. With the latest public release 3.6, CGAL provides the first model of an algebraic kernel for arbitrary degree univariate polynomials [17] based on the RS library [21], while work for a univariate algebraic kernel based on the bitstream Descartes' algorithm and a bivariate algebraic kernel based on the curve analysis approach, is currently underway and is expected to be part of the next public release of CGAL [2].

References

1. Berberich, E., Eigenwillig, A., Hemmer, M., Hert, S., Kettner, L., Mehlhorn, K., Reichel, J., Schmitt, S., Schömer, E., Wolpert, N.: EXACUS: Efficient and Exact Algorithms for Curves and Surfaces. In: Proc. 13th European Symposium on Algorithms (ESA). pp. 155–166 (2005)
2. Berberich, E., Hemmer, M., Kerber, M.: A generic algebraic kernel for non-linear geometric applications. Research Report 7274, INRIA (2010)
3. Boissonnat, J.D., Devillers, O., Pion, S., Teillaud, M., Yvinec, M.: Triangulations in CGAL. Comput. Geom.-Theor. Appl. 22, 5–19 (2002)

4. Boissonnat, J.D., Teillaud, M. (eds.): Effective Computational Geometry for Curves and Surfaces. Springer-Verlag, Mathematics and Visualization (2006)
5. Brönnimann, H., Burnikel, C., Pion, S.: Interval arithmetic yields efficient dynamic filters for computational geometry. Discrete Appl. Math. 109, 25–47 (2001)
6. de Castro, P.M.M., Cazals, F., Loriot, S., Teillaud, M.: Design of the CGAL 3D spherical kernel and application to arrangements of circles on a sphere. Comput. Geom.-Theor. Appl. 42(6-7), 536–550 (2009)
7. de Castro, P.M.M., Pion, S., Teillaud, M.: Exact and efficient computations on circles in CGAL. In: Proc. 23rd European Workshop on Computational Geometry. pp. 219–222. Technische Universität Graz, Austria (2007)
8. Cgal, Computational Geometry Algorithms Library, `http://www.cgal.org`
9. Core Number Library, `http://cs.nyu.edu/exact/core_pages`
10. Devillers, O., Teillaud, M.: Perturbations and vertex removal in a 3D Delaunay triangulation. In: Proc. 14th ACM-SIAM Symposium on Discrete Algorithms (SODA). pp. 313–319 (2003)
11. Emiris, I.Z., Karavelas, M.I.: The predicates of the Apollonius diagram: algorithmic analysis and implementation. Comput. Geom.-Theor. Appl. 33(1-2), 18–57 (2006)
12. GMP, GNU Multiple Precision Arithmetic Library, `http://www.swox.com/gmp/`
13. Hanniel, I., Wein, R.: An exact, complete and efficient computation of arrangements of Bézier curves. In: Proc. ACM Symposium on Solid and Physical Modeling. pp. 253–263 (2007)
14. Hemmer, M.: Algebraic foundations. In: CGAL User and Reference Manual. CGAL Editorial Board, 3.6 edn. (2010), `http://www.cgal.org/Manual/3.6/doc_html/cgal_manual/packages.html#Pkg:AlgebraicFoundations`
15. Karavelas, M.I.: A robust and efficient implementation for the segment Voronoi diagram. In: Proc. International Symposium on Voronoi Diagrams in Science and Engineering (VD2004). pp. 51–62. Hongo, Tokyo, Japan (2004)
16. Kettner, L., Mehlhorn, K., Pion, S., Schirra, S., Yap, C.K.: Classroom examples of robustness problems in geometric computations. Comput. Geom.-Theor. Appl. 40(1), 61–78 (2008)
17. Lazard, S., Peñaranda, L., Tsigaridas, E.P.: Univariate algebraic kernel and application to arrangement. In: Proc. 8th International Symposium on Experimental Algotirhms (SEA). LNCS, vol. 5526, pp. 209–220. Dortmund, Germany (2009)
18. LEDA: Library of efficient data-structures and algorithms, `http://www.mpi-sb.mpg.de/LEDA/leda.html`
19. Li, C., Pion, S., Yap, C.K.: Recent progress in exact geometric computation. J. Log. Algebr. Program. 64(1), 85–111 (2005)
20. Melquiond, G., Pion, S.: Formally certified floating-point filters for homogeneous geometric predicates. Informatique Théorique et Applications 41(1), 57–69 (2007)
21. The RS library, `http://www-salsa.lip6.fr/~rouillie/Software.html`
22. Russel, D., Karavelas, M.I., Guibas, L.J.: A package for exact kinetic data structures and sweepline algorithms. Comp. Geom.-Theor. Appl. 38(1-2), 111–127 (2007)
23. The CGAL Project: CGAL User and Reference Manual. CGAL Editorial Board, 3.6 edn. (2010), `http://www.cgal.org/Manual/3.6/doc_html/cgal_manual/packages.html`
24. Wein, R.: High-level filtering for arrangements of conic arcs. In: Proc. 10th European Symposium on Algorithms (ESA). LNCS, vol. 2461, pp. 884–895 (2002)
25. Wein, R., Fogel, E., Zukerman, B., Halperin, D.: Advanced programming techniques applied to CGAL's arrangement package. In: Proc. Library-Centric Software Design Workshop (LCSD'05). pp. 24–33 (2005)