Voronoi diagrams of semi-algebraic sets

François Anton

Voronoi diagrams of semi-algebraic sets

by

François Anton

M. A.T.D.R, Université Laval, 1990

Diplôme de l'Institut de Topométrie du Conservatoire National des Arts et Métiers, France, 1992

M. Sc., Université Laval, 1995

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF

THE REQUIREMENTS FOR THE DEGREE OF

Doctor of Philosophy

in

THE FACULTY OF GRADUATE STUDIES

(Department of Computer Science)
We accept this thesis as conforming
to the required standard

The University of British Columbia

January 2004

Abstract

Most of the curves and surfaces encountered in geometric modelling are defined as the set of solutions of a system of algebraic equations and inequalities (semi-algebraic sets). Many problems from different fields involve proximity queries like finding the (nearest) neighbours or quantifying the neighbourliness of two objects.

The Voronoi diagram of a set of sites is a decomposition of space into proximal regions. The proximal region of a site is the locus of points closer to that site than to any other one. Voronoi diagrams allow one to answer proximity queries after locating a query point in the Voronoi zone it belongs to. The dual graph of the Voronoi diagram is called the Delaunay graph. Only approximations by conics can guarantee a proper order of continuity at contact points, which is necessary for guaranteeing the exactness of the Delaunay graph.

The theoretical purpose of this thesis is to elucidate the basic algebraic and geometric properties of the offset to an algebraic curve and to reduce the semi-algebraic computation of the Delaunay graph to eigenvalues computations. The practical objective of this thesis is the certified computation of the Delaunay graph for low degree semi-algebraic sets embedded in the Euclidean plane.

The methodology combines interval analysis and computational algebraic geometry. The central idea of this thesis is that a (one time) symbolic preprocessing may accelerate the certified numerical evaluation of the Delaunay graph conflict locator. The symbolic preprocessing is the computation of the implicit equation of the generalised offset to conics. The reduction of the Delaunay graph conflict locator for conics from a semi-algebraic problem to a linear algebra problem has been possible through the use of the generalised Voronoi vertex (a concept introduced in this thesis).

The certified numerical computation of the Delaunay graph has been possible

by using an interval analysis based library for solving zero-dimensional systems of equations and inequalities (ALIAS). The certified computation of the Delaunay graph relies on theorems on the uniqueness of a root in given intervals (Kantorovitch, Moore-Krawczyk). For conics, the computations get much faster by considering only the implicit equations of the generalised offsets.

Contents

List of Tables

List of Figures

List of Symbols, Nomenclature and Abbreviations

Acknowledgements

I would like to express my most profound gratitude to my supervisor Dr. David Kirkpatrick for accepting to supervise me in a research field that was far from his research interests, for his continuous support and availability, and for his invaluable help with most useful comments, questions and suggestions as well as constant interest in my research. I would never have been able to achieve my dream of working on Voronoi diagrams for curves without his invaluable supervision. I would like to show my deepest gratitude to my co-supervisor Dr. James Carrell, and the other members of my thesis committee Drs. Anne Condon and Joel Friedman for their interest in my research and their support, interesting questions, comments and suggestions.

I would like to acknowledge and express my deep gratitude for the support of Dr. Jean-Pierre Merlet during and after my visit at INRIA (Institut National de Recherche en Informatique et en Automatique) Sophia Antipolis. He introduced me to the ALIAS library for analysing and solving zero-dimensional systems of equations and inequalities and to its technicalities. I would like to express my gratitude to Drs. Ioannis Emiris, Monique Teillaud, and Bernard Mourrain for their comments and corrections, and their help both during and after my visit at INRIA Sophia Antipolis. I would like to express my gratitude to Dr. Jean-Daniel Boissonnat for his comments and interest in my research during my visit at INRIA Sophia Antipolis Computational Geometry group (PRISME project).

I would like to thank the support staff from the "Unité Mixte de Service" CNRS/Ecole Polytechnique Médicis (France) and the Department of Computer Science at UBC for their kind technical support. I am particularly grateful to Dave Brent, Michael Sanderson, Teresa Gomez-Díaz, Chris Majewski, Eric Schost, and Pierre Lafon. I would like to thank my fellow graduate students at the Theory group and Beta Lab at UBC and at the PRISME and GALAAD projects at INRIA Sophia Antipolis for their interest for and support to my research. I am particularly grate-

ful to Maja Dimitrijevic, Sanja Rogic, Julia Flötotto, Philippe Guigue, François Rebufat, David Cohen-Steiner, Drs. Ellen Gethner and Alex Brodsky, Stéphane Durocher, Dan Tulpan, Mirela Andronescu and Mark McCann.

My Ph.D. thesis research has been made possible by a Natural Sciences and Engineering Research Council of Canada (NSERC) Post Graduate Studentship (PGS B), a British Columbia Advanced Systems Institute (BC ASI) Graduate Scholarship, a "bourse doctorale" of the Fonds Chercheurs et Aide à la Recherche (FCAR), a "Bourse du Gouvernement Français" (BGF), a scholarship from the Institut National de Recherche en Informatique et en Automatique (INRIA), teaching assistant contracts from the Department of Computer Science at UBC and research assistant contracts from my supervisor Dr. David Kirkpatrick.

Finally, I would like to express my strong gratitude to my wife Dr. Darka Mioc for accepting my visit at INRIA Sophia Antipolis as well as my long research and endless computations.

François Anton

The University of British Columbia
January 2004

A Marie-Christine et à Darka.

A mes parents Roger Anton et Francisca Castro.

Chapter 1

Introduction

We can encounter almost everywhere in the real world curved objects in a three dimensional Euclidean space. Most of the curves and surfaces encountered in geometric modelling are defined as the set of common zeroes of a set of polynomials (*algebraic varieties*) or subsets of algebraic varieties defined by one or more algebraic inequalities (semi-algebraic sets). More formally, let us recall the following definition.

Definition 1.0.1. (Semi-algebraic set, adapted from [BCR98, Definition 2.1.4]) A *semi-algebraic set* of $\mathbb{R}^N$ is a subset of the form

$$\bigcup_{i=1}^{s} \bigcap_{j=1}^{r_i} \left\{ x \in \mathbb{R}^N | f_{i,j} \star_{i,j} 0 \right\},$$

where the $f_{i,j}$ are polynomials in the variables $x_1, ..., x_N$ with coefficients in $\mathbb{R}$ and $\star_{i,j}$ is either $<$ or $=$, for $i = 1, ..., s$ and $j = 1, ..., r_i$.

See [BR90, BCR98] for an introduction on semi-algebraic sets. Examples of semi-algebraic sets include Bézier, Spline curves [Baj94] in geometric modelling, Geographic Information Systems, Computer Graphics and Aeronautics; the coupler curve [Mer96] in mechanism theory, workspace and singular configurations [Mou96] in robotics, etc. Maps are unthinkable without curved objects. Many problems from

different fields involve proximity queries like finding the nearest neighbour, finding all the neighbours, or checking or quantifying the neighbourliness of two objects.

The potential applications of proximity queries on semi-algebraic sets embedded in two or three dimensional spaces include the problem of motion planning in a real environment for robots [Mou96] and Geographic Information Systems with curved objects being spatial primitives [Mio02]. The retraction planning problem [ÓY85, ÓSY86, ÓSY87] in robotics is strongly linked to questions of proximity among the projections of real-world objects in real-world environments onto the plane. The optimal path of a robot (considered as a disk) to transport the widest load while avoiding obstacles is a subset of the Voronoi diagram of those obstacles. Even if we assume that the obstacles a robot might encounter will not be curved in most of practical situations, the trajectories of the other robots or moving pieces that the robot may encounter, are curves that are expressed as semi-algebraic sets. The growth models used in several natural sciences are also strongly linked to proximity: the growth of crystal aggregates (the Johnson-Mehl model [OBS92]), the growth of trees in a forest (Voronoi diagrams [MB97]), etc. In geography, the study of influence zones and spatial analysis is also strongly linked to proximity queries on curved objects in the plane [VC90]. In all these applications, the qualitative knowledge of the neighbourliness is more critical than the quantitative knowledge of the Voronoi diagram. In crystallography, the exact knowledge of the neighbourliness is necessary for the prediction of the crystallisation process.

Voronoi diagrams are irregular tessellations of the space, where space is continuous and structured by discrete objects [AK00, OBS92]. The Voronoi diagram [Vor07, Vor08, Vor10] (see Figure 1.0.1) of a set of sites is a decomposition of the space into proximal regions (one for each site). Sites were points for the first historical Voronoi diagrams [Vor07, Vor08, Vor10], but in this thesis we will explore sets of circles, conics and more generally semi-algebraic sets. The proximal region corresponding to one site (i.e. its Voronoi region) is the set of points of the space

that are closer to that site than to any other site of the set of sites [OBS92]. We will recall now the formal definitions of the Voronoi diagram and of the Delaunay graph. For this purpose, we need to recall some basic definitions.

Definition 1.0.2. (Metric) Let M be an arbitrary set. A *metric* on M is a mapping $d : M \times M \rightarrow \mathbb{R}_+$ such that for any elements a, b, and c of M, the following conditions are fulfilled: $d(a,b) = 0 \Leftrightarrow a = b$, $d(a,b) = d(b,a)$, and $d(a,c) \leq d(a,b) + d(b,c)$. (M,d) is then called a *metric space*, and $d(a,b)$ is the distance between a and b.

Remark 1.0.3. The Euclidean space $\mathbb{R}^N$ with the Euclidean distance δ is a metric space $\left(\mathbb{R}^N, \delta\right)$.

Let $M = \mathbb{R}^N$, and δ denote a distance between points. Let $\mathcal{S} = \{s_1, ..., s_m\} \subset M, m \geq 2$ be a set of m different subsets of M, which we call *sites*. The distance between a point x and a site $s_i \subset M$ is defined as $d(x, s_i) = \inf_{y \in s_i} \{\delta(x,y)\}$.

Definition 1.0.4. (Bisector) For $s_i, s_j \in \mathcal{S}, s_i \neq s_j$, the *bisector* $B(s_i, s_j)$ of s_i with respect to s_j is: $B(s_i, s_j) = \{x \in M | d(x, s_i) = d(x, s_j)\}$ (see example on Figure 1.0.2).

Definition 1.0.5. (Influence zone) For $s_i, s_j \in \mathcal{S}, s_i \neq s_j$, the *influence zone* $D(s_i, s_j)$ of s_i with respect to s_j is: $D(s_i, s_j) = \{x \in M | d(x, s_i) < d(x, s_j)\}$ (see example on Figure 1.0.3).

Definition 1.0.6. (Voronoi region) The *Voronoi region* $V(s_i, \mathcal{S})$ of $s_i \in \mathcal{S}$ with respect to the set $\mathcal{S}$ is: $V(s_i, \mathcal{S}) = \bigcap_{s_j \in \mathcal{S}, s_j \neq s_i} D(s_i, s_j)$ (see example on Figure 1.0.4).

Definition 1.0.7. (Voronoi diagram) The *Voronoi diagram* of $\mathcal{S}$ is the union $V(\mathcal{S}) = \bigcup_{s_i \in \mathcal{S}} \partial V(s_i, \mathcal{S})$ of all region boundaries (see example on Figure 1.0.5).

Definition 1.0.8. (Delaunay graph) The *Delaunay graph* $DG(\mathcal{S})$ of $\mathcal{S}$ is the dual graph of $V(\mathcal{S})$ defined as follows:

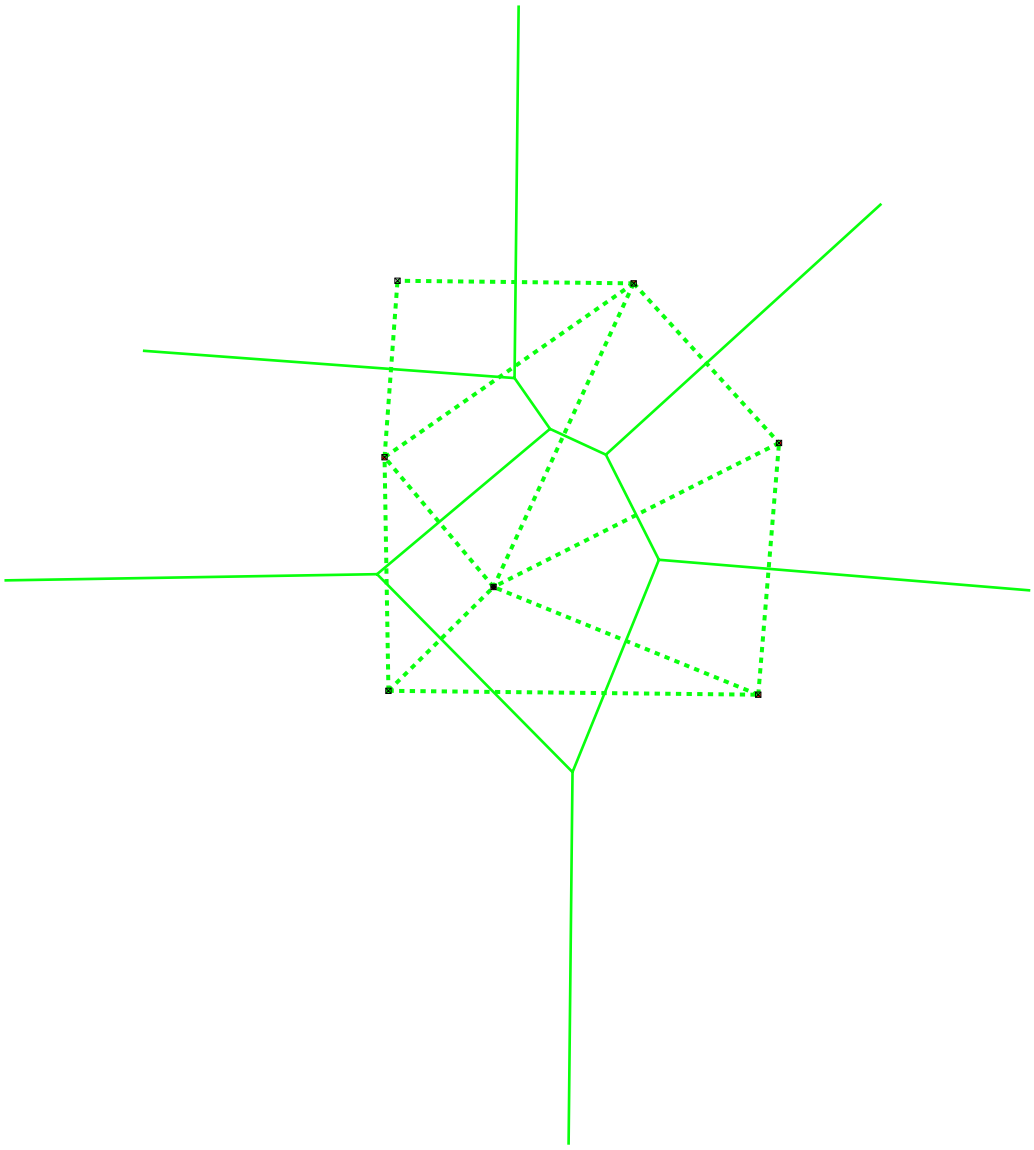

Figure 1.0.1: The ordinary Voronoi diagram (plain lines) of points (squares), and its topology expressed by the Delaunay triangulation (dashed lines)

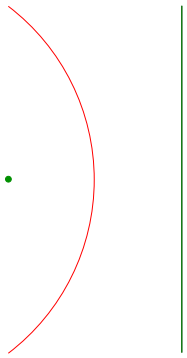

Figure 1.0.2: The bisector (parabola) of a point and a line segment

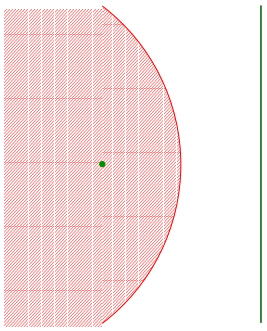

Figure 1.0.3: The influence zone (hashed) of a point with respect to a line

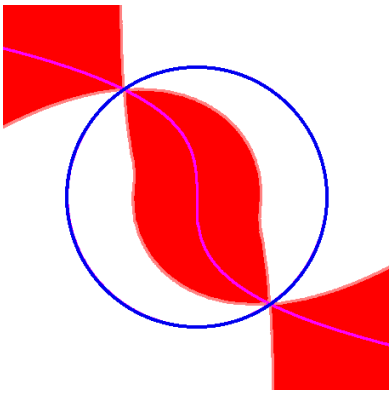

Figure 1.0.4: The Voronoi zone (dark grey) of a cubic (light grey). The two objects are a circle and a cubic.

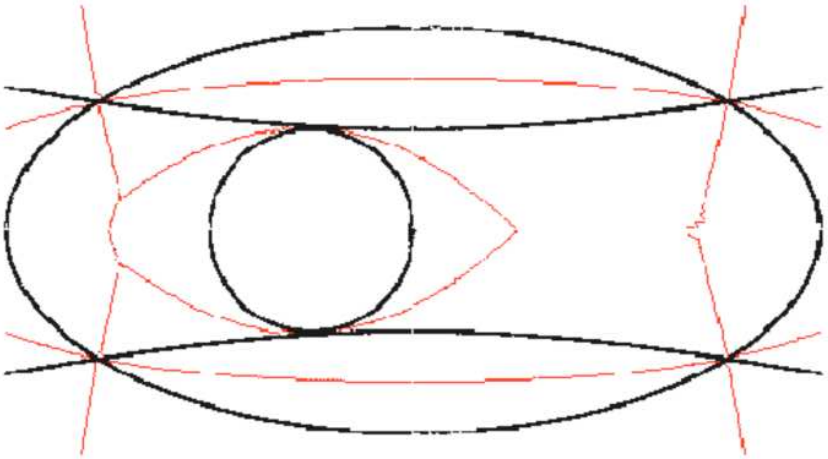

Figure 1.0.5: The Voronoi diagram (light lines) of a circle, an ellipse and a hyperbola (dark lines)

- the set of vertices of $DG(\mathcal{S})$ is $\mathcal{S}$,
- for each $(N-1)-$dimensional facet of $V(\mathcal{S})$ that belongs to the common boundary of $V(s_i, \mathcal{S})$ and of $V(s_j, \mathcal{S})$ with $s_i, s_j \in \mathcal{S}$ and $s_i \neq s_j$, there is an edge of $DG(\mathcal{S})$ between s_i and s_j and reciprocally, and
- for each vertex of $V(\mathcal{S})$ that belongs to the common boundary of $V(s_{i_1}, \mathcal{S}),\ldots,V\left(s_{i_{N+2}}, \mathcal{S}\right)$, with $\forall k \in \{1, ..., N+2\}, s_{i_k} \in \mathcal{S}$ all distinct, there exists a complete graph K_{N+2} between the $s_{i_k}, k \in \{1, ..., N+2\}$, and reciprocally (see example on Figure 1.0.6).

The one-dimensional elements of the Voronoi diagram are called Voronoi edges. The points of intersection of the Voronoi edges are called Voronoi vertices. The Voronoi vertices are points that have at least $N+1$ nearest neighbours among the sites of $\mathcal{S}$. In the plane, the Voronoi diagram forms a network of vertices and edges. In the plane, when sites are points in general position, the Delaunay graph is a triangulation known as the Delaunay triangulation. In the plane, the Delaunay graph satisfies the following empty circle criterion: no site intersects the interior of

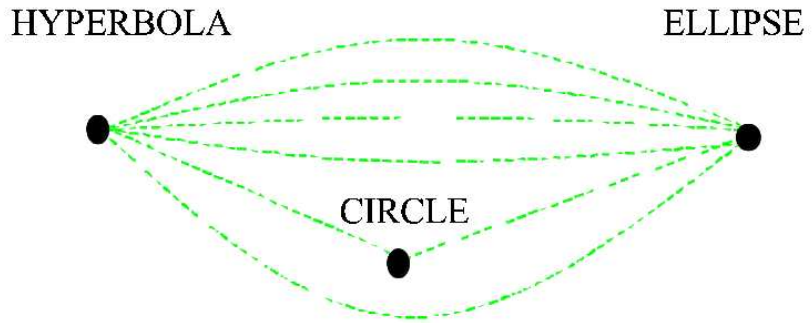

Figure 1.0.6: The Delaunay graph of the sites of Figure 1.0.5

the circles touching (tangent to without intersecting the interior of) the sites that are the vertices of any triangle of the Delaunay graph (see Figure 1.0.7).

Once the Voronoi region a query point belongs to has been identified, it is easy to answer proximity queries. The closest site from the query point is the site whose Voronoi region is the Voronoi region that has been identified. The Voronoi diagram defines a neighbourhood relationship among sites: two sites are neighbours if, and only if, their Voronoi regions are adjacent, or alternatively, there exists an edge between them in the Delaunay graph.

The certified computation of the Delaunay graph is important for two reasons. By certified computation, we mean a computation whose output is correct. First, unlike the Voronoi diagram, the Delaunay graph is a discrete structure, and thus it does not lend itself to approximations. Second, the inaccurate computation of this Delaunay graph can induce inconsistencies within this graph (see Sections 2.3 and 3.1.2), which may cause a program that updates this graph to crash. This is particularly true for the randomised incremental algorithm for the construction of the Voronoi diagram of semi-algebraic sets. The algorithm that certifies whether the facets of the Delaunay graph whose vertices are $N+1$ given sites would remain

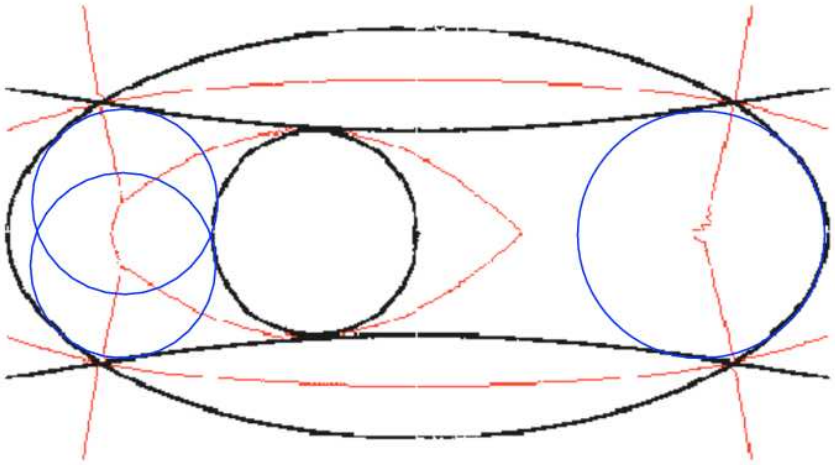

Figure 1.0.7: The 3 empty circles for the sites of Figure 1.0.5

or not after the addition of a given site is central to the design of a semi-dynamic (allowing additions of sites but not deletions) algorithm for the construction of the Voronoi diagram for semi-algebraic sets. This algorithm is called the "*Delaunay graph conflict locator*" in the reminder of this thesis. Its input is a $(N+2)$-tuple of sites, and its output is the list of all the Voronoi vertices corresponding to the $(N-1)-$dimensional facets of the Delaunay graph having the first $N+1$ sites as vertices that would not remain in the Delaunay graph after the addition of the $(N+2)^{\text{th}}$ site, and a value that certifies the presence in that list (for each Voronoi vertex). The fact the addition of the $(N+2)^{\text{th}}$ site would imply the disappearance of a Delaunay graph facet is called a conflict. Thus, it justifies the name of "Delaunay graph conflict locator". In the context of the ordinary Voronoi diagram of points in the plane, the concept that is analogous to the Delaunay graph conflict locator is the *Delaunay graph predicate*, which certifies whether a triangle of the Delaunay triangulation would remain or not after the addition of a point. The development of the Delaunay graph conflict locator is the main objective of this thesis.

The certified knowledge of the Delaunay graph for curved objects may sound like a purely theoretical knowledge that is not central in practical applications.

This is not the case in some applications. These applications include crystallography, metallography and VLSI layout. The Johnson-Mehl tessellations (which generalise several weighted Voronoi diagrams) [OBS92] play a central role in the Kolmogorov-Johnson-Mehl-Avrami [Kol37] nucleation and growth kinetics theory. The Kolmogorov theory provides an exact description of the kinetics during the heating and cooling processes in crystallography (the Kolmogorov equation [Kol37]). The certified knowledge of the neighbourliness among molecules is central to the prediction of the formation of crystal aggregates. In metallography, the analysis of precipitate sizes in aluminium alloys through Transmission Electronic Microscopy [Des03, Section 1.2.2] provides an exact measurement of the cross sections of these precipitates when they are rodes with a fixed number of orientations [Des03, Section 1.2.2]. In VLSI design, the second order Voronoi diagram of the layout is used in the computation of the critical area, a measure of a circuit layout's sensitivity to spot defects [CPX02, Section 1]. An important concern on critical area computation is the robustness [CPX02, Section 1].

In the context of the application to robotics, one might object that the real-world objects a robot may collide with are approximate because of their manufacturing process. This is indeed true, but their specification is exact and can be expressed as semi-algebraic sets owed to the possibility of algebraic translation of the geometric specification of the methods of manufacturing process. Indeed, the mechanical manufacturing process methods such as turning, countersinking, sharpening, drilling, bending, roll bending, or pressing can be expressed as geometric transformations, which in turn can be expressed algebraically. The finished result (mechanical component) is specified as a semi-algebraic set.

Another limitation of approximative algorithms for the computation of the Delaunay graph is that when approximate computations are performed on objects defined approximately (within some geometric tolerance), the propagation of the errors can be critical, especially if the final computation involves approximate in-

termediary computations. Interval analysis allows one to certify computations on objects defined approximately. Indeed, interval analysis allows one to consider objects defined approximately since these objects can be specified by intervals. Interval analysis allows one to certify computations on intervals by providing bounds on the results.

Finally, on another hand, the certified computation of the Delaunay graph participates to the recent move in the development of numerical and simulation software as well as computer algebra systems to exact systems [BCSS98].

1.1 The problem

The proximity queries stated above could be effectively answered if the Delaunay graph for sets of geometric objects could be computed in an efficient and certified way. This would require the embedding of the Delaunay graph and the location of the query point in that embedded graph. The embedded Delaunay graph and the Voronoi diagram are dual subdivisions of space, which can be stored in a quad-edge data structure [GS85].

The first and most explored Voronoi diagram is the Voronoi diagram for a set of points [Vor07, Vor08, Vor10] in the Euclidean plane or in the three-dimensional Euclidean space (see Figure 1.0.1). Voronoi diagrams have been generalised in many different ways including by modifying the space in which they are embedded (see [Aur87, OBS92] for a general survey of Voronoi diagrams): higher dimensional Euclidean spaces, non Euclidean geometries (e.g. Laguerre geometry, hyperbolic geometry, etc.). Fewer generalisations of Voronoi diagrams correspond to extending the possible sites from points to geometric objects. The only generalisations of this kind that have been explored are "abstract Voronoi diagrams" [Kle89] and the Voronoi diagram for lines [OBS92]: (the sites are points, line segments, circular arcs or piecewise analytic curves). The Voronoi diagram for curved objects

does not necessarily satisfy the property of the abstract Voronoi diagrams. *Abstract Voronoi diagrams* are two-dimensional Voronoi diagrams defined by topological properties [Kle89] (which is that the bisector of any pair of sites is an unbounded simple curve). Therefore, the Voronoi diagram of N curved objects cannot be computed with the randomised $O(N \log N)$ algorithm for the construction of abstract Voronoi diagrams [Kle89]. The Voronoi diagrams for lines that have already been studied are the Voronoi diagram for circles (set of sites comprising circles) [KKS01b, KKS01a, KKS00], the Voronoi diagram for sets of points and straight line segments, the Voronoi diagram for sets of points, line segments and circular arcs [Yap87], and the Voronoi diagram for planar domains with curved boundaries (piecewise analytic) [RF99a, RF99b]. In principle, the Voronoi diagram can be generalised to sets of sites comprising more general geometric objects (especially general curved objects). We will refer to these generalisations of the Voronoi diagram as the Voronoi diagram of sets of geometric objects (points, curves, surfaces; see Figure 1.0.5).

These Voronoi diagrams of sets of geometric objects have been far less explored, and they have been computed only approximately: as approximation algorithms decomposing the objects by points sets [OBS92] or approximating the computation of Voronoi vertices by a Newton-Raphson scheme for curves with rational parameterisation [RF99a, RF99b]. By computation of the Voronoi diagram, we mean computation of the coordinates of the Voronoi vertices, of the equations of the Voronoi edges, and of the network formed by theses vertices and edges. The first type of approximation algorithms for constructing Voronoi diagrams is not guaranteed to give topologically correct results, because the Voronoi diagram is very sensitive to the order of continuity at contact points (see Section 2.3 and [RF99a]). The second type of approximation algorithms for constructing Voronoi diagrams does not directly address the exactness of the Delaunay graph, because the basic compu-

tation is the computation of the Voronoi vertex instead of the computation of the Delaunay graph predicate, and this predicate was not addressed in [RF99a, RF99b]. The neighbourhood relationship among sites is addressed indirectly through the identification of the Voronoi vertices and their classification. The curves that they addressed are parametric curves admitting rational parameterisations (i.e. ratios of polynomials), and therefore, it excludes conics (see Section 2.3). Moreover, their computation of the Voronoi vertices uses basic techniques (projective resultants) of higher complexity because these techniques consider common zeroes in the projective space instead of in the affine space or in the algebraic torus $(\mathbb{C}^\star)^N$. This difference in complexity is explained later in the section on sparse elimination (see Section 3.3.1). This higher complexity of the algebraic computation techniques used is particularly significant because those projective resultant computations are not algebraic precomputations done only once, but computations done each time a bisector is computed. The algorithm for constructing the Voronoi diagram for points, line segments and circular arcs proposed in [Yap87] proceeds by a divide-and-conquer paradigm using vertical slabs, which excludes an incremental construction of the Delaunay graph.

The computation of the Delaunay graph conflict locator for conics and more generally for semi-algebraic sets is the main problem that is being addressed in this thesis. The Delaunay graph conflict locator is the basic tool for maintaining the Delaunay graph when the curved objects are introduced sequentially. A direct application of this Delaunay graph conflict locator is a randomised incremental algorithm for the construction of the Voronoi diagram of semi-algebraic sets. This algorithm could be used for maintaining a semi-dynamic (allowing only insertions but not deletions) Voronoi diagram for semi-algebraic sets. Such a Voronoi diagram could be stored with the embedded Delaunay graph in a quad-edge data structure [GS85]. We will see how such an algorithm can be designed from the Delaunay

graph conflict locator in the following section.

1.2 The motivation

In this section, we will examine how the Delaunay graph conflict locator can be used to maintain the Voronoi diagram of semi-algebraic sets in the plane as those semi-algebraic sets are introduced one by one. Finally, we will ennounce a necessary and sufficient condition for the connectivity of the Voronoi diagram of semi-algebraic sets in the projective plane, that has a direct application in the representation of spatial data at different resolutions.

Knowing the Voronoi diagram $V(\mathcal{S})$ of a set $\mathcal{S}=\{s_1,\ldots,s_m\} \subset \mathbb{R}^2$ of at least two semi-algebraic sets ($m > 1$) and its embedded Delaunay graph $DG(\mathcal{S})$ stored in a quad-edge data structure, we would like to get the Voronoi diagram $V(\mathcal{S} \cup \{s_{m+1}\})$, where s_{m+1} is a semi-algebraic set of $\mathbb{R}^2$. In all this section, we will say that a circle $\mathcal{C}$ *touches* a semi-algebraic set s_i if, and only if, $\mathcal{C}$ is tangent to s_i and no point of s_i is contained in the interior of $\mathcal{C}$. The Voronoi edges and vertices of $V(\mathcal{S})$ may or may not be present in $V(\mathcal{S} \cup \{s_{m+1}\})$. Each new Voronoi vertex w induced by the addition of s_{m+1} necessarily belongs to two Voronoi edges of $V(\mathcal{S})$, because two of the three closest sites to w necessarily belong to $\mathcal{S}$. The new Voronoi edges induced by the addition of s_{m+1} will clearly connect Voronoi vertices of $V(\mathcal{S})$ to new Voronoi vertices induced by the addition of s_{m+1} or new Voronoi vertices between themselves. Any of these later Voronoi edges e' must be incident with one of the former Voronoi edges at each extremity of e' (because the Voronoi vertex at each extremity of e' belongs to only one new Voronoi edge, i.e. e'). Any of the former Voronoi edges e must be a subset of a Voronoi edge of $V(\mathcal{S})$, since e must be a new Voronoi edge between sites of $\mathcal{S}$ (otherwise the Voronoi vertex belonging to $V(\mathcal{S})$ at one of the extremities of e by the definition of e would be a new Voronoi vertex). Thus, to get $V(\mathcal{S} \cup \{s_{m+1}\})$, we need to know which Voronoi vertices and edges of $V(\mathcal{S})$ will not be present in $V(\mathcal{S} \cup \{s_{m+1}\})$, which Voronoi edges of $V(\mathcal{S})$

will be shortened in $V(\mathcal{S} \cup \{s_{m+1}\})$ and which new Voronoi edges will connect new Voronoi vertices between themselves.

We can test whether each Voronoi vertex v of $V(\mathcal{S})$ will be present in $V(\mathcal{S} \cup \{s_{m+1}\})$. Let us suppose that v is a Voronoi vertex of s_i, s_j and s_k. v will remain in $V(\mathcal{S} \cup \{s_{m+1}\})$ if, and only if, no point of s_{m+1} is contained in the interior of the circle centered on v that touches s_i, s_j and s_k. This is a sub-problem of the Delaunay graph conflict locator that can be tested by giving s_i, s_j, s_k and s_{m+1} as input to the Delaunay graph conflict locator, and then retain only the solutions where the Voronoi vertex is v.

We can test whether each Voronoi edge e of $V(\mathcal{S})$ will be present in $V(\mathcal{S} \cup \{s_{m+1}\})$. Let us suppose that e is a locus of points having s_i and s_j as closest sites. e will disappear entirely from $V(\mathcal{S} \cup \{s_{m+1}\})$ if, and only if, a point of s_{m+1} is contained in the interior of each circle centered on e and touching s_i, s_j and each common neighbour s_k to s_i and s_j in $DG(\mathcal{S})$ in turn. This can be tested by giving s_i, s_j, s_k and s_{m+1} as input to the Delaunay graph conflict locator and then retaining only the solutions where the Voronoi vertex belongs to e. e will be shortened (possibly inducing one or more new Voronoi edges) in $V(\mathcal{S} \cup \{s_{m+1}\})$ if, and only if, there exists Voronoi vertices of s_i, s_j and s_{m+1} on e and there is no point of any common neighbour s_k to s_i and s_j in $DG(\mathcal{S})$ in the interior of a circle centered on e and touching s_i, s_j and s_{m+1}. The centre of each one of such circles will be a new Voronoi vertex in $V(\mathcal{S} \cup \{s_{m+1}\})$. This can be tested by giving s_i, s_j, s_{m+1} and s_k as input to the Delaunay graph conflict locator and then retaining only the solutions where the Voronoi vertex belongs to e.

This may not be the best way to proceed, but the Delaunay graph conflict locator is sufficient to maintain the Voronoi diagram of semi-algebraic sets. Tests might be limited to edges and vertices on the boundaries of the Voronoi regions $V(s_i, \mathcal{S}), s_i \in \mathcal{S}$ that intersect s_{m+1} and of the Voronoi regions $V(s_j, \mathcal{S}), s_j \in \mathcal{S}$

adjacent to a Voronoi region $V(s_i, \mathcal{S})$. Indeed, a point (and thus a semi-algebraic set) can steal its Voronoi region only from the Voronoi region it belongs to and the adjacent Voronoi regions.

We will finish this section with a necessary and sufficient condition for the connectivity of the Voronoi diagram of connected semi-algebraic sets in the projective plane. This result allows the characterisation of dangling edges in the Delaunay graph corresponding to the presence of closed edges in the Voronoi diagram. In order to proceed, let us recall some notations used in point set topology: let $\overline{s}$ denote the closure of s, and $\overset{\circ}{s}$ denote the interior of s in the sense of the point set topology in $\mathbb{R}^2$. Note that if s bounds a closed domain then the interior of s is meant to be the interior of the closed domain bounded by s.

Proposition 1.2.1. *(Connectivity of the Voronoi diagram in the plane) The Voronoi diagram $V(\mathcal{S})$ of a set $\mathcal{S} = \{s_1, \ldots, s_m\} \subset \mathbb{R}^2$ of at least two connected semi-algebraic sets ($m > 1$) considered in $\mathbb{P}^2$ is not connected if, and only if, there exist a subset I of $[1, \ldots, m]$ and one index j of $[1, \ldots, m]$ such that $\forall i \in I, s_i \subset \overset{\circ}{s_j}$ and $\forall k \in [1, \ldots, m] \setminus I, \overline{s_i} \cap \overline{s_k} = \overline{s_j} \cap \overline{s_k} = \emptyset$.*

Proof. If: Assume there exist a subset I of $[1, \ldots, m]$ and one index j of $[1, \ldots, m]$ such that $\forall i \in I, s_i \subset \overset{\circ}{s_j}$ and $\forall k \in [1, \ldots, m] \setminus I, \overline{s_i} \cap \overline{s_k} = \overline{s_j} \cap \overline{s_k} = \emptyset$. Let $s_l \in \mathcal{S}$ with $l \in [1, \ldots, m] \setminus I$. Let $S = \bigcup_{i \in I} s_i$. Since $S \subset \overset{\circ}{s_j}$, any circle touching both a $s_i, i \in I$ and s_j must be contained in $\overline{s_j}$. Since $\overline{S} \cap \overline{s_l} = \overline{s_j} \cap \overline{s_l} = \emptyset$, no circle can touch each of a $s_i, i \in I$, s_j and s_l. Thus, there is no point that has a $s_i, i \in I$, s_j and s_l as nearest neighbours. Thus, there is no Voronoi vertex of a $s_i, i \in I$, s_j and s_l. Since there is no Voronoi vertex of a $s_i, i \in I$, s_j and an s_l with $l \in [1, \ldots, m] \setminus I$, there are no Voronoi vertices on the bisector of S and s_j. Since $\overline{S} \cap \overline{s_l} = \overline{S} \cap \overline{s_l} = \emptyset$, any circle centred on the bisector of S and s_j and touching both S and s_j does not intersect any site s_k with $k \in [1, \ldots, m] \setminus I$. Thus, the bisector of S and s_j is contained in $V(\mathcal{S})$. Since s_j is connected and $S \subset \overset{\circ}{s_j}$, the bisector of S and s_j is a closed curve.

Thus, the Voronoi diagram of $\mathcal{S}$ is not connected in $\mathbb{P}^2$.

Only if: Assume the Voronoi diagram of $\mathcal{S}$ is not connected in $\mathbb{P}^2$. Then, $V(\mathcal{S})$ has at least two connected components. Thus, at least one of these connected components does not have points at infinity. Let us consider the connected component (let us call it C_1) that does not have points at infinity. Since C_1 is composed of Voronoi edges[1], each edge in C_1 must end at either a Voronoi vertex or a point at infinity. Since C_1 does not have any point at infinity, all Voronoi edges in C_1 connect Voronoi vertices. Thus C_1 is a network of vertices and edges linking those vertices. The regions that this network defines are Voronoi regions. Let $\mathcal{D}$ be the union of the closure of those Voronoi regions. $\mathcal{D}$ is a closed set set by its definition. Let us consider now the semi-algebraic sets $s_l, l \in L$ whose Voronoi regions are contained in $\mathcal{D}$. Let $S = \bigcup_{l \in L} s_l$. From the definition of a semi-algebraic set, its is straightforward that the union of two semi-algebraic sets is a semi-algebraic set. Thus S is a semi-algebraic set. We will now consider S as a site instead of each one of the $s_l, l \in L$. The influence zone of $S = \bigcup_{l \in L} s_l$ is clearly $\overset{\circ}{\mathcal{D}}$, because the influence zone of a union of semi-algebraic sets is clearly the closure of the union of the Voronoi regions of those semi-algebraic sets. Let $e = \partial\mathcal{D}$. It is a portion of the bisector of S and another semi-algebraic set. Let us call it s_j. If not all the bisector of S and s_j was contained in $V(\mathcal{S})$, then e would end at Voronoi vertices (a point on the Voronoi diagram has at least two closest sites) or the point at infinity, a contradiction with e not being connected. Thus, the bisector of S and of s_j is contained in $V(\mathcal{S})$, and it is equal to e. By the definition of e, e must be a closed curve. Assume the positions of S and s_j with respect to e are not always the same. Then, S and s_j must intersect. The bisector of S and s_j must have two branches near the intersection points (see Figure 1.2.1). Since e is a closed curve and S is contained in the interior of e, s_j must be closed, and the other branches must be unbounded (a contradiction with e not being connected in $\mathbb{P}^2$). Thus, the positions

[1] a one-dimensional component of the Voronoi diagram, which is also the locus of points having two nearest sites

of S and s_j with respect to e are always the same along e. Since s_j is connected, S is contained in the interior of e and the positions of S and s_j with respect to e are always the same along e, $S \subset \overset{\circ}{s_j}$. Since e is the bisector of S and s_j and belongs to $V(\mathcal{S})$, any circle centred on e and touching both S and s_j does not intersect any site s_k with $k \in [1, \ldots, m] \setminus I$. Thus, $\forall k \in [1, \ldots, m] \setminus I, \overline{s_i} \cap \overline{s_k} = \overline{s_j} \cap \overline{s_k} = \emptyset$.

□

The only cases of disconnected (considered in $\mathbb{P}^2$) Voronoi diagrams correspond to one or more sites (semi-algebraic sets) contained in the interior of another site. This property has a direct application in Geographic Information Systems. When the same region $\mathcal{R}$ bounded by a semi-algebraic set S is represented at different scales, the representation of the details inside $\mathcal{R}$ does not change the Voronoi diagram outside $\mathcal{R}$. The edges of the Delaunay graph corresponding to a disconnected Voronoi diagram (considered in $\mathbb{P}^2$) are respectively dangling edges or cut edges (the Delaunay graph is not bi-connected and removing a cut edge induces two connected components). It is possible to detect if there exists one or more sites $s_i, i \in I$ contained in the interior of another site s_j by checking that there exists no Voronoi vertex of s_i, s_j and any $s_k \in \mathcal{S}$ distinct from s_i and s_j. This is a subproblem of the Delaunay graph conflict locator.

1.3 Outline of the research

The main theoretical objectives of this thesis are the determination of a general formula for degree of the *offset* to (i.e., the locus of points at a given distance from) an algebraic curve, and the reduction of the Delaunay graph conflict locator for algebraic curves from a semi-algebraic problem to a linear algebra problem (computing the eigenvalues of a matrix). The main practical objectives of this thesis are the computation of an implicit equation of the *generalised offset* (i.e., the locus of centres of circles of a given radius tangent) to a conic defined by a formal polynomial, the

Figure 1.2.1: The relative position with respect to the bisector must be constant

exact symbolic computation of the sparse resultant matrix for the Delaunay graph conflict locator for conics, and the certified computation of the Delaunay graph conflict locator for conics and for semi-algebraic sets. In all the computations, the central ideas were to use the lightest computational techniques and to simplify the formalisation of the solutions.

The original contributions in this thesis are as follows.

1. A general formula for the degree of the generalised offset to an algebraic curve has been determined by studying the algebraic properties of these offsets. The knowledge of the degree of the generalised offset to a conic is used in the identification of the implicit equation of the generalised offset to a conic as a factor of a sparse resultant;

2. The number of points on which the Delaunay graph conflict locator for conics is evaluated has been computed. It corresponds to the number of lines of the matrix whose eigenvalues need to be computed for the algebraic computation of the Delaunay graph conflict locator for conics;

3. The Voronoi diagram and the Delaunay graph for circles have been computed exactly and symbolically through a completely symbolic conflict locator;

4. The computation of the Delaunay graph of conics has been reduced from a semi-algebraic problem to a linear algebra problem. The matrix for which the eigenvalues of a Schur complement of one of its submatrices give the answer to the Delaunay graph conflict locator for conics has been computed symbolically;

5. The computation of the Delaunay graph of semi-algebraic sets embedded in a two-dimensional space has been done using ALIAS. Although there is no known lower nor upper bound for this problem, the running time is satisfactory for exploration purposes.

We have also outlined how the Delaunay graph conflict locator could be used for the incremental construction of the Voronoi diagram of semi-algebraic sets. We have also proved a necessary and sufficient condition of connectivity of the Voronoi diagram for semi-algebraic sets in the plane, which has a direct application in Geographic Information Systems.

The originality of this contribution resides in the formalisms that have been used (the generalised offset, see Chapter 4) and introduced (the generalised Voronoi vertex, see Chapter 5), and in the fact this work represents the first computation of the Delaunay graph conflict locator for general semi-algebraic sets.

The thesis contents are organised as follows. Chapter 2 reviews the generalisations of the Voronoi diagram for curved objects. Voronoi diagrams for general semi-algebraic sets have not been studied previously. The problems closest to the Voronoi diagram for semi-algebraic sets are the Voronoi diagrams for manifolds, studied by Devillers et al. [DMT92] in 1992, the Voronoi diagram for curved objects, studied by Alt and Schwarzkopf [AS95] in 1995, and the Voronoi diagram for planar domains with curved boundaries [RF99a, RF99b].

Chapter 3 introduces the approach used to design the computation of the Delaunay graph conflict locator for semi-algebraic sets. The approach used in this thesis involves combining symbolic precomputations and numerical computations to find the fastest computation of that conflict locator. The central tools for formalising the Delaunay graph conflict locator are the generalised offset (the locus of points that are locally at a given distance from a given geometric object [ASS99]) and the generalised Voronoi vertex (a concept introduced in this thesis: see Section 5.2).

With regard to symbolic computing, the approach includes working with the "geometry" of the monomials (see Section 3.3.1) composing a polynomial (i.e. representing the exponents of the monomials appearing in a polynomial as points in

an $N-$dimensional space, where N is the number of variables we wish to eliminate) and the sparse resultant algorithms of Emiris [EC95, CE00] and of Singular [GPS01]. This is used to get an implicit equation of the generalised offset to a conic (see Section 4.4), and the matrix for which the eigenvalues of a Schur complement give the answer to the Delaunay graph conflict locator for conics (see Chapter 5).

With respect to scientific computing, the interval analysis and consistency methods implemented in ALIAS [Mer00] have allowed us to design and implement the Delaunay graph conflict locator for semi-algebraic sets and to obtain another Delaunay graph conflict locator for conics (by applying the interval analysis and consistency methods to the implicit equation of the generalised offset to a conic). Matlab eigs function has been used to compute the eigenvalues of the matrix obtained through symbolic computing for the Delaunay graph conflict locator for conics

Chapter 4 studies the algebraic properties of the offset to an algebraic curve. From these algebraic properties, we have obtained a general formula for the degree of the offset to an algebraic curve. We applied this formula to conics in order to get the degree of the offset to conics. We used the sparse resultant algorithm of Emiris [EC95, CE00] and the geometry of monomials to get an implicit equation of the generalised offset to a conic.

Chapter 5 presents the algebraic computation of the Delaunay graph conflict locator for conics. Recent theoretical achievements from Algebraic Geometry and Computational Algebraic Geometry allow one to solve systems of algebraic equations by solving a linear algebra problem (computing eigenvalues [CLO98]). The eigenvalue problem has been studied for a long time and it is computationally tractable [CD00]. By computing in the quotient algebra [Lan02, Section 3, Chapter II] of the ring of polynomials by the ideal corresponding to a zero-dimensional algebraic variety, it is theoretically possible to transform a complex problem of resolution of

a system of algebraic equations into the more tractable and explored linear algebra problem of finding eigenvalues. Indeed, the quotient of the ring of polynomials in the variables of the problem by the ideal corresponding to the zero-dimensional variety object of the study is a finite dimensional vector space. Thus, every linear mapping can be expressed by a matrix in any (finite) basis of the quotient algebra. Moreover, the values of a given polynomial g at the points of the zero-dimensional variety correspond to the eigenvalues of the matrix of the map corresponding to the multiplication by g (see Theorem 4.5 on page 54 in [CLO98]). This is the basis of the symbolic part of the Delaunay graph conflict locator for conics. The sparse resultant matrix corresponding to that conflict locator is a sparse matrix, and sparse methods for eigenvalues [Pin02, Nik00, vdV99, CD00, Krä92] can be applied on it.

Chapter 6 presents the interval analysis based computation of the Delaunay graph conflict locator for semi-algebraic sets. Interval analysis provides a more general approach for solving systems of algebraic equations and inequalities. Some new tools have appeared recently for solving systems of equations and inequalities with real coefficients based on interval analysis (see ALIAS [Mer00] and Section 3.3.3). ALIAS is a library developed at INRIA Sophia Antipolis, by Dr. Jean-Pierre Merlet. ALIAS considers the real roots of zero-dimensional systems of equations and inequalities. ALIAS uses the PROFIL/BIAS (Programmer's Runtime Optimized Fast Interval Library [Knü94]) library for evaluating intervals. It uses different theorems from Real Algebraic Geometry [BCR98] for analysing as well as solving zero-dimensional semi-algebraic systems. The certified computation of the Delaunay graph conflict locator relies on theorems on the uniqueness of a root in given intervals (Kantorovitch, Moore-Krawczyk) and on the certified computation of function intervals by the PROFIL/BIAS library. This computation uses a bisection process on one or all the variables using either only the equations of the system, or using the Jacobian of the system (Moore-Krawczyk test for finding "exactly" the solutions),

or using the Jacobian and the Hessian of the system (with Kantorovitch, Moore-Krawczyk tests). We first used ALIAS on the original system of algebraic equations and inequalities that specifies the Delaunay graph conflict locator for semi-algebraic sets. Then, we have obtained faster computations for conics by replacing the original system by a system simplified by introduction of the implicit equation of the generalised offset to a conic.

Chapter 7 summarises the results obtained in this thesis and presents the avenues of future work.

Chapter 2

Generalisations of the Voronoi diagram for curved objects

In this chapter, we present an overview of the generalisations of the Voronoi diagram for curved objects. These are the generalised Voronoi diagrams which are most closely related to the problem we are addressing in this thesis. These generalisations will be presented in increasing order of relevance for the present thesis. In Section 2.1, we will briefly introduce the Voronoi diagrams for manifolds, studied by Devillers et al. [DMT92] in 1992. In Section 2.2, we will introduce the Voronoi diagram for curved objects, studied by Alt and Schwarzkopf [AS95] in 1995. Finally, in Section 2.3, we will present in more detail the Voronoi diagram for planar domains with curved boundaries [RF99a, RF99b].

2.1 Voronoi diagrams of general manifolds

The Voronoi diagrams for manifolds were analysed in the context of the space of generalised spheres by Devillers et al. [DMT92] in 1992 (see Figure 2.1.1). In $\mathbb{R}^N$, an hypersphere S centred on $\Phi \in \mathbb{R}^N$ and of radius r is mapped to a point $S \in \mathbb{R}^{N+1}$ (at the distance r^2 below the intersection of a vertical line passing through Φ and the paraboloid of equation $\kappa - \Phi_1^2 - \Phi_2^2 = 0$) of the space of generalised spheres (see

the upper part of Figure 2.1.1). The image of a concentric pencil of circles is the vertical line passing through its centre (see Figure 2.1.1).

The sites of the Voronoi diagram for manifolds are manifolds of any dimensions embedded in $\mathbb{R}^N$ [DMT92]. The distance induced by the metric in $\mathbb{R}^N$ is denoted by δ. An N-dimensional topological *manifold* is a Hausdorff (or separated) space i.e., such that every point in it has an open neighbourhood homeomorphic to the open ball in $\mathbb{R}^N$. The distance is the usual distance from a point M to a manifold Z: $\delta(M, Z) = \min_{P \in Z} |MP|$. Therefore, $\delta(M, Z)$ is the radius of the minimum sphere centred at M and "tangent" to Z (such that Z intersects the sphere but not its interior). Let Γ_Z be the set of all the spheres tangent to Z. An example of this set Γ_Z with Z being a circle S is shown on Figure 2.1.2 (the corresponding set is denoted Γ_S). In the space of generalised spheres $S(E)$, if Z is an analytic manifold (i.e. such as all the connecting maps are infinitely often differentiable), then Γ_Z is an analytic manifold (see [DMT92, page 20]): it is the upper envelope of the polar[1] planes of the point-spheres (points) of Z with respect to the earlier mentioned paraboloid. Let $\mathcal{S}$ be a set of sites (manifolds). Let $U_{\mathcal{S}}$ be the upper envelope of the manifolds Γ_Z for all $Z \in \mathcal{S}$.

The intersection of the $U_{\mathcal{S}}$ with a vertical line $\Phi = M$ in the space of spheres gives the sphere with centre M whose radius is the distance to the nearest neighbour of M in $\mathcal{S}$. The projection of the $U_{\mathcal{S}}$ on the n-dimensional Euclidean space is the Voronoi diagram of $\mathcal{S}$. However, by bounding the upper envelope $U_{\mathcal{S}}$, what corresponds to considering only the spheres that are contained in a big sphere enclosing all the sites, we get a compact convex set. The work of Devillers et al. [DMT92] did not include any algorithm for the construction of the Voronoi diagram of general manifolds.

[1]The polar plane of a point M with respect to a quadric Q is the locus of the harmonic conjugates of M with respect to the two intersections of a line through M intersecting Q with Q.

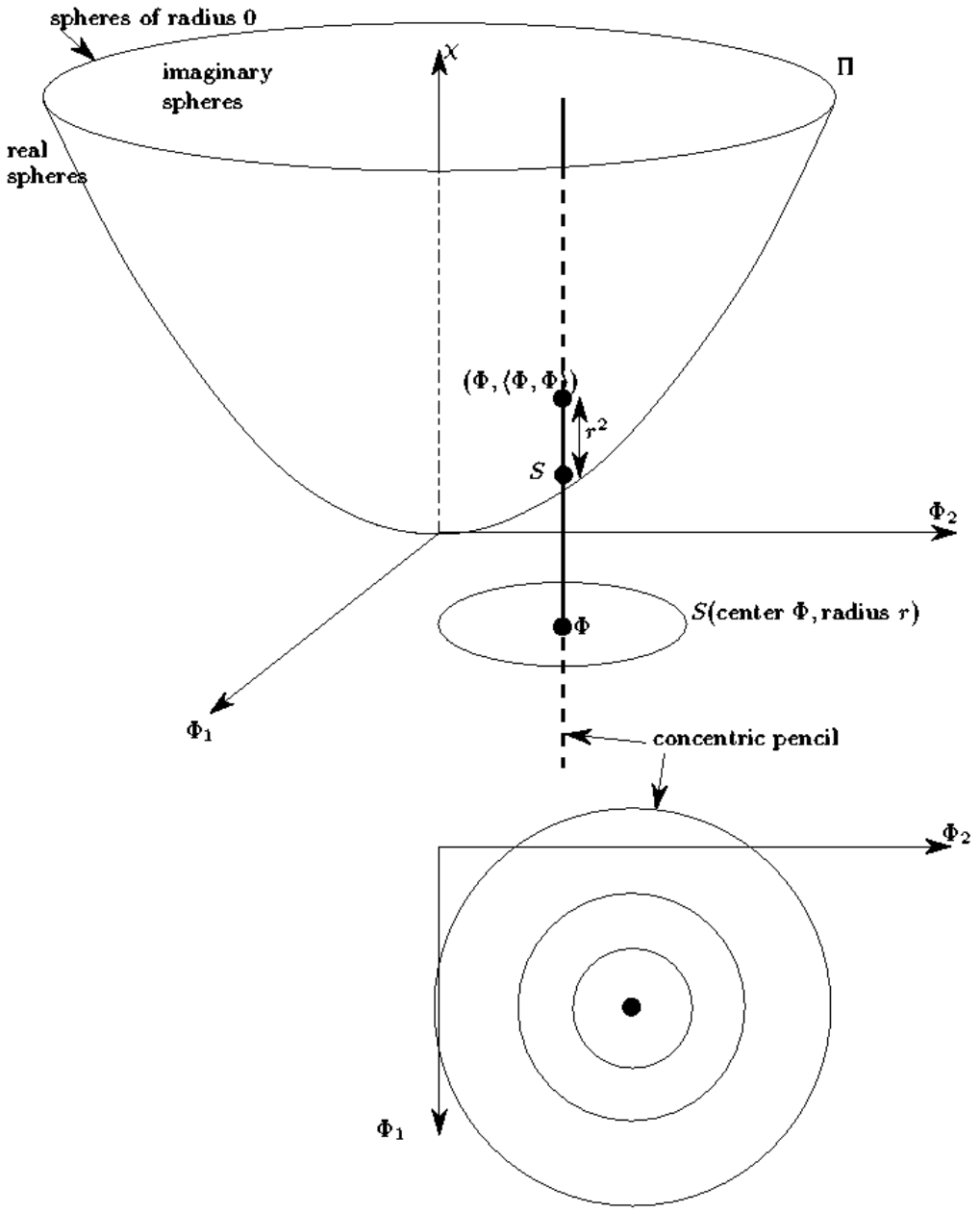

Figure 2.1.1: The space of generalised spheres (taken from [DMT92])

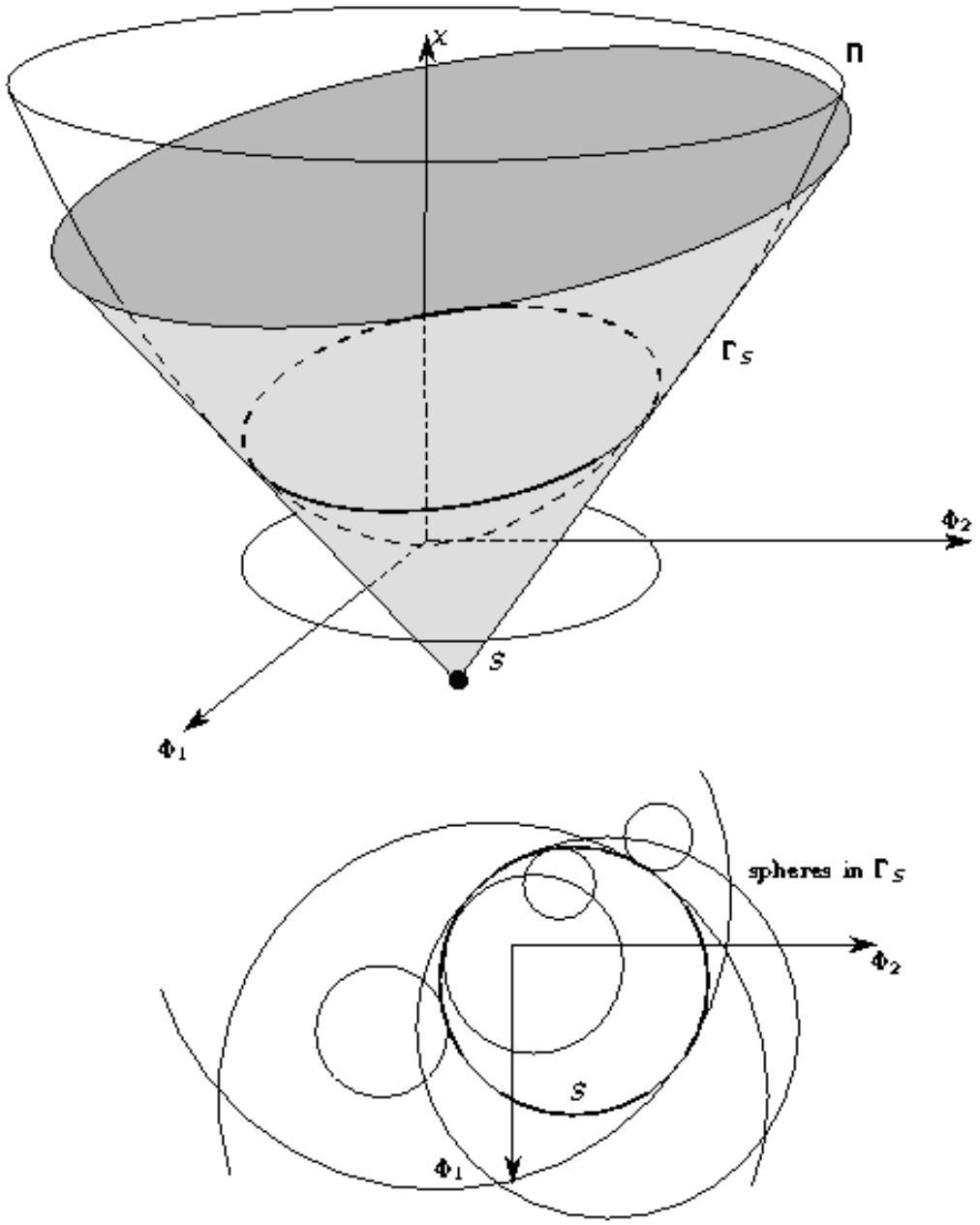

Figure 2.1.2: The circles tangent to a given circle in the space of generalised spheres (taken from [DMT92])

2.2 Voronoi diagrams for curved objects

An incremental randomised algorithm for the construction of the Voronoi diagram for curved objects in the Euclidean plane has been proposed by Alt and Schwarzkopf [AS95], and it is motivated by the applications in motion planning "which lead to the so-called retraction method" [AS95]. The approach of Alt and Schwarzkopf is to decompose the complex curves into open curves that are characterised by the property that no circle touches them in more than one point. Such curves are called "harmless". This decomposition is motivated by the fact that the Voronoi diagram for connected curved objects is not necessarily connected and there may be Voronoi edges between two different parts of the same site (these edges are called self Voronoi edges) and Voronoi vertices between three different parts of the same site (these Voronoi vertices are called self Voronoi vertices). The self Voronoi edges are loci of centres of circles tangent to without containing different parts of the same site, while self Voronoi vertices are centres of circles tangent to without containing different parts of the same site. The curves are broken up into "harmless" pieces to prevent the computation of self Voronoi vertices and self Voronoi edges by the randomised incremental algorithm for the construction of the Voronoi diagram. They prove that this decomposition into harmless pieces assures that the Voronoi diagram is connected, no region is empty, and each region is simply connected.

The first basic assumption of this approach is that curves are abstract objects, and "certain elementary operations are available as black boxes" [AS95]. These operations are the following constructions: "finding the points having the same distance from three given points, finding all points of a given slope, finding points where the curvature has a local maximum, finding the representation of a bisector given the representation of two curves, and finding intersection points of given curves" [AS95]. The curves are encoded as their parametric representation ($\gamma : I \rightarrow \mathbb{R}^2$ where $I \subset \mathbb{R}$ is some closed interval). The second basic assumption is that curves

are supposed to be regular and simple. They define a harmless site as either a point, an open circular arc, or a harmless curve (which is also open by definition); and they define a harmless site collection as a finite set S of pair-wise disjoint harmless sites with the condition that for every circular arc and harmless curve of S, its endpoints are also members of S. This definition of harmless sites excludes closed straight line segments and circular arcs as well as circles (a circular arc should be open, and a circle has to be partitioned into a point and an open circular arc). The points responsible for the non-harmlessness of curves are the local maxima of the absolute value of the curvature (see proof at the beginning of section 3 in [AS95]).

In the incremental randomised algorithm for constructing the Voronoi diagram for curved objects in the plane, the insertion of the sites is done in two steps. First, a point acting as place holder is computed for each one-dimensional harmless site. The point objects and these place holders are inserted first in a random order. Then, the one-dimensional harmless sites are inserted in a random order. The predicates and constructions needed for this incremental randomised algorithm have been considered as "black boxes", and no implementation results have been presented. Algebraic curves are harmful, and the decomposition of an algebraic curve into harmless sites may require a number of cuts bounded by its degree minus one.

2.3 The Voronoi diagram and medial axis transform for planar domains with curved boundaries

We will review the work of Rajesh Ramammurthy and Rida T. Farouki [RF99b, RF99a]; of Rida T. Farouki and Rajesh Ramammurthy [FR98b, FR98a]; and of Rida T. Farouki and John K. Johnstone [FJ94b, FJ94a] on Voronoi diagrams for planar domains with curved boundaries, and point-curve and curve-curve bisectors. This review will also consider a result from [CCM97] stating the sensitivity of the

Voronoi diagram to the order of continuity of the approximations at contact points that is cited in some of the earlier mentioned papers.

The Voronoi diagram and medial axis of a closed bounded planar domain are basic geometric entities associated with that domain. The Voronoi diagram of a domain bounded by N curve segments is a network specifying a partition of the plane into N regions such that each point within a given region is at least as close to its associated curve segment as to all other curve segments. The medial axis is the locus of centres of maximum-radius circles that may be inscribed within the domain. This locus forms also a network. The computation of these geometric entities involves the computation of the nodes and edges of these networks. The edges of these networks are part of point/curve, curve/curve, or self bisectors. The nodes of these networks are the centres of circles touching the boundary at at least two distinct points. The basic assumption (and focus of the work) of Rida T. Farouki et *al.* [RF99b, RF99a, FR98b, FR98a, FJ94b, FJ94a] is that planar domains are considered with piecewise analytic boundaries. For domains with polygonal or piecewise linear/circular boundaries, the bisectors are conics, and efficient algorithms that yield essentially (i.e. topologically) exact Voronoi diagrams have been developed. These bisectors admit rational parameterisations. However, for planar domains bounded by curves having a polynomial or rational parameterisation, the corresponding curve/curve bisectors do not necessarily admit such simple rational parameterisations.

One possible approach for computing the bisectors of those curves having a rational parameterisation is to use approximations by the earlier mentioned piecewise linear/circular curves. However, this yields results "that are not even qualitatively (topologically) correct" [RF99a]. The argument presented in [RF99a] involves two counter examples. In one of them, "the discrepancy between the true Voronoi

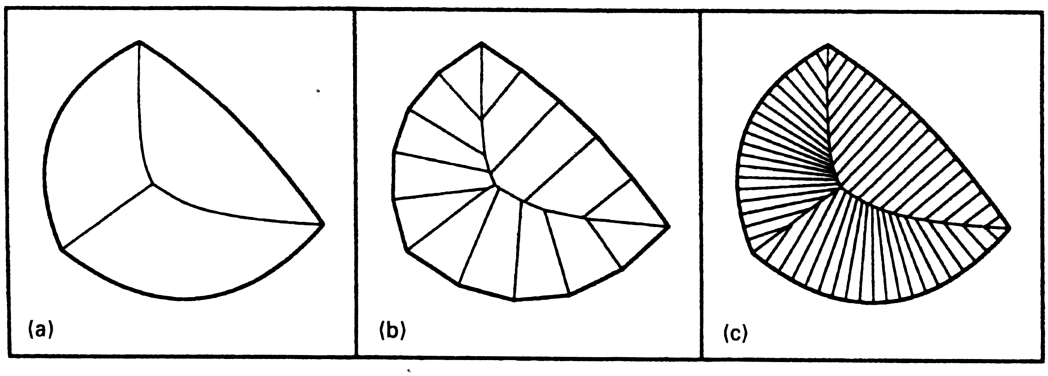

Figure 2.3.1: The difference between (a) the true Voronoi diagram of a planar domain bounded by curves, and the computed from piecewise-linear boundary approximations with (b) 15 segments and (c) 60 segments (taken from [RF99a])

diagram and the approximated one grows as the tolerance on the approximation is tightened by introducing further linear/circular approximating segments" (see Figure 2.3.1). In the other one, "the Voronoi diagram and the medial axis are identical for the exact boundary while they differ for the approximate boundary" (see Figure 2.3.2). The explanation that is given to justify this strange behaviour is that both the Voronoi diagram and the medial axis are very sensitive to the order of continuity of the boundary curve segments at their contact points [CCM97].

The approach used by Ramammurthy and Farouki [RF99b, RF99a] is:

- to determine the rational parameterisations of the Voronoi edges that admit them, and
- to provide piecewise-rational approximations that satisfy a prescribed geometrical tolerance for the remaining Voronoi edges.

The main results that are at the basis of their algorithm are:

- the bisector of a point and a rational curve segment can be described exactly (e.g. in the customary Bézier form [FJ94b]);

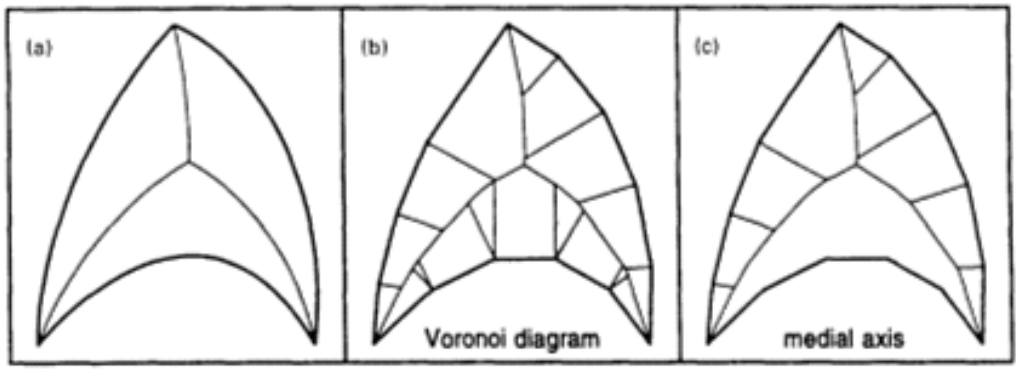

Figure 2.3.2: The internal part of the Voronoi diagram and the medial axis are identical for the exact boundary while they differ for the approximate boundary (taken from [RF99a])

- the bisector of two rational (or even polynomial) curves is not necessarily a rational locus, but by expressing the curve/curve bisector as the envelope of a family of point/curve bisectors (see Figure 2.3.3), the generation of the sequences of singularities (tangent discontinuities) of the curve/curve bisector can be reduced to a family of univariate polynomial root-finding problems,
- the true curve/curve bisector can be approximated to any given geometric tolerance by means of adaptive subdivision and error measures for geometric Hermite interpolants, and its "singularities (tangent discontinuities) can be captured in an essentially exact manner" [FR98b].

This work allows one to only construct the Voronoi diagram approximately since some Voronoi vertex computations are approximate. Moreover, the computation of the bisectors is very heavy since it involves discretising both curves, and then computing the envelopes of the two families of point-curve bisectors. The only algebraic curves that can be processed with the proposed projective resultant based techniques are the algebraic curves admitting rational parameterisations. Finally,

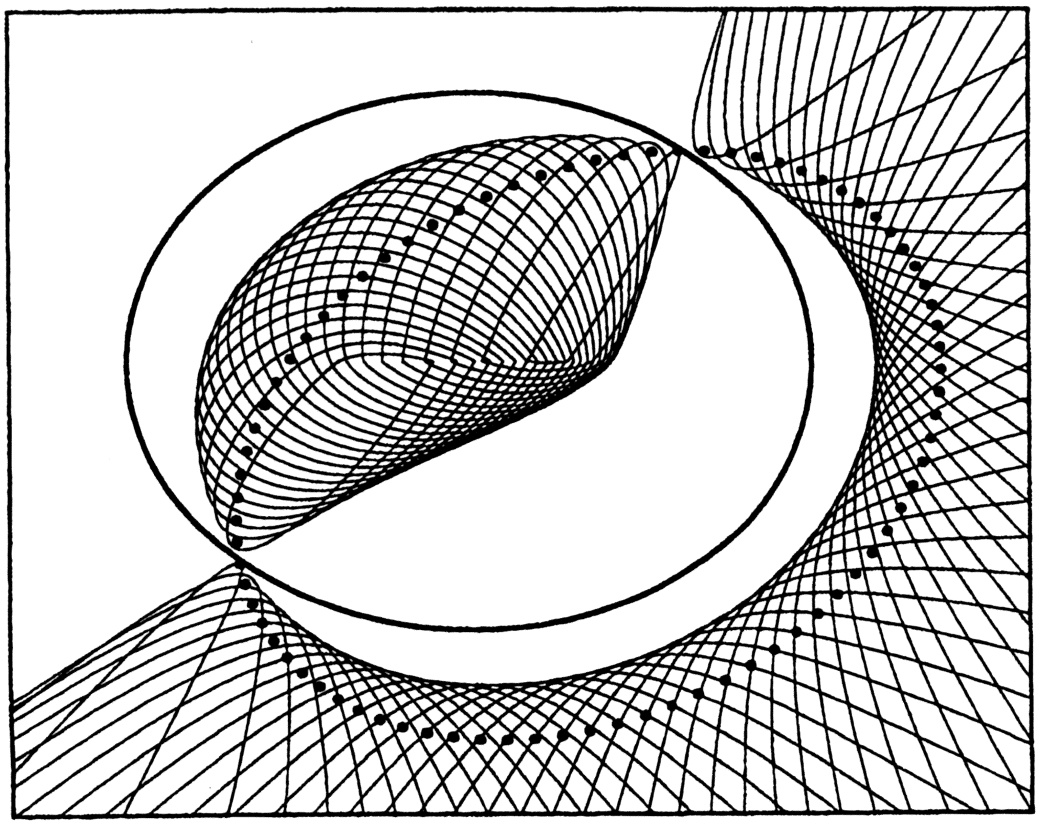

Figure 2.3.3: The bisector of two curves as the envelope of a family of point/curve bisectors (taken from [FR98b])

this work does not directly address the problem of computing exactly the Delaunay graph conflict locator.

We will review the results presented above as well as the algorithm for computing the Voronoi diagram. We have presented this work in further detail hereafter because it is the only implementation of an algorithm for constructing the Voronoi diagram of quite general curved objects.

The algorithm for computing the Voronoi diagram of a planar domain bounded by a rational curve is incremental: one boundary segment is introduced at a time. To perform the Boolean operations involved in the incremental construction of the Voronoi diagram, a complete description of the oriented boundaries of the involved regions is necessary and sufficient. Such a description involves a classification of the Voronoi vertices.

While some kinds of Voronoi vertices involve only rational bisectors, and they can be computed by standard curve intersection algorithms, which are essentially exact; other kinds of Voronoi vertices involve non-rational bisector segments, which must be approximated. Therefore, their location computed as intersection of non-rational bisectors is inherently approximate. Such non-rational bisectors are approximated using a polynomial interpolation method that guarantees the desired order of continuity at contact points (the Hermite interpolant). Moreover, the computed Voronoi vertices are refined through a Newton-Raphson scheme.

The computation and classification of the Voronoi vertices are done by intermediate computations of curve/curve bisectors. These curve/curve bisectors are generally portions of high-order algebraic curves that do not admit rational parameterisations. The approach of Farouki and Ramammurthy is to approach these segments to a prescribed geometrical tolerance. In order to do so, they provide a means of recognising transition points between segments belonging to different types and singularities.

The philosophy of the method to computing curve/curve bisectors is to break the curve/curve bisector into a sequence of segments between singular points, and then to transform the problem of generating ordered sets of points along the bisector to a sequence of univariate polynomial root-finding problems, using point/curve bisectors as an intermediate tool.

Each curve is discretised in turn. The curve/curve bisector is computed as the curves formed by the left and right candidate points (see Figure 2.3.4). The left and right candidate points are the points on the left and right side of the discretised curve respectively that belong to the envelope of the family of point/curve bisectors. The left and right candidate points are the centres of circles passing through the point on the discretised curve and tangent to the other curve or passing through an extremity of that other curve.

The locus of the left and right candidate points is a superset of the curve/curve bisector, called the untrimmed bisector. The untrimmed curve/curve bisectors are trimmed by identifying the values between discrete samples on the untrimmed bisector at which there is a corresponding change in the status for the candidates: retained/discarded, i.e., belonging/not belonging to the trimmed bisector. The Voronoi vertices and the singular points on the curve/curve bisector are also identified.

For each retained point, the unit tangent and curvature of the curve/curve bisector are computed. Hermite interpolants are used to construct the curve/curve bisectors between Voronoi vertices and/or singular points. If the Hermite interpolants do not satisfy the specified tolerance, then additional intermediary points may be added.

Finally, the point/curve bisectors are computed by trimming in a similar way the point/curve bisectors (which admit rational parameterisations).

The problem we address in this thesis is the certified computation of the

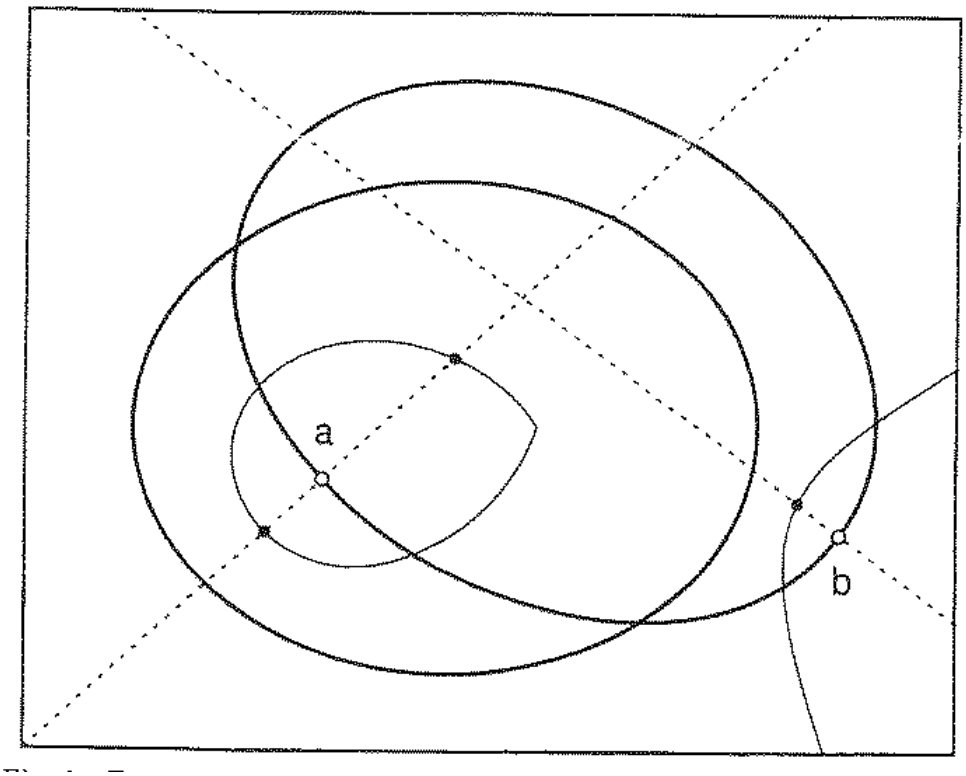

Figure 2.3.4: The left and right candidate points (taken from [FR98b])

Delaunay graph of sets of semi-algebraic sets. The fact that we address the Voronoi diagram of semi-algebraic sets makes of our work an extension of the work on Voronoi diagrams for planar domains with curved boundaries. Indeed, we consider semi-algebraic sets which include algebraic varieties, which in turn strictly include algebraic curves with rational parameterisations. The proper conics (circle, ellipse, parabola and hyperbola) are the simplest example of algebraic curves that do not admit rational parameterisations, and that cannot be processed by the work presented in this section. This results from that the resultant techniques used on the rational parameterisations exclude curves without an algebraic rational parameterisation.

Chapter 3

Combining symbolic computation and scientific computation

In the first section, I will present the Delaunay graph conflict locator for additively weighted points and for circles. This presentation will allow me to introduce in an easier way the reader to the formalisms of the generalised offset (which will be studied in Chapter 4), and of the generalised Voronoi vertex (which will be introduced in Chapter 5).

In the second section, I will present different attempts made to compute the Delaunay graph conflict locator by using a formulation of the conflict locator that was based on the original curves, both using symbolic computational techniques and numerical computational techniques. These were the first attempts at computing the Delaunay graph conflict locator. These experiments suggested that a purely algebraic solution starting from the original curves was not tractable (except for the simple case of the Voronoi diagram of circles).

In the third section, I will briefly introduce the key computational techniques used in this research that constitute a hybrid approach integrating symbolic algeb-

raic precomputations with numerical computational techniques for finding eigenvalues and for solving systems of equations and inequalities. The different aspects of this hybrid symbolic-numerical approach will be introduced in Chapters 5 (for conics) and 6 (for arbitrary semi-algebraic sets). Generally, one performs first algebraic precomputations, and then take over with numerical computations. The main challenge is to identify where the symbolic precomputations should stop, or alternatively, where the numerical analysis techniques should start.

Two central ideas have driven this thesis. The first one is that knowing the structure of the set of solutions may help finding the solutions. For the structure of the solution set, algebraic geometry plays a central role. The second one is that some (one time) symbolic preprocessing may accelerate the certified numerical evaluation of the Delaunay graph conflict locator to be performed several times.

The main computational challenges that have been encountered during this thesis are identifying which computational techniques might work and which computational techniques will probably not work, and finding the optimum between the legitimate wish to compute everything exactly, symbolically and algebraically, and the universal scope of the numerical computational methods for solving systems of equations and inequalities.

We will see that although the Delaunay graph of sets of circles can be computed symbolically and exactly, and the Delaunay graph of sets of algebraic curves can be specified theoretically, the exact computation of the Delaunay graph of sets of algebraic curves is beyond the present limits of exact computational methods such as Gröbner bases [Grö39, Buc92, Buc70, Buc79, Buc88, BCK88, Buc98] and classic projective resultants. Indeed, the complexity of the computation of Gröbner bases is doubly exponential in the number of variables (see the doubly exponential lower bound in [MM84, Huỳ86] and the doubly exponential upper bound in [MM84, Giu84]), and the complexity of the computation of the sparse resultant

is exponential in the number of variables [Emi96]. This exponential complexity of the computation of Gröbner bases or resultants does not affect the complexity of the evaluation of the conflict locator because those computations are algebraic precomputations done only once before implementing the algorithm for evaluating the Delaunay graph conflict locator. So, it makes sense to try to attempt such a huge complexity algebraic precomputations. However, the most limitative resource for those algebraic precomputations is the amount of Random Access Memory and of Virtual Memory accessible. Also, the size of the matrix of the sparse resultant determines the complexity of the numerical computations for finding eigenvalues performed each time the Delaunay graph conflict locator is computed.

3.1 The exact symbolic Delaunay graph conflict locator for circles

We will first present the exact symbolic Delaunay graph conflict locator for additively weighted points when weighted points are introduced one by one, and then introduce what changes for circles. For this purpose, we will present some preliminaries about Additively Weighted Voronoi diagrams.

3.1.1 Preliminaries

Let $\mathbb{N}$ be the set of integers, $\mathbb{R}$ be the set of real numbers, and $\mathbb{R}^2$ be the Euclidean plane. Let $\mathcal{P} = \{P_1, ..., P_N\}$ be the set of generators or sites, where P_i is the *weighted point* located at $p_i \in \mathbb{R}^2$ and of weight $w_i \in \mathbb{R}$. Let C_i be the circle centred at p_i and of radius w_i, which we call *weight circle* hereafter.

The definitions of bisector, influence zone, Voronoi region and Voronoi diagram presented in Chapter 1 generalise to the case where the set of sites $\mathcal{S}$ is a set of weighted points $\mathcal{P}$, and the distance $d(M, P_i)$ (called *additive distance*) between a point M and a site P_i is $d(M, P_i) = \delta(M, p_i) - w_i$, where δ is the Euclidean distance

between points.

The Voronoi region of P_i with respect to the set $\mathcal{P}$ is defined by: $\mathcal{V}(P_i, \mathcal{P}) = \left\{M \in \mathbb{R}^2 | \forall j \neq i : \delta(M, p_i) - w_i < \delta(M, p_j) - w_j\right\}$. The *Additively Weighted Voronoi diagram* of $\mathcal{P}$ is defined by: $V(\mathcal{P}) = \bigcup_{P_i \in \mathcal{P}} \partial V(P_i, \mathcal{P})$. The Additively Weighted Voronoi diagram is illustrated on Figure 3.1.1: the weight circles are drawn as plain disks with a small hole at their centres, the Additively Weighted Voronoi diagram is drawn in plain thick hyperbola segments, and the Delaunay graph is drawn in dashed lines.

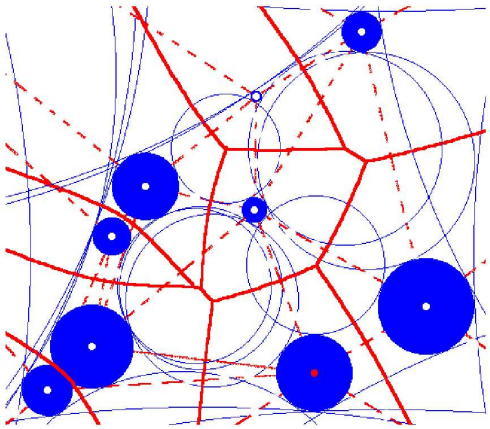

Figure 3.1.1: The Additively Weighted Voronoi diagram

The Additively Weighted Voronoi diagram defines a network composed of edges (loci of points having two nearest neighbours), and vertices (loci of points having three nearest neighbours).

The Additively Weighted Voronoi diagram is related to the Apollonius Tenth problem. The Apollonius Tenth problem is to find a circle Γ tangent to three given circles C_1, C_2, C_3 (see Figure 3.1.2). For additively weighted points, we will see later in this section that only the circles that are either externally tangent to each of three given circles C_1, C_2, C_3 or internally tangent to each of C_1, C_2, C_3, are relevant to the Delaunay graph conflict locator. The centres of the circles that are solutions to the Apollonius Tenth problem are the first example encountered in this

thesis of generalised Voronoi vertices (a concept that we will introduce in Section 5.2). Informally, generalised Voronoi vertices are the centres of circles tangent to $N + 1$ sites, where N is the dimension of the Euclidean space.

Hereafter we will call the solutions of the Apollonius Tenth problem *Apollonius circles*. The centres of the Apollonius circles that are either externally tangent to each of three given circles C_1, C_2, C_3 or internally tangent to each of C_1, C_2, C_3 are the first example encountered in this thesis of true Voronoi vertices (i.e. centres of circles that are touch $N + 1$ sites where N is the dimension of the Euclidean space).

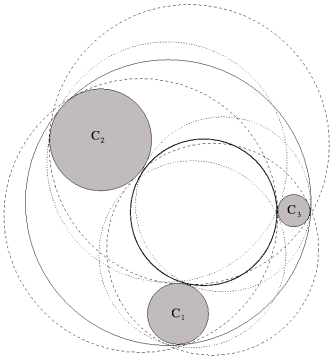

Figure 3.1.2: The Apollonius Tenth problem

3.1.2 The Delaunay graph conflict locator for additively weighted points

In this subsection, we present an exact algebraic conflict locator for the Delaunay graph of additively weighted points (i.e. the dual graph of the Additively Weighted Voronoi diagram). The maximum degree of the polynomials which need to be evaluated to compute this Delaunay conflict locator is 16 (thus, we say that the degree of the conflict locator is 16). This Delaunay graph conflict locator would be the core of a randomised incremental algorithm for constructing the Additively Weighted

Voronoi diagram since the Additively Weighted Voronoi diagram is an abstract Voronoi diagram [Kle89], and thus, it can be constructed with the randomised incremental algorithm of Klein [Kle89]. This work solves the robustness issue in the work of Anton, Mioc and Gold [AMG98] on dynamic Additively Weighted Voronoi diagrams by providing an exact conflict locator. The exact computation of the Additively Weighted Voronoi diagram has not been addressed until Anton et al. [ABMY02]. That paper addressed the exact predicate for the off-line construction of the dual graph of the Additively Weighted Voronoi diagram from the dual of the Power Voronoi diagram of spheres by using the relationship between the Additively Weighted Voronoi diagram in the plane and the Power Voronoi diagram[1] of spheres in the three-dimensional space. In their independent work, Karavelas and Emiris [KE02] provided several exact predicates of maximum degree 16 for achieving the same "in-circle/orientation/edge-conflict-type/difference of radii" test as we do in a single conflict locator. They reduced the degree of their predicate from 28 to 20 and then to 16 using Sturm sequences and invariants.

The motivation for an exact conflict locator lies in the fact that without an exact computation of the Delaunay graph of additively weighted points, some geometric and topologic inconsistencies may appear. This is illustrated with an example. The starting configuration is shown on Figure 3.1.3. There are three weighted points (whose corresponding weight circles are drawn). The Delaunay graph is drawn in dashed lines. The Apollonius circles tangent to the weight circles have been drawn in dotted lines. The real configuration after addition of a fourth weighted point is shown on Figure 3.1.4. The configuration that might have been computed by an approximate algorithm is shown on Figure 3.1.5: the difference between real and perceived situations has been exaggerated to show the difference.

[1]The Power Voronoi diagram is a generalised Voronoi diagram where sites are hyperspheres and the distance between a point and a site is the power of that point with respect to that site [Aur87].

The old Apollonius circles have been adequately perceived to be invalid with respect to the newly inserted weighted point. About the new Voronoi vertices, while on the right of the figure two new Voronoi vertices have been identified as valid with respect to their potential neighbours, on the left of the figure, only one Voronoi vertex has been identified as being valid with respect to its potential neighbours. While the new Voronoi edge between the middle and bottom weighted points can be drawn between the two new Voronoi vertices of the new, middle and bottom weighted points; the Voronoi edge between the top and new weighted points cannot be drawn, because there is no valid Voronoi vertex on the left. There is an inconsistency within the topology: there is one new Voronoi vertex (the Voronoi vertex of the top new and middle weighted points) that cannot be linked by a new Voronoi edge to any other new Voronoi vertex and thus, that Voronoi vertex is incident to only two Voronoi edges. That additively weighted Voronoi diagram that might have been computed by an approximative algorithm is not an additively weighted Voronoi diagram. Thus, even if we perturbate the input weighted points, we will never get that additively weighted Voronoi diagram.

We consider the maintenance of the Delaunay graph of additively weighted points in an incremental way: we check the validity of all the triangles of the Delaunay graph whose vertices are P_1, P_2, P_3 with respect to a newly inserted weighted point P_4 [AKM02]. Thus, the input of the conflict locator is constituted by four points: the first three are supposed to define a triangle in the Delaunay graph, and the last one is the newly inserted weighted point. Let (x_i, y_i) be the coordinates of p_i, for $i = 1, 2, 3, 4$. There are two possible outcomes to the above test of validity: either the triangles are valid with respect to the newly inserted weighted point and the triangles remain in the Delaunay graph, or one or two triangles are not valid with respect to the newly inserted weighted point and those triangles will not be present in the Delaunay graph any longer. We can see an example of the later case in Figure 3.1.6. A triangle having $P_1P_2P_3$ as vertices is not valid with respect

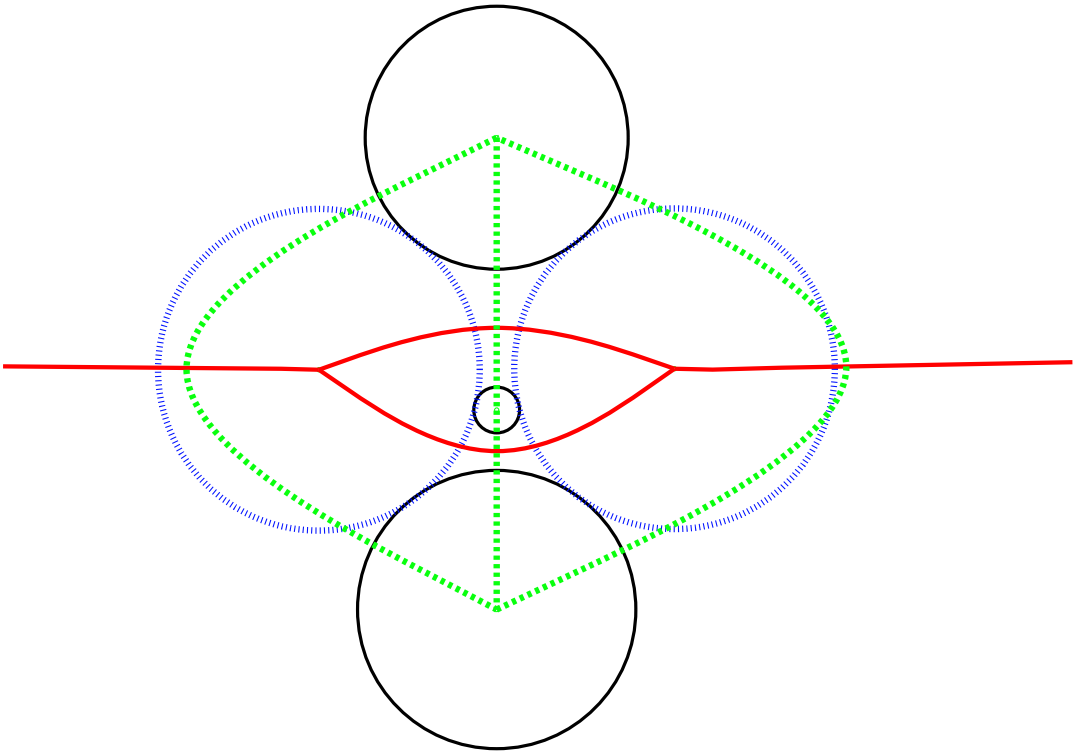

Figure 3.1.3: The starting configuration

to the weighted point P_4. Thus, it will not belong any longer to the Delaunay graph after the insertion of P_4.

The conflict locator consists of determining which ones of the additively weighted Voronoi vertices of P_1, P_2 and P_3 will not remain after the insertion of P_4. This is equivalent in turn to the additive distance from which ones of the additively weighted Voronoi vertices of P_1, P_2 and P_3 to P_4 is smaller than the additive distance of that Voronoi vertex to P_1 (or P_2 or P_3, see Figure 3.1.6).

Any *additively weighted Voronoi vertex* I of P_1, P_2, and P_3 with coordinates (x, y) can be obtained algebraically by computing the common intersection of the three circles C_1', C_2' and C_3' expanding (see Figure 3.1.7), or shrinking (see Figure 3.1.8) from the three first circles C_1, C_2 and C_3 all at the same rate. The common

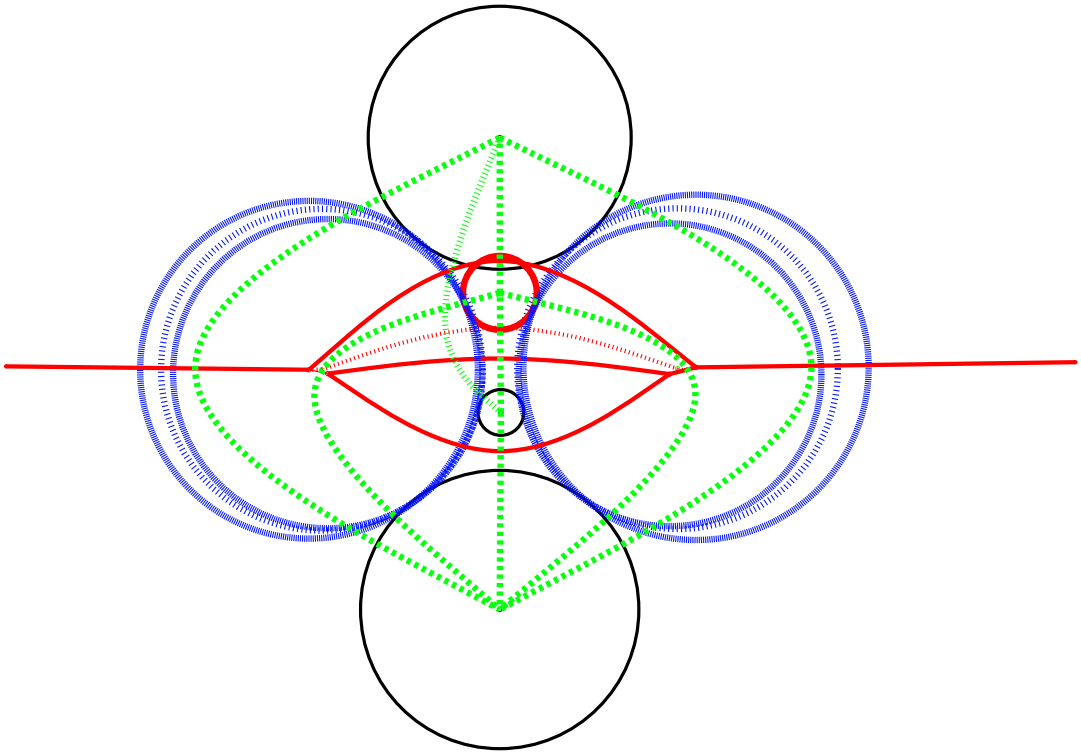

Figure 3.1.4: The real configuration after addition of the fourth weighted point (bold weight circle)

signed expansion of the first three circles is denoted by r. Each circle C'' centred on (x, y) and of radius r is either externally tangent to the first three circles (if the expansion r is positive) or internally tangent to the first three circles (if the expansion r is negative).

The centres coordinates x, y and radii r of the circles C'' centred on the intersections $I = C_1' \cap C_2' \cap C_3'$ and either externally or internally tangent to each of C_1, C_2, and C_3 can be computed algebraically as the solutions of the following system of three quadratic equations in the variables x, y and r:

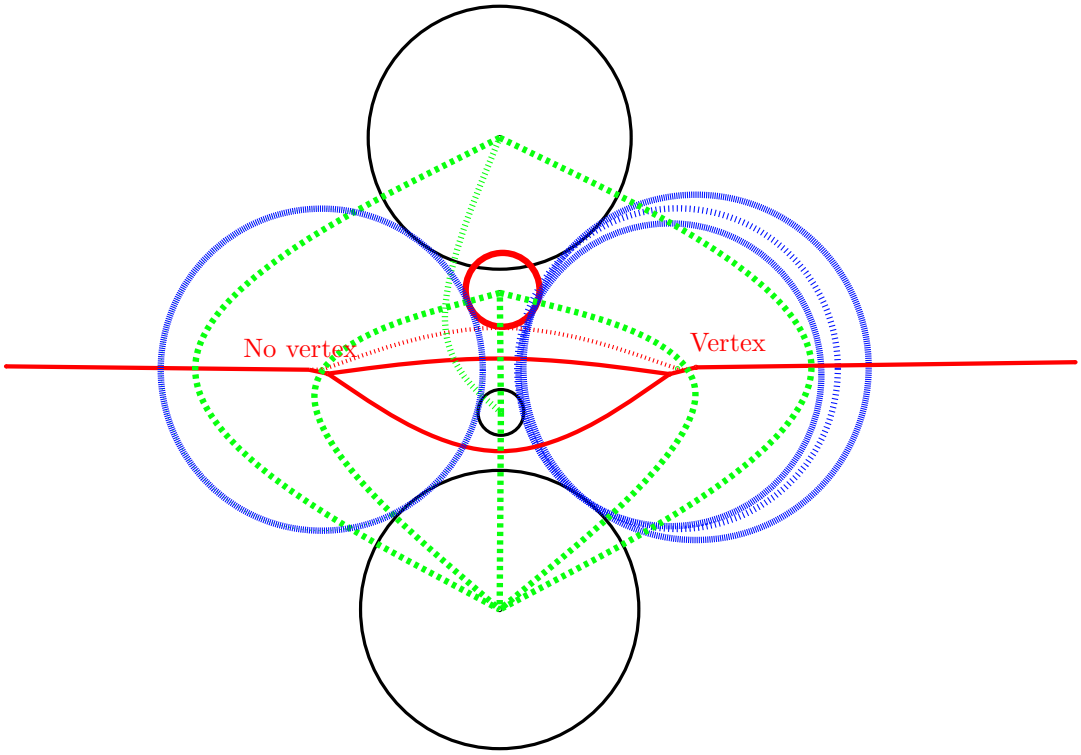

Figure 3.1.5: The configuration computed by an approximate algorithm

$$\begin{cases} c_1'(x,y,r) = (x-x_1)^2 + (y-y_1)^2 - (w_1+r)^2 = 0 \\ c_2'(x,y,r) = (x-x_2)^2 + (y-y_2)^2 - (w_2+r)^2 = 0 \\ c_3'(x,y,r) = (x-x_3)^2 + (y-y_3)^2 - (w_3+r)^2 = 0 \end{cases}$$

Subtracting one of the equations (say $c_1'(x,y,r) = 0$) from the remaining two ($c_2'(x,y,r) = 0$ and $c_3'(x,y,r) = 0$) results in a system of 2 linear equations, from which x and y may be expressed as linear functions of r. Substitution in the first equation $c_1'(x,y,r) = 0$ then leads to a quadratic equation in r. This means that the unknown quantities x, y, r can be expressed with quadratic radicals as functions of the given centres and radii.

Though the simplest thing to do now would be to compute the two Voronoi vertices and use their computed coordinates and corresponding signed expansion in

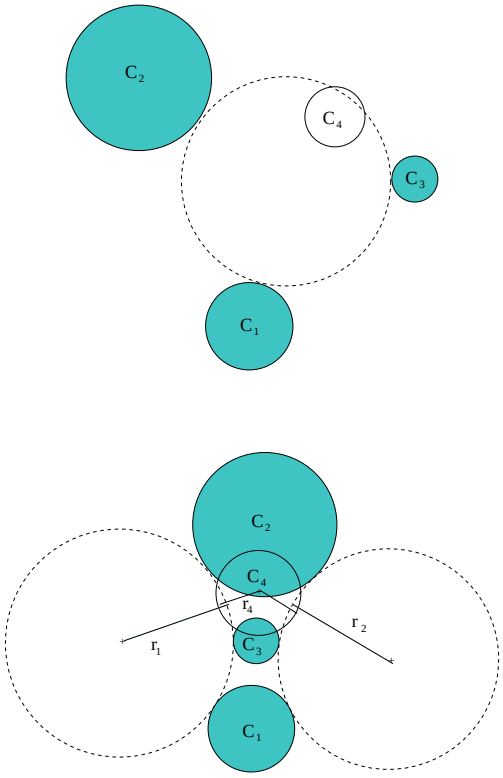

Figure 3.1.6: The Delaunay graph conflict locator for the Additively Weighted Voronoi diagram

the computation of the values certifying the output of the Delaunay graph conflict locator, it is not desirable because this method would not be generalisable to conics or higher degree algebraic curves. We will detail hereafter only the computation of the values certifying the presence in the output list. To get the exact Delaunay graph conflict locator in a more elegant and generalisable way, we evaluated the values certifying the conflict locator output without relying on the computation of the Voronoi vertices as an intermediary computation. This is done by evaluating the values taken by the polynomial function expressing the relative position of C_4 with respect to C'' on the set of solutions of the system (i.e. the common zeroes of

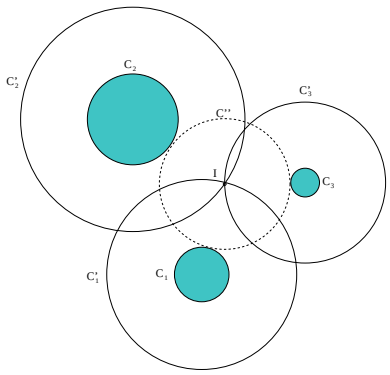

Figure 3.1.7: The Additively Weighted Voronoi vertex as the common intersection of three expanding circles

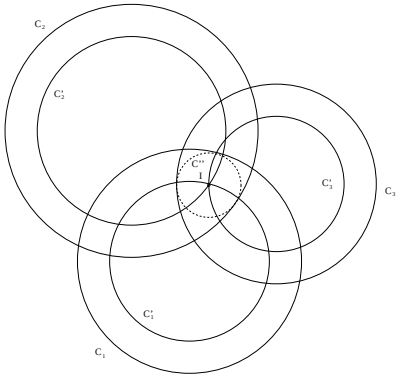

Figure 3.1.8: The Additively Weighted Voronoi vertex as the common intersection of three shrinking circles

the three polynomials c'_1, c'_2 and c'_3). This is possible thanks to the translation that exists between geometry and algebra. More specifically, to the geometric set X of the set of common zeroes of the three polynomials c'_1, c'_2 and c'_3 in K^3, where K is an algebraically closed field [Lan02, Definition before Theorem 1, Section 2, Chapter VII], we can associate the set of all polynomials vanishing on the points of X, i.e., the set of polynomials $f_1c'_1 + f_2c'_2 + f_3c'_3$ where the $f_i, i = 1, 2, 3$ are polynomials in the three variables x, y, r with coefficients in K. This set is the *ideal* [GP02, Definition 1.3.1] $\langle c'_1, c'_2, c'_3 \rangle$. The set of polynomials with coefficients in K forms with

the addition and the multiplication of polynomials a ring: the *ring of polynomials* [GP02, Definition 1.1.3]. It is easy to see that a polynomial function $g\,(x, y, r)$ on K^3 is mapped to a polynomial function on X if we recursively subtract from g any polynomial in g belonging to $\langle c'_1, c'_2, c'_3\rangle$ until no monomial in g can be divided by each one of the lexicographically highest monomials in c'_1, c'_2 and c'_3. The result of this mapping gives a canonic representative of the reminder of the Euclidean division of the polynomial g by the polynomials c'_1, c'_2 and c'_3. The image of the ring of polynomials by this mapping is called the *quotient algebra* [Lan02, Section 3, Chapter II] of the ring of polynomials by the ideal $\langle c'_1, c'_2, c'_3\rangle$. It is also easy to see that $\langle c'_1, c'_2 - c'_1, c'_3 - c'_1\rangle$=$\langle c'_1, c'_2, c'_3\rangle$. Moreover, if we recursively subtract from g any polynomial in g belonging to $\langle c'_1, c'_2 - c'_1, c'_3 - c'_1\rangle$ till the only monomials in g are 1 and r, we get the same result as the preceding mapping. The polynomials $c'_1, c'_2 - c'_1, c'_3 - c'_1$ constitute what is called a *Gröbner basis* [GP02, Definition 1.6.1] of the ideal $\langle c'_1, c'_2, c'_3\rangle$. The monomials 1 and r are standard monomials. Gröbner bases are used in Computational Algebraic Geometry in order to compute a canonic representative of the remainder of the division of one polynomial by several polynomials generating a given ideal I. This canonic representative belongs to the quotient algebra of the ring of polynomials by the ideal I. The Gröbner basis for this system provides a set of polynomials that define uniquely the algebraic relationships between variables for the solutions of the system. The initial (largest with respect to some monomial order [CLO98]) monomials of each one of the polynomials of the Gröbner basis form an ideal. The monomials that do not pertain to this ideal form a basis for the representatives of the equivalence class of the remainders of the division of a polynomial by the polynomials of the system in the quotient algebra. These monomials are called standard monomials. The size of this basis equals the *dimension* [GP02, see definition on page 414] of the quotient algebra and the number of solutions of the system counted with their multiplicity [Lan02]. In the case of the conflict locator for the additively weighted Voronoi diagram, there

are two solutions.

The polynomial $g = (x_4 - x)^2 + (y_4 - y)^2 - (r + r_4)^2$ expresses the relative position of C_4 with respect to C''. Indeed C'' is tangent to C_4 if, and only if, the Euclidean distance between the centres of C'' and of C_4 (i.e., (x, y) and p_4) equals the sum of the radii r and r_4, i.e. $(x_4 - x)^2 + (y_4 - y)^2 - (r + r_4)^2 = 0$. The open balls bounded by C'' and C_4 intersect if, and only if, the Euclidean distance between the centres of C'' and of C_4 is smaller than the sum of the radii r and r_4, i.e. $(x_4 - x)^2 + (y_4 - y)^2 - (r + r_4)^2 < 0$. The circles C'' and C_4 are disjoint if, and only if, the Euclidean distance between the centres of C'' and of C_4 is greater than the sum of the radii r and r_4, i.e. $(x_4 - x)^2 + (y_4 - y)^2 - (r + r_4)^2 > 0$. We considered the operation of multiplication of polynomials by the polynomial g. This *multiplication operator* is a linear mapping. The operation of this mapping on the canonic representative of the reminder of the division of a polynomial by c'_1, c'_2 and c'_3 is also a linear mapping that can be expressed by a matrix since the quotient algebra has a finite dimension.

First, we compute the matrix $M_g = \begin{pmatrix} m_{00} & m_{01} \\ m_{10} & m_{11} \end{pmatrix}$ of the following multiplication operator on the quotient algebra:
$m_g : [f] \longrightarrow [gf]$.

The eigenvalues of M_g are the values of g taken on X (see Theorem 4.5, page 54 in [CLO98]). The eigenvalues of M_g are the solutions of $det\,(M_g - \lambda I) = 0$, where I denotes the 2×2 identity matrix, i.e. the roots of

$$\lambda^2 - \lambda\,(m_{00} + m_{11}) + (m_{00}m_{11}) - (m_{01}m_{10}) = 0 \tag{3.1.1}$$

The values certifying the presence in the list output by the Delaunay graph conflict locator are the signs of the values taken by g, and they are determined by the sign of the roots of Equation 3.1.1 (which are the eigenvalues of M_g). If there is only one eigenvalue and it is 0 then the fourth circle is tangent to the circle externally tangent to the first three circles. The sign of Δ (where $\Delta =$

$(m_{00} + m_{11})^2 - 4\,(m_{00}m_{01} - m_{01}m_{10})\ )$ cannot be negative because this would be equivalent to the fact there would be no triangle with vertices C_1, C_2 and C_3 in the old Delaunay graph (because of the absence of real Voronoi vertex, see Figure 3.1.9). Thus, $sign\,(\Delta)$ is 0 or positive, and we have to evaluate the sign of the roots of the quadratic equation.

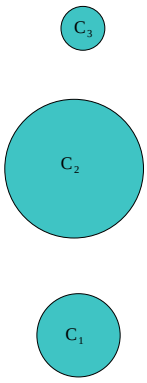

Figure 3.1.9: There is no such triangle in the old Delaunay graph because of the absence of a real Voronoi vertex

When there is only one double root of Equation 3.1.1 then we have the following two possibilities. Either the value of the root of Equation 3.1.1 is positive or 0 and the triangle will remain in the new Delaunay graph, or the value of the root of Equation 3.1.1 is negative and the triangle will disappear in the new Delaunay graph (see Figure 3.1.6). When there are two real roots of Equation 3.1.1, we have two triangles to consider (see Figure 3.1.10). The triangles that correspond to the roots with a negative value will disappear in the new Delaunay graph (see Figure 3.1.10).

There is not much interest in showing the elements of the matrix of the multiplication operator here, but the Macaulay 2 [GS] code is presented in Appendix A. The exact algebraic computation of the Delaunay graph conflict locator we have presented in the previous paragraph is not generalisable to the other proper conics or

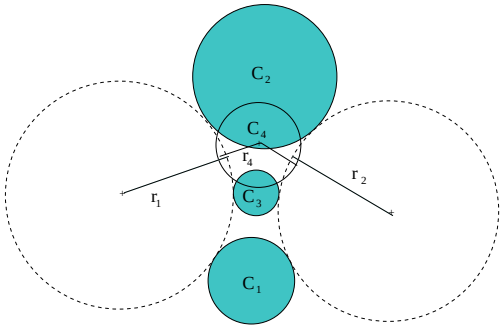

Figure 3.1.10: Two triangles can possibly disappear simultaneously by the addition of a single weighted point

higher degree algebraic curves. Indeed, the size of the multiplication operator matrix is greater than 4 for the other proper conics and for higher degree algebraic curves (see Section 5.3), and an algebraic equation of degree 5 or more is not necessarily solvable by radicals (see [BB96, Theorem 8.4.8]). Even if we can obtain the matrix of the multiplication operator symbolically, we will need numerical methods for computing the eigenvalues of that matrix, which give the answer to the Delaunay graph conflict locator. We will now present the Delaunay graph conflict locator for circles, emphasising on the changes with respect to the Delaunay graph of additively weighted points presented in this subsection.

3.1.3 The Delaunay graph conflict locator for circles

Let $\mathcal{C} = \{C_1, ..., C_N\}$ be the set of generators or sites, with all the C_i being circles in $\mathbb{R}^2$. Let p_i be the centre of C_i and r_i be the radius of C_i.

The definitions of bisector, influence zone, Voronoi region and Voronoi diagram presented in Chapter 1 generalise to the case where the set of sites $\mathcal{S}$ is a set of circles $\mathcal{C}$, and the distance $d(M, C_i)$ between a point M and a site C_i is the Euclidean distance between M and the closest point on C_i from M, i.e.

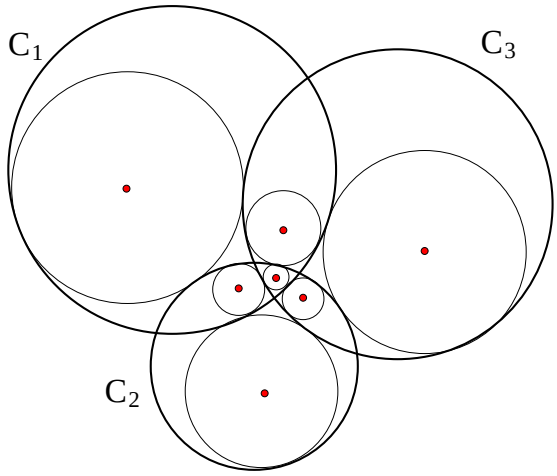

Figure 3.1.11: Seven Apollonius circles centres that are true Voronoi vertices

$d(M, C_i) = |\delta(M, p_i) - r_i|$, where δ is the Euclidean distance between points. Observe that assuming C_i is centred on p_i and $r_i = w_i$ for $i = 1, .., N$, this distance is the absolute value of the additive distance used in the previous subsection. The Voronoi region of C_i with respect to the set $\mathcal{C}$ is thus defined by:
$\mathcal{V}(C_i, \mathcal{C}) = \left\{M \in \mathbb{R}^2 | \forall j \neq i : |\delta(M, p_i) - r_i| < |\delta(M, p_j) - r_j|\right\}$. The Voronoi diagram of $\mathcal{C}$ is defined by: $V(\mathcal{C}) = \bigcup_{C_i \in \mathcal{C}} \partial V(C_i, \mathcal{C})$.

In the previous subsection, we observed that two Apollonius circles centres are true Voronoi vertices of the Additively Weighted Voronoi diagram (the circles that are either externally or internally tangent to three given circles). When the sites are circles, up to seven of the eight Apollonius circles may be relevant to the Delaunay graph conflict locator (see Figure 3.1.11).

We consider the maintenance of the Delaunay graph of circles in an incremental way: we check the validity of all the triangles of the Delaunay graph whose vertices are a given triple of circles with respect to a given newly inserted circle. Thus, four circles C_1, C_2, C_3 and C_4 are given: the first three are supposed to define one or more triangles in the Delaunay graph, and the last one is the newly inserted

circle. Let (x_i, y_i) be the coordinates of p_i for $i = 1, 2, 3, 4$. There are two possible outcomes to the above test of validity. Either the triangles are valid with respect to the newly inserted weighted point and the triangles remain in the new Delaunay graph, or there is at least one triangle that is not valid with respect to the newly inserted weighted point and these triangles will not be present in the Delaunay graph any longer.

The Apollonius circles of C_1, C_2 and C_3 can be obtained algebraically by computing the common intersection of the three circles C_1', C_2' and C_3' (see Figure 3.1.7) expanding or shrinking from the three first circles C_1, C_2 and C_3 all with the same absolute value of the rate. The common unsigned expansion of the first three circles is denoted by r. The coordinates of the intersection I of C_1', C_2' and C_3' are denoted (x, y). The circle C'' centred on (x, y) and of radius r is tangent to the first three circles.

Thus, the Apollonius circles are the solutions of one of the eight following systems (I) of three quadratic equations in three unknowns x, y, r:

$$\begin{cases} (x - x_1)^2 + (y - y_1)^2 - (r_1 \pm r)^2 = 0 \\ (x - x_2)^2 + (y - y_2)^2 - (r_2 \pm r)^2 = 0 \\ (x - x_3)^2 + (y - y_3)^2 - (r_3 \pm r)^2 = 0 \end{cases} .$$

By replacing r by $-r$ in one of the preceding systems of equations, we still get another one of the preceding systems of equations. Thus, let us suppose r is the signed expansion of C_1. Then, we can reformulate the preceding systems of equations as the following systems (II) of equations:

$$\begin{cases} (x - x_1)^2 + (y - y_1)^2 - (r_1 + r)^2 = 0 \\ (x - x_2)^2 + (y - y_2)^2 - (r_2 \pm r)^2 = 0 \\ (x - x_3)^2 + (y - y_3)^2 - (r_3 \pm r)^2 = 0 \end{cases}$$

Now let us consider for each system (II) the set X of solutions of the system (II) of equations in K^3, where K is an algebraically closed field.

Subtracting one of the equations from the remaining two results in a system

of 2 linear equations, from which x and y may be expressed as linear functions of r. Substitution in the first equation then leads to a quadratic equation in r. This means that the unknown quantities x, y, r can be expressed with quadratic radicals as functions of the given centres and radii for each one of the systems of equations above.

As before, though the simplest thing to do now would be to compute the two Voronoi vertices and use their computed coordinates and corresponding signed expansion in the computation of the values certifying the output of the Delaunay graph conflict locator, it is not desirable because this method would not be generalisable to conics or higher degree curves.

For the Delaunay graph of additively weighted points, the true Voronoi vertices are the solutions of one system of algebraic equations. Unlike the previous case, for the Delaunay graph of circles, the true Voronoi vertices are not all the solutions of one system of algebraic equations, but a subset of the solutions of four systems of algebraic equations. The solutions of the algebraic equations are the Apollonius circles, whose centres are generalised Voronoi vertices (a concept that we will introduce in Section 5.2). We thus need to determine which Apollonius circles centres are potentially true Voronoi vertices (only the real Apollonius circles centres can be true Voronoi vertices).

There are four possible determinations of the true Voronoi vertices from Apollonius circles centres of C_1, C_2 and C_3:

first case if C_1, C_2 and C_3 mutually intersect, then the real circles among the seven Apollonius circles that are not internally tangent to each of C_1, C_2 and C_3 correspond to true Voronoi vertices (their centres are true Voronoi vertices, see Figure 3.1.11), and reciprocally.

second case if one circle (say C_1) intersects the two others (C_2 and C_3) which do not intersect, then only the real Apollonius circles that are either externally tangent to each of C_1, C_2 and C_3, or internally tangent to C_1 and externally

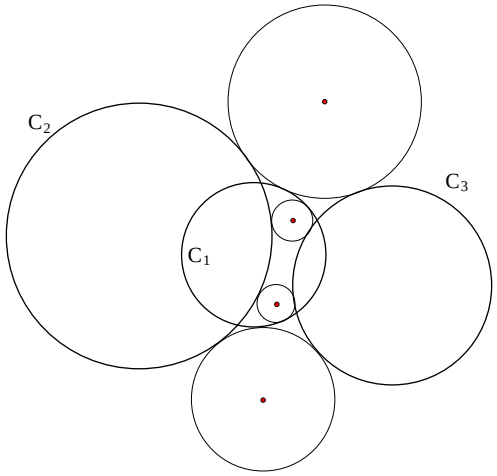

Figure 3.1.12: Four Apollonius circles centres that are true Voronoi vertices

tangent to C_2 and C_3 correspond to true Voronoi vertices (their centres are true Voronoi vertices, see Figure 3.1.12).

third case if two circles (say C_1 and C_2) intersect the interior of the third one (C_3) and at least one of them (say C_1) is contained in the interior of C_3, then only the real Apollonius circles that are externally tangent to C_1 and C_2 and internally tangent to C_3 correspond to true Voronoi vertices (their centres are true Voronoi vertices, see Figure 3.1.13).

fourth case otherwise (if none of the three situations above apply), only the real Apollonius circles that are externally tangent to C_1, C_2 and C_3 correspond to true Voronoi vertices (their centres are true Voronoi vertices, see Figure 3.1.14).

The case where one circle (say C_1) lies in the interior of a second circle (say C_2), which lies in the interior of the third circle (C_3), or only one circle (say C_1) lies within the interior of one of the other ones (say C_2) cannot happen because then, there would be no Voronoi vertices and the triangle $C_1C_2C_3$ would not exist in the

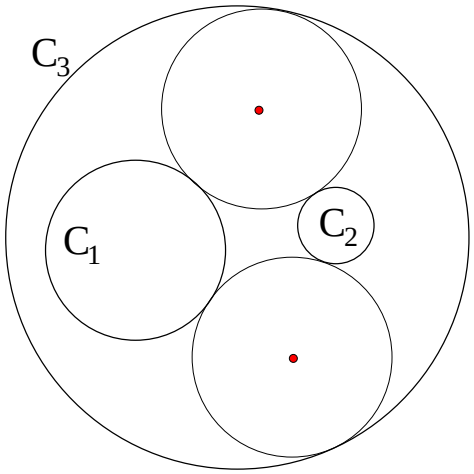

Figure 3.1.13: Two Apollonius circles centres that are true Voronoi vertices (first case)

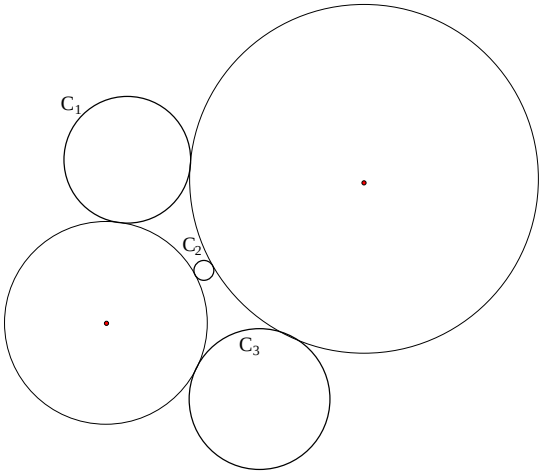

Figure 3.1.14: Two Apollonius circles centres are true Voronoi vertices (second case)

Delaunay graph.

Now that we have seen the different cases of true Voronoi vertices, we will see how we can test in which case we are and which solutions of the systems of equations (II) described above correspond to true Voronoi vertices.

first case C_1, C_2 and C_3 mutually intersect if, and only if, $d(p_1, p_2) - r_1 - r_2 \leq 0$ and $d(p_1, p_3) - r_1 - r_3 \leq 0$ and $d(p_2, p_3) - r_2 - r_3 \leq 0$. The computation of this test can be done exactly, since the only variables that are not input to the Delaunay graph conflict locator are the distances, and these distances are expressed by radicals. Indeed, we need to test the sign of the difference of a radical and a number which do not depend on intermediary computations. The true Voronoi vertices are the real solutions of all the systems of equations (II) such that $r > 0$.

second case C_1 intersects C_2 and C_3, and C_2 and C_3 have no point of intersection if, and only if, $d(p_1, p_2) - r_1 - r_2 \leq 0$ and $d(p_1, p_3) - r_1 - r_3 \leq 0$ and $d(p_2, p_3) - r_2 - r_3 > 0$. The computation of this test can be done exactly for the same reasons as the previous case. The true Voronoi vertices are the real solutions of the system of equations:

$$\begin{cases} (x - x_1)^2 + (y - y_1)^2 - (r_1 \pm r)^2 = 0 \\ (x - x_2)^2 + (y - y_2)^2 - (r_2 - r)^2 = 0 \\ (x - x_3)^2 + (y - y_3)^2 - (r_3 - r)^2 = 0 \end{cases}$$

with $r < 0$.

third case C_1 lies in the interior of C_3 and C_2 intersects the interior of C_3 if, and only if, $d(p_1, p_3) + r_1 - r_3 < 0$ and $d(p_2, p_3) - r_2 - r_3 < 0$ and $(x_1 - x_3)^2 + (y_1 - y_3)^2 - r_3^2 < 0$. The computation of this test can be done exactly for the same reasons as the previous case. The true Voronoi vertices are the real solutions of the system of equations:

$$\begin{cases} (x-x_1)^2+(y-y_1)^2-(r_1+r)^2=0 \\ (x-x_2)^2+(y-y_2)^2-(r_2+r)^2=0 \\ (x-x_3)^2+(y-y_3)^2-(r_3-r)^2=0 \end{cases}$$

such that $r > 0$.

fourth case this is the case if all the previous three tests failed. The true Voronoi vertices are the real solutions of the system of equations:

$$\begin{cases} (x-x_1)^2+(y-y_1)^2-(r_1+r)^2=0 \\ (x-x_2)^2+(y-y_2)^2-(r_2+r)^2=0 \\ (x-x_3)^2+(y-y_3)^2-(r_3+r)^2=0 \end{cases}$$

with $r > 0$.

As before, we used the same algebraic machinery to compute the values of polynomials that are taken by the true Voronoi vertices without solving any intermediate system of equations. We computed the Gröbner basis of the ideal of X for each one of the systems (II) encountered. Each one of these Gröbner basis consists of the earlier mentioned quadratic equation in r and linear equations in x, y and r.

For the Delaunay graph of additively weighted points, we observed that evaluating the signs of a single polynomial ($g = (x_4 - x)^2 + (y_4 - y)^2 - (r + r_4)^2$) taken on the real points of X was enough to provide the values certifying the presence in the list output by the conflict locator. As before, we can check for the existence of real solutions by evaluating the sign of the discriminant of the characteristic polynomial. We will suppose the real solutions to the systems (II) have been tested. Unlike in the previous case, here we need to evaluate the signs taken by both g and r on each one of the points of X. Indeed, we need not only to check the relative position of C_4 with respect to the Apollonius circles, but we need for each Apollonius circle, to check the relative position of C_4 with respect to that Apollonius circle, and to check whether that Apollonius circle corresponds to a true Voronoi vertex.

As before, we considered the operation of multiplication of polynomials by

the polynomial g, whose sign expresses the relative position of C_4 with respect to C''. We also considered the operation of multiplication of polynomials by the polynomial r, whose sign allows one to check whether the solutions correspond to true Voronoi vertices. These operations are linear mappings. The operations of these mappings on the canonic representative of the remainder of the Euclidean division of a polynomial by the three polynomials of the system are also linear mappings that can be expressed by a matrix.

We need to be able to associate the signs of the values of g with the signs of the values of r taken on the (real) solutions of each system (II). For a given system (II), let M_g and M_r be the matrices of the result of the multiplication by g and by r respectively on the canonic representative of the remainder of the division of a polynomial by the three polynomials of the system. Since these multiplication maps commute, it is possible to use the transformation matrix obtained during the computation of the Jordan form of one of these matrices to triangularise the other matrix by a simple multiplication of matrices [CLO98]. Indeed, the computation of the Jordan form for M_g gives the triangular matrix $P^{-1}M_gP$ of the Schur form of that matrix where P is a unitary matrix called the transformation matrix; and $P^{-1}M_rP$ is triangular. Finally, we can obtain the solutions by reading the diagonal entries in turn in each one of the Jordan forms of these matrices (the diagonal entries of the Jordan form of a matrix are its eigenvalues). The row number on each one of these matrices corresponds to the index of the solution. By evaluating the signs of the diagonal entries in the Jordan forms of M_g and of M_r on the same line, we associate the signs of the values of g with the signs of the values of r taken on the solutions of each system (II).

We will see that though this method for computing several polynomials in the quotient algebra can be generalised to algebras of higher dimension, it will not be possible in practice to use this method to compute the Delaunay graph for conics.

3.2 Formulating the Delaunay graph conflict locator for curves from those curves

In this section, we describe different experiences with different Computer Algebra Systems and the interval analysis based solver ALIAS in trying to compute the Delaunay graph conflict locator for algebraic curves.

To compute that conflict locator, we need first to write the algebraic equations (and inequalities) that should be satisfied by all the geometric loci that will intervene in the evaluation of the conflict locator. The input of the Delaunay graph conflict locator is a quadruple (C_1, C_2, C_3, C_4) of semi-algebraic one-dimensional sets in the plane. These geometric loci that are considered in the conflict locator are one point (x_i, y_i) on each one of the four curves $C_i, i = 1, .., 4$, and the circle (whose centre will be denoted (x, y) and radius r) touching C_1, C_2 and C_3 (see Figure 3.2.1). The distance between (x, y) and (x_4, y_4) will be denoted R. The formulation of the Delaunay graph conflict locator will be described in more detail in Sections 4.1 (generalised offset), 5.3 (conflict locator for conics) and 6.1 (conflict locator for semi-algebraic sets). For each one of the four curves $C_i, i = 1, .., 4$, there are three equations: the implicit equation of the curve C_i: $c_i(x_i, y_i)$, the equation of the normal to the curve C_i at the point (x_i, y_i): $n_i(x_i, y_i, x, y)$, and the equation expressing the distance between the point (x_i, y_i) and the point (x, y): $d_i(x_i, y_i, x, y, r)$ for $i = 1, 2, 3$ and $d_4(x_4, y_4, x, y, R)$. The equality of the distances between (x, y) and (x_1, y_1), between (x, y) and (x_2, y_2), and between (x, y) and (x_3, y_3) and r expresses the fact that the circle centred at (x, y) of radius r is tangent to the three curves C_1, C_2, and C_3. The equation d_4 expresses that the distance between (x, y) and (x_4, y_4) is R. The equation of the normal n_i is half of the differential of the square distance between the point (x_i, y_i) and the point (x, y). It vanishes at the local extrema of that square distance and expresses the necessary condition for (x_i, y_i) to be a closest point on C_i from the point (x, y). These equations form a system of 12 equations

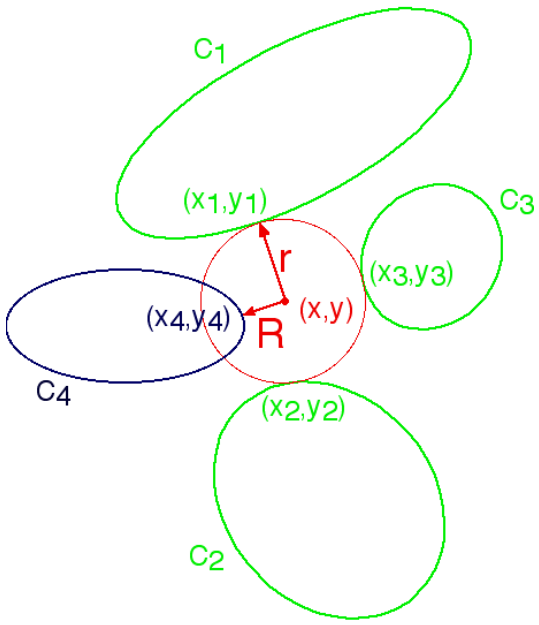

Figure 3.2.1: The Delaunay graph conflict locator that certifies whether the addition of a curve C_4 changes or does not change each one of the triangles induced by three curves C_1, C_2, and C_3

in 12 unknowns.

As in Section 3.1, we need to write programs for solving the corresponding system of algebraic equations. Such systems have a finite number of solutions (the solutions constitute a finite set of points i.e. a zero-dimensional variety). As before we consider the quotient algebra of the ring of polynomials in the variables of the system $(x_1, ..., x_N)$ by the zero-dimensional ideal I generated by the algebraic equations of the system to be solved. Computing in the quotient algebra involves generalising Euclid's algorithm for the division of one polynomial in one variable by another polynomial in one variable to the division of one polynomial f in the variables $x_1, ..., x_N$ by the generators $f_1, ..., f_s$ of I and having a canonic representative for the coset (i.e. the equivalence class) of f. This can be done by computing a

Gröbner basis of the ideal I, and then dividing the polynomial f by the generators of the Gröbner basis. If the system has a finite number of solutions (points), the quotient algebra dimension is finite and therefore, its bases are finite. We can define a multiplication map m_g in this quotient algebra that maps the coset of p to the coset of gp. Let M_g be the matrix of this linear map in the basis B of the quotient algebra. The key property of multiplication operators for evaluating a polynomial g at the points of a zero-dimensional algebraic variety is that the eigenvalues of M_g are the results of the evaluation of the polynomial g on the points of the zero-dimensional variety (see Section 3.3.1).

As before we can compute the matrices M_{x_i} of the multiplication maps by each one of the variables $(x_1, ..., x_N)$. Since these multiplication maps commute, it is possible to use the transformation matrix obtained during the computation of the Jordan form of one of these matrices to triangularise other matrices by a simple multiplication of matrices [CLO98] (see Section 3.1).

Gröbner bases can be computed in most Computer Algebra systems. Singular [GPS01] gave incorrect normal forms (i.e. the remainder of the Euclidean division of a polynomial by the polynomials of the Gröbner basis), while CoCoA [CNR00] did not give any result within a month of computation. The tests with GB/RS [Fau, Rou] have been done on the LEON machines of the UMS Medicis [CNR] at Ecole Polytechnique (1 proc alpha 500 Mhz, 640 Mo RAM) from UMS Medicis [CNR]. The tests with Macaulay 2 have been done on the LEON machines also. The tests with Maple [CGGL92] have been done on a Pentium III 550 Mhz, 1024 Mo RAM. The tests with Maxima [GG82] have been done on an Ultra 5-10 station 440 Mhz, 768 Mo RAM, because the affine module (used for Gröbner basis computations) was not available from UMS Medicis machines. The tests with Maxima were done on GIULIA machines (2 proc PIII 933 Mhz, 1 Go RAM) from UMS Medicis. The tests with toric resultants involve a preprocessing on Maple in order to produce some files

necessary for calling Emiris's [Emi97] "Resin" C program of computation of toric resultants, and a post-processing with Maxima to compute the determinant of the matrix returned by Resin where the symbolic coefficients have been replaced by their formal values. The time corresponding to the Toric Resultant row includes all these three steps. The precomputations with Maple for "Resin" were done on the same machine as the one used for the earlier mentioned tests on Maple. Similarly, the computations with Maxima after "Resin" were done on the same machine as the one used for the earlier mentioned tests on Maxima. The column titled "Canonic conic" corresponds to an equation of the conic of the form $ax^2 + by^2 + c = 0$. The column titled "Generic conic" corresponds to a generic equation of a conic of the form $ax^2 + bxy + cy^2 + dx + ey + f = 0$. The column titled "+ monomial" corresponds to a preprocessing step on the system to remove the leading monomial by linear combination of the current polynomial with the other polynomials. In the case of Gröbner basis, new variables have been introduced and these variables correspond to invariants. Invariants can be useful for rewriting the polynomials as polynomials with fewer monomials and lower degree to make them more manageable in Computer Algebra Systems. Particularly the following property of the Voronoi diagram is useful: the image of the Voronoi diagram of a set of curves by a motion is the Voronoi diagram of the images of the original curves by that motion. Even a partial rewriting of polynomials in the variables $x_1, ..., x_N$ as polynomials in $x_1, ..., x_N$ and some invariants might simplify the polynomials. The invariants that are relevant here are the fundamental invariants of the group of motions. These are the invariants of the special orthogonal group for points (scalar products and vector products of the vectors involved in the problem: the vectors between points on the original curves and the centre of the circle touching the first three curves).

The following tests deal with the Delaunay graph conflict locator for an input constituted by a circle, a parabola, a hyperbola and a circle defined with numeric parameters. The results are summarised in Table 3.1. When there is no numerical

	without mon	+ monomial
GB/RS	12 hours	segmentation fault
Toric Resultant	"not enough memory"	"not enough memory"
Maxima	bad normal forms	bad normal forms
Bezoutian-Maple	"object too large"	"object too large"
Macaulay 2	"not zero-dimensional ideal"	"not zero-dimensional ideal"
ALIAS	11 min.	11 min.

Table 3.1: The comparison of different algebraic methods for computing the Delaunay graph conflict locator for a circle, a parabola, a hyperbola and a circle defined with numeric coefficients

value, the table reports either the error reported by the Computer Algebra System between quotation marks, or the problem that induced a wrong computation (bad normal forms). The normal forms are canonic representatives of the equivalence class of the reminder of the division of one polynomial by the polynomials of an ideal.

We can observe that using invariants to simplify the writing of the polynomials before the computation does not bring any acceleration in the computation. We can also observe that approaches based on Gröbner bases induce much slower computations (GB/RS [Fau, Rou]) than interval analysis based methods (ALIAS). Finally, projective resultants based approaches do not induce any result in the case of Maple [CGGL92].

3.3 An hybrid approach linking symbolic computation and scientific computation

These first experiments with Computer Algebra Systems forced us to distance ourselves from a pure algebraic approach oriented towards Gröbner bases and to adopt a hybrid symbolic/scientific computing approach to try to find the optimum running time combination with the guarantee of a certified Delaunay graph conflict locator. This optimum corresponds to a tradeoff point between what can be

achieved with algebraic precomputations (possibly giving rise to huge matrices), and what can be achieved with numerical computations.

We will present in this section the key characteristics of the main tools that have been used to find this optimum, and to obtain the Delaunay graph conflict locator for semi-algebraic sets.

3.3.1 The sparse resultant

In this subsection, we review the main concepts and results about mixed subdivisions, mixed volumes, and sparse resultants that will be used in Section 4.4 and Chapter 5. We put the emphasis on the properties of the sparse resultant that have allowed us to get the results presented in Section 4.4 and Chapter 5. Indeed, the sparse resultant of the equations presented in the last section cannot be computed because of a lack of memory (see Table 3.1) even on machines with 4Gb of RAM and more than 6Gb of virtual memory [CNR]. We could compute the sparse resultant needed for the Delaunay graph conflict locator only after simplifying the polynomials involved to reduce the sparse complexity of the system of equations. For this purpose, we have used a key property of the Newton polytope and the mixed volume as well as a specific usage of a sparse resultant.

The key property of the mixed volume for our work with sparse resultants is that if we replace a polytope (say Q_i) by a polytope Q'_i whose Newton polytope is strictly included in the Newton polytope of Q_i, the mixed volume of $Q_1, \ldots, Q_N$ decreases. We will present this property later in this section. This key property allowed us to get the equation of the generalised offset to a conic (see Section 4.4) and the sparse resultant matrix needed for the algebraic computation of the conflict locator (see Chapter 5). Let us now introduce the concepts of Newton polytope and mixed volume and the key property mentioned in this paragraph.

The classical projective resultant of $N+1$ polynomials in $N+k$ affine variables is a polynomial in k variables, which characterises the solvability of a system

[CE00, CLO98]. Thus, it allows the elimination of N variables, and is therefore also called eliminant. The degree (i.e., the algebraic complexity) of the projective resultant of several polynomials relatively to the coefficients of one of these polynomials is the product of the degrees of the other polynomials. Sparse resultants generalise the classical (projective) resultant and exploit the monomial structure of the polynomials of the system [CE00]. The Newton polytope expresses the monomial structure of a polynomial, i.e. the monomials appearing in the polynomial. The degree of the sparse resultant is determined only by information about the exponent vectors of the polynomials and it is generally lower than the complexity of the classical projective resultant [CLO98]. The sparse resultant has been extensively treated in Canny and Emiris [CE00, EC95].

Let K be an algebraically closed field (see [Lan02] for a definition of "algebraically closed field"). Let $K[x_1, ..., x_N]$ denote the ring of polynomials in the variables $x_1, ..., x_N$ over the field K. Let $K\left[x_1^{\pm 1}, ..., x_N^{\pm 1}\right] = K\left[x, x^{-1}\right]$ denote the field of *Laurent polynomials* in the variables $x_1, ..., x_N$ over K. If $a = (a_1, \ldots, a_N) \in \mathbb{Z}^N$, then let x^a denote the monomial $x_1^{a_1} x_2^{a_2} \ldots x_N^{a_N}$. If Q is a polytope of $\mathbb{R}^N$, then let $Vol\,(Q)$ and $Vol_N\,(Q)$denote the $N-$dimensional volume of Q.

Definition 3.3.1. (Support of a polynomial [CE00, Definition 3.1, page 420]) The *support* A_i of a polynomial $f_i \in K[x_1^{\pm 1}, ..., x_N^{\pm 1}]$ is the set of exponent vectors in $\mathbb{Z}^N$ corresponding to non-zero coefficients, i.e. $f_i = \sum_{a \in A_i} c_a x^a,\ c_a \neq 0$. The Newton polytope Q_i of f_i in $\mathbb{R}^N$ is the convex hull of A_i.

For example, for the strophoid of equation $y^2 - x^2 - x^3 = 0$, the exponent vectors are: $(0, 2)$, $(2, 0)$, and $(3, 0)$.

Definition 3.3.2. (Minkowski sum [EC95, Definition 3.2, page 121]) The *Minkowski sum* $A+B$ of point sets A and B in $\mathbb{R}^N$ is the point set $A+B = \{a + b | a \in A, b \in B\}$ (see example on Figure 3.3.1).

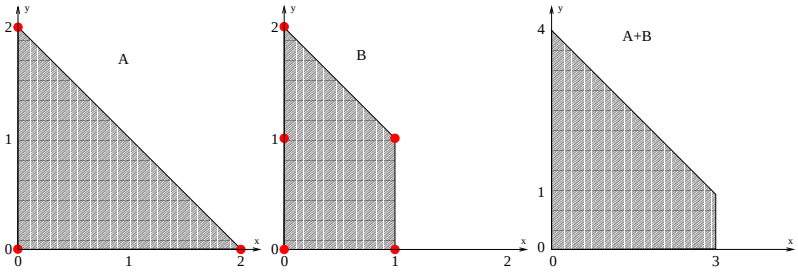

Figure 3.3.1: The Minkowski sum of the point sets A and B

Definition 3.3.3. (Mixed volume [CE00, Definition 3.4, page 421]) Let N polytopes $Q_1, ..., Q_N \subset \mathbb{R}^N$ whose vertices belong to $\mathbb{Z}^N$. The *mixed volume* $MV(Q_1, ..., Q_N)$ is the coefficient of the monomial $\lambda_1 ... \lambda_N$ in $Vol(\lambda_1 Q_1 + ... + \lambda_N Q_N)$.

Definition 3.3.4. (Polyhedral subdivision [CE00, Definition 4.3, page 421]) A *polyhedral subdivision* of a compact set $S \subset \mathbb{R}^N$ is a collection of polyhedra whose union equals S, such that each intersection of two polyhedra of the same dimension is another polyhedron in the subdivision of lower dimension. The polyhedra of maximal dimension are called maximal cells or facets.

Definition 3.3.5. (Mixed subdivision [CLO98, Definition 6.5, page 344]) Let $Q = Q_1 + ... + Q_m \subset \mathbb{R}^N$ be a Minkowski sum of polytopes, and assume that Q has dimension N. Then a subdivision $R_1, ..., R_s$ of Q is a *mixed subdivision* if each cell R_i can be written as a Minkowski sum $R_i = F_1 + ... + F_m$ where each F_i is a face of Q_i and $N = dim(F_1) + ... + dim(F_m)$ (see example on Figure 3.3.2).

Definition 3.3.6. (Mixed cell [CLO98, Definition 6.6]) Suppose that $R = F_1 + ... + F_m$ is a cell in a mixed subdivision of $Q = Q_1 + ... + Q_m$. Then R is called a *mixed cell* if $dim(F_i) \leq 1$ for all i.

Theorem 3.3.7. *([CLO98, Theorem 6.7]) Given polytopes $Q_1, ..., Q_N \subset \mathbb{R}^N$ and a mixed subdivision of $Q = Q_1 + ... + Q_N$, the mixed volume $MV_N(Q_1, ..., Q_N)$ is computed by the formula $MV_N(Q_1, ..., Q_N) = \Sigma_R Vol_N(R)$, where the sum is over all mixed cells R of the mixed subdivision.*

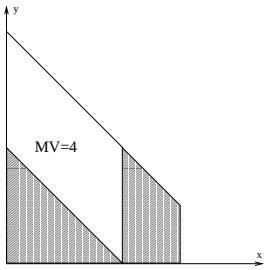

Figure 3.3.2: A mixed subdivision of the Minkowski sum shown in Figure 3.3.1 and the corresponding mixed volume (the mixed cells are not filled and the filled cells are copies of the Newton polytopes)

Proposition 3.3.8. *(Bernstein's Theorem [EC95, Theorem 3.6, page 122]) For the system* $p_1, ..., p_N \in K\left[x_1^{\pm 1}, ..., x_N^{\pm 1}\right] = K\left[x, x^{-1}\right]$*, the sum of the multiplicities (see Definition 4.1.5) of solutions in* $\left(K^\star\right)^N$ *is infinite, or bounded by the mixed volume of the Newton polytopes. If the coefficients are generic, then this bound is exact.*

Generally, this bound (also known in the literature as the BKK bound because of the existence of closely related papers by Kushnirenko and Khovanskii) is lower than the *Bézout number* (i.e., the product of the degrees of the polynomials), which expresses the sum of multiplicities of isolated solutions in $\mathbb{P}^N$.

Now let us introduce the sparse resultant and its use for the computation of the implicit equation of the generalised offset to a conic (see Section 4.4) and the multiplication operator matrix needed for the algebraic computation of the conflict locator (see Chapter 5). The main interest of the sparse resultant in the computation of the implicit equation of the generalised offset to a conic is that it allows one to do the elimination of N variables from $N + 1$ polynomials like projective multi-polynomial resultants with an algebraic complexity that is lower than the Bézout number, and therefore, the matrices obtained are smaller, and the computations can be performed faster.

In order to introduce the sparse resultant, let us recall some basic definitions:

Definition 3.3.9. (Zariski topology, adapted from [GP02, Definition A.2.1, para-

graph following Lemma A.2.4 and Lemma A.4.3]) In the *Zariski topology*, a *closed set* is a subset consisting of all common zeroes of finitely many polynomials with coefficients in K. A *quasi-projective variety* is an open subset of a closed projective set. The *Zariski closure* of a set X is the smallest closed set containing X.

The sparse resultant is a necessary condition of existence of solutions of a system of algebraic equations in $(\mathbb{C}^{\star})^N$:

Given a set of exponent vectors $\mathcal{A}=\{\alpha_1,...,\alpha_l\}\subset\mathbb{Z}^N$, let $\mathcal{L}(\mathcal{A})=\{c_1x^{\alpha_1}+...+c_lx^{\alpha_l}:c_i\in\mathbb{C}\}$ be the set of polynomials whose terms all have exponents in $\mathcal{A}$. Consider $N+1$ Laurent polynomials $f_i\in\mathcal{L}(\mathcal{A}_i)$. Let $Q_i=\text{Convex hull}(\mathcal{A}_i)$ and $MV_{-i}=MV(Q_0,\ldots,Q_{i-1},Q_{i+1},\ldots,Q_N)$. Let $Z(\mathcal{A}_0,...,\mathcal{A}_N)\subset\mathcal{L}(\mathcal{A}_0)\times...\times\mathcal{L}(\mathcal{A}_N)$ be the Zariski closure of the set of all $(f_0,...,f_N)$ for which $f_0(x_1,...,x_N)=...=f_N(x_1,...,x_N)$ has a solution in $(\mathbb{C}*)^N$.

Definition 3.3.10. (Sparse resultant, [CLO98, Theorem 6.2 p. 342]) Assume that Q_i is an N-dimensional polytope for all i. Then, there is an irreducible polynomial called the *sparse resultant* $Res_{\mathcal{A}_0,...,\mathcal{A}_N}$ in the coefficients of the f_i such that $(f_0,...,f_N)\in Z(\mathcal{A}_0,...,\mathcal{A}_N)\Leftrightarrow Res_{\mathcal{A}_0,...,\mathcal{A}_N}=0$.

However, it is not sufficient generally.

On the toric variety constructed from the Minkowski sum of the Newton polytopes of the polynomials of a system of algebraic equations, the sparse resultant is a necessary and sufficient condition of existence of solutions of the system. Let us define first the toric variety constructed from the Minkowski sum of the Newton polytopes of the polynomials of a system of algebraic equations. For this purpose, we recall that $\mathbb{P}^N$ denotes the $N-$dimensional projective space over K, i.e., the set of lines of K^{N+1} going through the origin O of the coordinate system of K^{N+1} (adapted from [GP02, Definition A.4.1]).

Definition 3.3.11. (Toric variety, [CLO98, p. 307]) Let $\mathcal{A}=\{m_1,...,m_l\}\subset\mathbb{Z}^N$, and suppose that $f_i=a_{i1}t^{m_1}+...+a_{il}t^{m_l}$, $i=0,...,N$ are $N+1$ Laurent polynomials

in $L(\mathcal{A})$. Assume that the convex hull of $\mathcal{A}$ has dimension N. Then consider the map $\phi_{\mathcal{A}} : (\mathbb{C}*)^N \mapsto \mathbb{P}^{l-1}$ defined by $\phi_{\mathcal{A}}(t_1, ..., t_N) = (t^{m_1}, ..., t^{m_l})$. Then, the *toric variety* $\mathcal{X}_A$ is the Zariski closure of the image of $\phi_{\mathcal{A}}$.

The necessary and sufficient condition of existence of solutions of the system in the preceding toric variety is:

Theorem 3.3.12. *(adapted from [CLO98, Theorem 3.4]) $Res_{\mathcal{A}}(f_0, ..., f_N) = 0$ if, and only if, $f_0 = ... = f_N = 0$ has a solution in $\mathcal{X}_A$.*

The sparse resultant is a factor of the determinant of the Newton matrix, which can be computed using a mixed subdivision [CE00, CLO98]. The sparse elimination using the sparse resultant allowed us to obtain an implicit equation of the generalised offset to a conic (see Section 4.4).

The main interest of the sparse resultant of the polynomials $f_0, ..., f_N$ in the variables $x_1, ..., x_N$ in the algebraic computation of the Delaunay graph is that it allows one to compute the values taken by the polynomial f_0 at $V(f_1, ..., f_N)$. The sparse resultant matrix allows one to do computations in the quotient algebra $A = K[x_1, \ldots, x_N] / \langle f_1, ..., f_N \rangle$. We will see this now. We need now to introduce some notations for this purpose. Let $f_0, ...f_N$ be $N + 1$ Laurent polynomials. Let $Q_0, ..., Q_N$ be their Newton polytopes. Let R be the sparse resultant of the polynomials $f_0, ...f_N$. Let $deg_{f_i} R$ denote the total degree of the resultant R in the coefficients of the polynomial f_i. Let $MV_{-i} = MV(Q_0, ..., Q_{i-1}, Q_{i+1}, ..., Q_N)$.

Theorem 3.3.13. *([EC95, Theorem 3.10]) The sparse resultant is separately homogeneous in the coefficients c_i of each f_i and its degree in these coefficients equals the mixed volume of the other N Newton polytopes, i.e., $deg_{f_i} R = MV_{-i}$.*

From the last theorem, it follows that the total degree of the sparse resultant equals the sum of the mixed volumes of N Newton polytopes: $deg(R) = \sum_{i=0}^{N} MV_{-i}$. Let $Q = Q_0 + ... + Q_N$. For an infinitesimal vector $\delta \in \mathbb{Q}^N$, we define $\tilde{\varepsilon} = (Q + \delta) \cap \mathbb{Z}^N$.

Theorem 3.3.14. *[CLO98, Theorem 6.17 p. 354] For the set $\tilde{\varepsilon}$ described above, the cosets $[x^{\beta}]$ for $\beta \in \tilde{\varepsilon}$ form a basis of the quotient ring $\mathbb{C}\left[x_1^{\pm 1}, .., x_N^{\pm 1}\right] / \langle f_1, ..., f_N \rangle$.*

We define $\mu : \tilde{\varepsilon} \to \bigcup_i \mathcal{A}_i : p \mapsto a_{ij} \in \mathcal{A}_i \Leftrightarrow p \in \sigma = F_0 + ... + a_{ij} + ... + F_N, dim\, F_j > 0, \forall j > i$, and $B_i = \{p - \mu(p) : p \in \tilde{\varepsilon}, \mu(p) \in \mathcal{A}_i\}$. We decompose the sparse resultant matrix (i.e. the Newton matrix): $M_0 = \begin{pmatrix} M_{00} & M_{01} \\ M_{10} & M_{11} \end{pmatrix}$ so that the rows and columns of M_{00} correspond to elements of B_0. The elements of B_0 are in one-to-one correspondence with the integer coordinates points of the mixed cells of any mixed subdivision of $f_1, ..., f_N$. Thus, the cardinality of B_0 is the mixed volume MV_{-0}. Let $A = \mathbb{C}\left[x_1^{\pm 1}, .., x_N^{\pm 1}\right] / \langle f_1, ..., f_N \rangle$ be the quotient algebra of the ring of polynomials $\mathbb{C}\left[x_1^{\pm 1}, .., x_N^{\pm 1}\right]$ by the ideal $\langle f_1, ..., f_N \rangle$ generated by $f_1, ..., f_N$.

Theorem 3.3.15. *[CLO98, Theorem 6.21, p. 356] Let $f_i \in L(\mathcal{A}_i)$ be generic Laurent polynomials, and let $f_0 = u_0 + u_1 x_1 + ... + u_N x_N$. Using the basis from Theorem 3.3.14, the matrix M_{f_0} of the multiplication map $m_{f_0} : A \to A$ which maps $[g]$ to $[f_0 g]$ is the transpose of the Schur complement of the matrix M_{11}: $\tilde{M} = M_{00} - M_{01} M_{11}^{-1} M_{10}$.*

Thus the size of the matrix of the multiplication map m_{f_0} is the mixed volume of $Q_1, ..., Q_N$, which is the sparse bound for the number of common zeroes of $f_1, ..., f_N$ (counted with their multiplicities). This is coherent with the fact the dimension of the quotient algebra is the number of common zeroes of $f_1, ..., f_N$ (counted with their multiplicities). The values taken by f_0 on $V(f_1, ..., f_N)$ are the eigenvalues of the matrix of the multiplication map $m_{f_0} : A \to A$ (see [CLO98, Theorem 4.5])

Moreover, since $M_{f_0} = u_0 I + u_1 M_{x_1} + ... + u_N M_{x_N}$, where M_{x_i} is the matrix of the map of the multiplication by x_i, by [CLO98, Corollary 4.3, p. 53], $\tilde{M} = u_0 I + u_1 \tilde{M}_1 + ... + u_N \tilde{M}_N$, where each $\tilde{M}_i$ is obtained as in Theorem 3.3.15. Then $M_{f_0} = \tilde{M}^T$ implies that $M_{x_i} = \tilde{M}_i^T$.

As observed in Section 3.1, we can compute the matrices M_{x_i} of the multiplication maps by each one of the variables $(x_1, ..., x_N)$ by a simultaneous diagonalisation of all these matrices [CLO98].

3.3.2 Numerical methods for computing exactly the signs of the eigenvalues of large sparse matrices

In this subsection, we present numerical methods for computing exactly the signs of the eigenvalues of large sparse matrices used in the numerical computation of the conflict locator that follows the algebraic precomputation of the matrix of the sparse resultant for the Delaunay graph conflict locator (see Section 5.5). The results reported in this section can be found in [Hea02]. For computing all the eigenvalues of an arbitrary real matrix, the standard approach is to reduce the matrix to the Hessenberg form (lower Hessenberg: $a_{ij} = 0$ for $i > j + 1$, or upper Hessenberg: $a_{ij} = 0$ for $i < j - 1$), and then apply QR iteration on that Hessenberg matrix [Hea02]. However, for very large sparse matrices, like the matrix of the sparse resultant or the multiplication operator matrix for the Delaunay graph conflict locator computation, standard algorithms for reduction to the Hessenberg form are prohibitively time and memory consuming [Hea02].

The current method of choice for general sparse square matrices of size N is the Arnoldi method [Hea02]. The Arnoldi method for large sparse non-symmetric eigenvalue problems is implemented in ARPACK [LSY98]. It is the basis for the MATLAB [Hea02] function eigs for computing a few (the six having highest modulus by default) to almost all (up to the $N - 1$ having the highest modulus) eigenvalues and eigenvectors of a matrix.

The Arnoldi method is a Krylov subspace method, whose main assumption is that the input matrix is best considered as a linear operator, with which one can form matrix-vector products. Krylov subspace methods are based on a

simple method (known as the power iteration) for computing a single eigenvalue of an $N \times N$ matrix A, which multiplies an arbitrary nonzero vector by successively higher powers of the matrix A. Assuming A has a unique eigenvalue λ_1 of maximum modulus and v_1 is its corresponding eigenvector, power iteration converges to a multiple of v_1. Subspace iteration methods (also known as simultaneous iteration methods in the literature) use power iteration with several different starting vectors to compute several eigenvalues of a matrix simultaneously. The sequence of vector spaces spanned by the product of these starting vectors by successively higher powers of the matrix A will converge towards the invariant space spanned by the eigenvectors $v_1, ..., v_p$ corresponding to the p largest eigenvalues of A in modulus $\lambda_1, ..., \lambda_p$. Krylov subspace methods provide similarity reduction to the Hessenberg form using only matrix-vector multiplication. The application of the mathematical setup presented above for computing eigenvalues is difficult because the columns of the matrices computed by the Krylov subspace methods converge towards multiples of the dominant eigenvector of A and thus, they become exceedingly ill-conditioned. In order to remedy to this an orthonormalisation (othogonalising the new vector with respect to all the previous ones and normalising it) of the vector used for matrix-vector multiplication is done at each iteration. This algorithm is owed to Arnoldi [Hea02]. Since the Arnoldi method requires at iteration k a matrix-vector multiplication by A plus $O(kN)$ for the orthonormalisation, plus $O\left(k^3\right)$ for the computation of the eigenvalues, it is run for a few iterations and then restarted with a new starting vector that is relatively rich in components of the desired eigenvectors. A few repetitions of the restarted Arnoldi process produce excellent approximations to the extreme eigenvalues of A.

The certified computation of the sign of eigenvalues of sparse matrices can be done by computing tight bounds on the intervals taken by eigenvalues [Krä92]. This method gives an interval bound for the exact eigenvalue without the user providing error estimations. The existence of the true eigenvalue within the computed bounds

is "simultaneously proven by the method" [Krä92].

3.3.3 ALIAS

In this subsection, we present the key properties of the ALIAS library that have been used for the computation of the Delaunay graph conflict locator for semi-algebraic sets presented in Chapter 6. Interval analysis is a well-known method for computing bounds of a function, being given bounds on the variables of that function [Han92, Mer01]. The basic mathematical object in interval analysis is the interval instead of the variable. The operators need to be redefined to operate on the intervals instead of the variables. This leads to an interval arithmetic. In the same way, most usual mathematical functions are redefined by an interval equivalent. Many packages implement basic interval operations [Han92, Mer01]. However, interval analysis has two main drawbacks. The first drawback is that the bounds of a function depend heavily on the way the function is written [Mer01]. This can lead to or be accompanied with an over-estimation of the bounds [Mer01]. The second drawback is that it is difficult to test rapidly the efficiency of an interval analysis based algorithm [Mer01]. The implementation of such an algorithm involves three different levels of software: a basic level where the basic operations of interval arithmetic are performed, an end-user level where the bounds for the function to be evaluated will be computed using the functions of the first level, and an algorithm level where the function evaluation of the second level will be used to solve a problem [Mer01]. ALIAS [Mer00] is a library of algorithms enabling to analyse and solve zero-dimensional systems of equations and inequalities with real coefficients. ALIAS allows "the user to focus on the algorithmic part of the problem, while offering a convenient way to use interval analysis and, furthermore, enabling one to easily change the analytic form of the function that will be evaluated" (from [Mer01]). ALIAS is composed of a C++ library (the kernel of ALIAS), a Maple [CGGL92] interface (which enables to produce the code that will solve a system of equations

and inequalities directly from MAPLE by using the C++ library), and a parser (which enables to perform the interval evaluation of a function whose analytical formulation is written in a file).

The analysis of the zero-dimensional system of equations and inequalities encountered in the Delaunay graph conflict locator computation (see Chapter 6) relies on results about the uniqueness of a root in an interval (Kantorovitch [Kan57] and Moore-Krawczyk [Moo77]).

The mathematical setup is as follows [Han92]. An *interval number* is a real, closed interval $(\underline{x}, \overline{x})$. Arithmetic rules are replaced by *interval arithmetic rules*, e.g. let two interval numbers $X = (\underline{x}, \overline{x})$, $Y = \left(\underline{y}, \overline{y}\right)$, then $X + Y = \left(\underline{x} + \underline{y}, \overline{x} + \overline{y}\right)$ and $X - Y = \left(\underline{x} - \overline{y}, \overline{x} - \underline{y}\right)$. An *interval function* is an interval-valued function of one or more interval arguments. An interval function F is said to be *inclusion monotonic* if, $X_i \subset Y_i$ for $i \in [1, N]$ implies $F(X_1, ..., X_N) \subset F(Y_1, ..., Y_N)$. A fundamental theorem is that any rational interval function evaluated with a fixed sequence of operations involving only addition, subtraction, multiplication and division is inclusion monotonic [Han92].

Theorem 3.3.16. *Moore theorem [Moo77] Let a system of N equations in N unknowns: $F = \{F_i(x_1, ..., x_N) = 0, i \in [1, N]\}$ each F_i being at least C^1. Let $\mathbf{X}$ be an interval vector for $\{x_1, ..., x_N\}$, y a point inside $\mathbf{X}$ and Y an arbitrary nonsingular real matrix. Define K as $K(\mathbf{X}) = y - YF(y) + \{I - YF'(\mathbf{X})\}(\mathbf{X} - y)$. If $K(\mathbf{X}) \subset \mathbf{X}$ and $\|I - YF'(X)\| < 1$ then there is a unique solution of F in $\mathbf{X}$.*

This unique solution can be found using the Krawczyk solving method [Moo77].

Theorem 3.3.17. *Kantorovitch theorem [Kan57] There exists a unique solution x^* of the functional equation $P(x) = 0$, where P is an operator twice continuously differentiable, which maps the normed space X onto Y, if the following conditions are satisfied:*

1. *There exists a real valued majoring function* $Q(x)$ *on an interval* (z_0, z') *(i.e.,* $\|P(x_0)\| \leq Q(z_0)$ *and* $\|P'(x)\| \leq Q'(z)$ *if* $\|x - x_0\| \leq z - z_0 \leq z' - z_0$*) for which the relation* $Q(z) = 0$ *has real roots* z_1, z_2 $(z_0 \leq z_1 \leq z_2 \leq z')$*;*

2. *There exists an inverse operator* $\Gamma_0 = -[P'(x_0)]^{-1}, B = -[Q'(z_0)]^{-1} > 0$*;*

3. $\|\Gamma^0 P(x_0)\| \leq BQ(z_0)$*;*

4. $\|\Gamma^0 P"(x)\| \leq BQ"(z)$ *for* $\|x - x_0\| \leq z - z_0 \leq z' - z_0$.

The solution x^* *is bounded by* $\|x^* - x_0\| \leq z_1 - z_0$ *and furthermore it is unique in* $\|x - x_0\| \leq z_2 - z_0$*. The approximations* x_N *obtained by the Newton method (*$x_{N+1} = x_N - [P'(x_N)]^{-1} P(x_N)$*) and its modification (*$x_{N+1} = x_N - [P'(x_0)]^{-1} P(x_N)$*) converge to* x^*.

All the general purpose solving procedures for zero-dimensional systems have been tested in the computation of the Delaunay graph conflict locator for semi-algebraic sets. These general purpose solving procedures are based on a bisection process on one (*single bisection*), several (*mixed bisection*) or all the variables (*full bisection*) using either:

- only the equations and inequalities of the system,
- the equations and inequalities of the system and the Jacobian of the system (Moore-Krawczyk test for finding "exactly" the solutions),
- the equations and inequalities of the system and the Jacobian and Hessian of the system (with Kantorovitch and Moore-Krawczyk tests).

The bisection process (single, mixed or full) can be set by setting the "ALIAS/single_bisection" parameter (to 2, 1 or 0) and in the case of mixed bisection, the number of bisected variables can be set by setting the "ALIAS/mixed_bisection" parameter. The variables that will be bisected will be the ones having the largest interval width.

Let $x_1, ..., x_m$ be the set of unknowns and let $\mathcal{I}_1 = \left\{ \left(\underline{x_1^1}, \overline{x_1^1} \right), ..., \left(\underline{x_m^1}, \overline{x_m^1} \right) \right\}$ be the set of m intervals in which we are searching the solutions of the N equations or inequalities $\mathcal{F}_1 (x_1, ..., x_m) \{=, \geq\} 0, ..., \mathcal{F}_N (x_1, ..., x_m) \{=, \geq\} 0$. The indices in the x_i^e are unknowns indices, while the exponents in the x_i^e (and indices in the $\mathcal{I}_j$) are iteration indices. Let F_i be the interval value of $\mathcal{F}_i$ when this equation is evaluated for the interval value $\left(\underline{x_1}, \overline{x_1} \right), ..., \left(\underline{x_m}, \overline{x_m} \right)$ of the unknowns while $F(\mathcal{I}_j)$ be the $N-$dimensional interval vector constituted of the F_i when the unknowns have the interval value defined by the set $\mathcal{I}_j$.

The algorithms use a list of interval vectors (or *boxes*) $\mathcal{I}$ whose maximal size M is an input of the program. The general purpose solving algorithm bisects the input intervals until either their width is lower than an accuracy on the variables ϵ or the width of the equation interval is lower than an accuracy on the equations ϵ_F (provided there is enough storage space in the list to store the intervals) [Mer00]. Then, if all the equations and inequalities intervals are acceptable (they contain 0 for an equation or they contain positive or negative values according to the sense for an inequality), we get a new solution, if one of them is not acceptable, there is no solution of the system within the current variable intervals. The new interval vectors are added to the list of $\mathcal{I}$ with an ordering which aims at considering first the input intervals having the highest probability of containing a solution. There are two ordering criteria: maximum equation ordering and maximum middle point equation ordering. In the maximum equation ordering, the boxes are ordered along the value of $C = Max \left(\overline{F_k(\mathcal{I})}, \underline{F_k(\mathcal{I})} \right)$ for all k in $[1, N]$ (the first box will have lowest C). In the maximum middle-point ordering, the boxes are ordered along the value of $C = Max \left(\overline{F_k(C_i)}, \underline{F_k(C_i)} \right)$ where C_i is the vector whose components are the middle points of the intervals $\mathcal{I}$. This full bisection process on all the variables simultaneously may induce a combinatorial explosion (at each iteration, 2^N new interval vectors or boxes are produced). Instead of bisecting on all the variables simultaneously, it is possible to bisect only one variable at each iteration. This may

reduce the computation time as the number of function evaluations may be reduced [Mer00]. The variable that will be bisected is the one for which the bisection will produce the intervals with the lowest criteria except if for at least one variable the intervals that will result from the bisection cannot possibly contain a solution. Moreover, to avoid bisecting always the same variable, another test is used: let d_i be the width of the interval $\left[\underline{x_i}, \overline{x_i}\right]$ and d_{max} be the maximum of all the d_i; if $\frac{d_i}{d_{max}} < 0.1$, x_i is not considered as a possible bisection variable.

Aside from these bisection processes, it is possible to use another bisection method called the $3B$ approach (by setting the "ALIAS/3B" parameter to 1 and the maximal range "ALIAS/Max3B" and minimal range "ALIAS/Delta3B" parameters). Each variable x_i and its range $\left[\underline{x_i}, \overline{x_i}\right]$ are considered in turn. Let x_i^m be the middle point of this range. First, the interval evaluations for the equations and inequalities in the system with the full ranges of the variables except for the variable i where the range is $\left[\underline{x_i}, x_i^m\right]$ are computed. Clearly, if one of the equations or inequalities is not satisfied, it is possible to reduce the range of the variable i to $[x_i^m, \overline{x_i}]$. If this is not the case, let's define a new x_i^m as the middle point of the interval $\left[\underline{x_i}, x_i^m\right]$ and repeat the process until either we have found an equation or an inequality that is not satisfied (inducing a new reduction of a variable interval) or the width of the interval $\left[\underline{x_i}, x_i^m\right]$ is lower than a given threshold δ. A similar procedure can be used to reduce the input interval on the right. We may additionally select a subset of equations and/or inequalities whose intervals will be evaluated. This can be done by setting the "ALIAS/SubEq3B" variable [Mer00].

Chapter 4

The offset to an algebraic curve

While the offset (i.e. locus of points at a given distance, see an example on Figure 4.1.1) to a curve or a surface and the bisector of two curves or surfaces have been studied for their applications (see [HV91]), nothing has been written about the degree of the polynomials defining these objects, and no implicit equation has been given, even in the simple case of conics. In the reminder of this thesis, we will call the offset true offset (to contrast it with the generalised offset, that we will review in this chapter). What Hoffmann and Vermeer [HV91] define as offset curves, Arrondo, Sendra and Sendra [ASS99] define as generalised offset curves (see Figure 4.1.2). The extraneous solutions (corresponding to a singular point of the curve or surface, for example, the circle centred on the self intersection of the strophoid on Figure 4.1.2 and of radius the offset parameter) have been addressed in [HV91]. Hoffmann and Vermeer [HV91] did not address the computations, but they gave some examples computed using Gröbner bases (see also [Hof90]). Arrondo, Sendra and Sendra [ASS99] computed the genus of the generalised offset curve when the field has characteristic [Lan02, end of Section 2 of Chapter II] zero. Farouki and Neff studied the analytic properties [FN90b] as well as the algebraic properties [FN90a] of the true offset to a planar parametric curve.

In this chapter, we will address the degree of the true offset to an algebraic

plane curve in its most general setting, i.e. in an algebraically closed field of zero characteristic. Knowing the degree of the generalised offset to conics allowed us to identify the factor of the sparse resultant that correspond to the implicit equation of the generalised offset to a conic (see Section 4.4). This implicit equation is central to the formalisation and computation of the Delaunay conflict locator for conics (see Chapter 5 and Section 6.3). Moreover, the degree of the generalised offset to conics determines the Bézout bound[1] on the degree of the algebraic variety on which we will evaluate the Delaunay graph conflict locator. Our main contributions are a general formula for the degree of the true offset curve and its application for the determination of an implicit equation of the generalised offset to a conic (which is the Zariski closure of the true offset, see Section 4.1). The conic is defined implicitly by a formal polynomial (i.e. a polynomial whose coefficients are formal constants). We used the general formula for the degree of the true offset curve to eliminate the extraneous factors from the sparse resultant [CE00, CLO98] to get an implicit equation defining the generalised offset to a conic (see Section 4.4).

This chapter is organised as follows: in section 4.1, we study the equation of the offset. In section 4.2, we study the algebraic properties of the true offset to an algebraic curve in order to determine its degree. In section 4.3, we apply the results of section 4.2 to the conics. In section 4.4 we use the results of sections 4.3 and 3.3.1 to compute an implicit equation of the generalised offset to a conic.

4.1 Equations defining the offsets

Let us first recall some basic definitions about algebraically closed fields, rings of polynomials and algebraic varieties. Let K be a zero characteristic *algebraically closed field*, i.e., a field such that all polynomials with coefficients in K have a root in K [Lan02, Definition before Theorem 1, Section 2, Chapter VII] and $\forall x \in K, N \in \mathbb{N} : Nx \neq 0$. Let $K[x_1, ..., x_N]$ be the *ring of polynomials* [GP02, Defin-

[1] In this case, the Bézout number is the degree of the generalised offset to the power 4

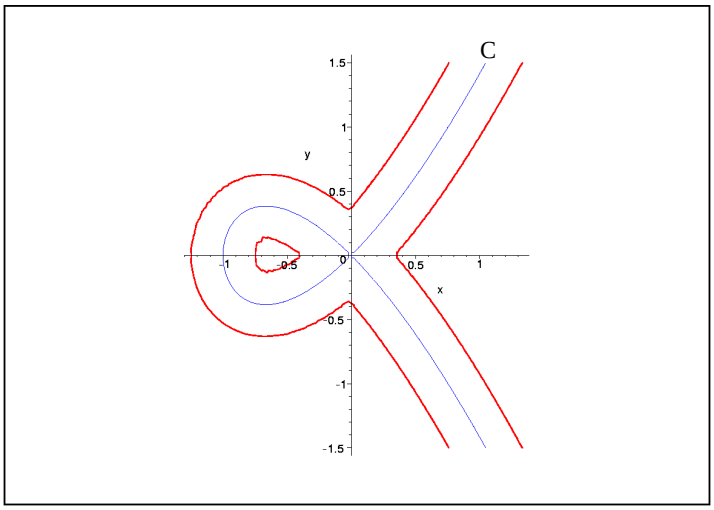

Figure 4.1.1: The strophoid (C) and its true offset (thick lines)

ition 1.1.3] in the variables $x_1, ..., x_N$ with coefficients in K. In all this chapter, we assume that $V(f_1, ..., f_s) \subset E$ denotes the *algebraic variety* embedded in E defined by the polynomials $f_1, ..., f_s$, i.e., the set of all the points of E whose coordinates are common zeroes of all the polynomials $f_1, ..., f_s$ [Sha94]. If $E = K^N$ is an affine space, then the variety is an *affine variety* [GP02, Definition A.2.1]. If $E = \mathbb{P}^N$ is a projective space over K, then the variety is a *projective variety* [GP02, Definition A.4.2]. We can now recall the notion of degree. The *degree* of a projective variety $X \subset \mathbb{P}^N$ is the maximum number of points of intersection of X with a projective linear subspace $\mathbb{P}^{N-\dim X}$ of complementary dimension in general position with respect to X (see page 234 in [Sha94]). Thus, the degree of a projective variety is the degree of its maximal dimensional component.

In this chapter, we focus on algebraic curves in the affine space $K^2 = \mathbb{C}^2$. We thus suppose $K = \mathbb{C}$ in the remaining of this thesis.

We will now introduce the true offset curve.

Definition 4.1.1. (True offset) Let $C = V(f) \subset K^2$, for $f \in K[x, y]$, be an algebraic curve and $R \in \mathbb{R}^+$ be the offset parameter. The *true* $R-$*offset curve* to C

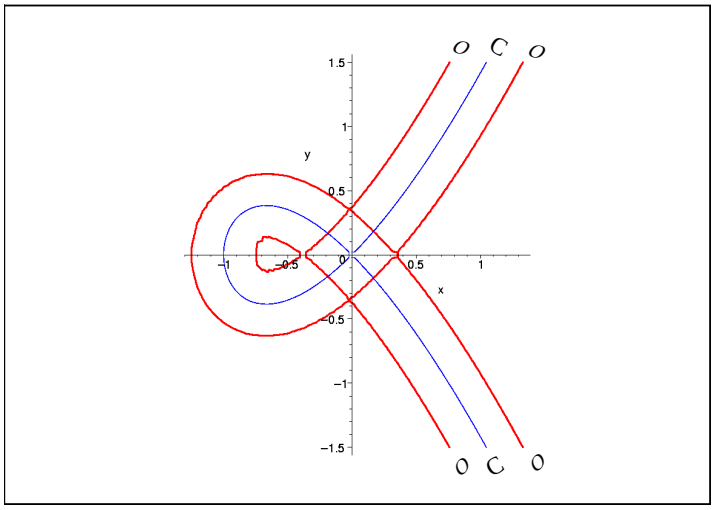

Figure 4.1.2: The strophoid and its generalised offset

is the locus of points of $\mathbb{R}^2$ being at the distance R from C (see Figure 4.1.1).

Remark 4.1.2. This is equivalent to saying that each point $q = (u, v)$ of the true offset curve is the centre of a circle $\mathcal{D}$ of radius R that is tangent to C, and does not contain any point of C in its interior.

We will suppose that f has positive degree (i.e., f is not constant), which implies that C is not empty and not equal to K^2. We will also suppose that $R \neq 0$ unless stated otherwise.

Let us now introduce the generalised offset and emphasize its differences with the true offset. There exists a superset of the true R-offset curve, called the generalised R-offset curve and denoted by $\mathcal{O}$ that is defined as the locus of points that are *locally* at the distance R from the given curve (see example on Figure 4.1.2):

Definition 4.1.3. (Generalised offset, adapted from [ASS99]) The *generalised offset* to a hypersurface ν at distance R is the Zariski closure of the set of intersection points of the spheres with centre on a non-singular point of ν and radius R, and the normal lines to ν at the centre of the spheres.

This is equivalent to saying that each point $q = (u, v)$ of the generalised offset

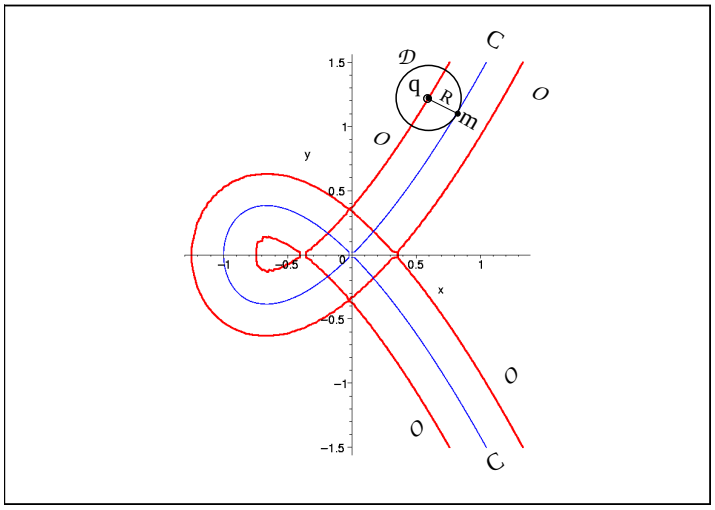

Figure 4.1.3: Relationships between the generalised offset to the strophoid and the strophoid

curve $\mathcal{O}$ is the centre of a circle $\mathcal{D}$ of radius R that is tangent to C, but *may contain* points of C in its interior (see Figure 4.1.3).

We will now establish the systems of equations and inequalities that define the generalised R-offset $\mathcal{O}$ and the true R-offset to an algebraic curve C. For this purpose, we will describe which points have to be removed from the generalised offset in order to get the true offset. Let us first introduce the following notation and definitions. Let $q = (u, v)$ be an arbitrary point on the generalised R-offset (see Figure 4.1.3).

If C is the affine variety defined by $f \in K[x, y]$ (i.e. $C = V(f)$), then the normal to C at a given point $m = (\alpha, \beta) \in C$ is the variety defined by n (i.e. $V(n)$), for

$$n(u, v) = -f_y(m) \cdot (u - \alpha) + f_x(m) \cdot (v - \beta)$$

where $f_x = \frac{\partial f(x,y)}{\partial x}$ and $f_y = \frac{\partial f(x,y)}{\partial y}$ denote the partial derivatives of f.

Definition 4.1.4. (Tangent space [Sha94]) The *tangent space* to an algebraic variety $V(f_1, ..., f_s) \subset E$ at $m \in V(f_1, ..., f_s)$ is the locus of points on lines tangent to $V(f_1, ..., f_s)$ at m.

Definition 4.1.5. (Singular point) A point m of an algebraic variety $V(f_1, ..., f_s) \subset E$ where $f_i \in K[x_1, x_2, ..., x_N]$ is called a *singular point* if, and only if, the tangent space at m is E, or equivalently, $\frac{\partial f_i}{\partial x_j}(m) = 0$ for all $i = 1, .., s$ and $j = 1, .., n$.

We have already seen an example of singular point: the self intersection of the strophoid.

We are ready to describe the points that belong to both the generalised offset and the true offset to an algebraic curve. For a given $q = (u, v) \in \mathbb{R}^2$, let $\mathcal{M}$ be the set of points $m = (\alpha, \beta) \in C \bigcap \mathbb{R}^2$ such that $q \in V(n)$. This condition is achieved whenever the normal to C at m passes through q, or m is singular. In the general case, $\mathcal{M}$ is finite. However, if C is a circle centred on q, then $\mathcal{M} = C \bigcap \mathbb{R}^2$. To get in all cases a finite set of points m of $C \bigcap \mathbb{R}^2$ such that $q \in V(n)$, we use $\mathfrak{S} = \mathcal{M}$ when $\mathcal{M}$ is finite, and $\mathfrak{S} = \{w\}$ for an arbitrary point w of $C \bigcap \mathbb{R}^2$ when C is a circle centred on q.

Lemma 4.1.6. *The set of all the closest points on $C \bigcap \mathbb{R}^2$ from q is contained in $\mathcal{M}$.*

Proof. The polynomial n defining $\mathcal{M}$ expresses half of the differential of the scalar product $\overrightarrow{qm} \cdot \overrightarrow{qm}$ (which is equal to the square of the Euclidean distance $\delta(q, m)$) with respect to m. The closest points on $C \bigcap \mathbb{R}^2$ from q are global minima of the Euclidean distance $\delta(q, m)$, so, the differential (and thus, n) vanishes on them. □

Recall the definition of the power of a point p, with respect to a circle centred at c of radius r: it is equal to $\overrightarrow{cp} \cdot \overrightarrow{cp} - r^2$. The power is positive, zero or negative if p is outside, on, or inside the circle respectively.

Let $\mathcal{D}$ be the circle of radius R centred on q (see Figure 4.1.3).

Lemma 4.1.7. *The minimum power of the points of $C \bigcap \mathbb{R}^2$ with respect to $\mathcal{D}$ is at least the minimum power of the points of $\mathfrak{S}$ with respect to $\mathcal{D}$.*

Proof. The points of $C \bigcap \mathbb{R}^2$ having minimum power with respect to $\mathcal{D}$ are the closest points on $C \bigcap \mathbb{R}^2$ from q, because $\mathcal{D}$ is centred on q. In Lemma 4.1.6, we

have seen that the set of all closest points on $C \bigcap \mathbb{R}^2$ from q is contained in $\mathcal{M}$. Thus, the minimum power of the points of $C \bigcap \mathbb{R}^2$ with respect to $\mathcal{D}$ is the power of a point of $\mathcal{M}$ with respect to $\mathcal{D}$. If C is not a circle centred on q, $\mathcal{M} = \mathfrak{S}$, and we are done. If C is a circle centred on q, all the points of $\mathcal{M}$ are at the same distance from q: the radius of C. Thus, the minimum power of the points of $C \bigcap \mathbb{R}^2$ with respect to $\mathcal{D}$ is the power of the point w of $\mathfrak{S}$. □

A direct consequence of Remark 4.1.2 is that the power of any point of $C \bigcap \mathbb{R}^2$ with respect to any circle $\mathcal{D}$ centred on a point q of the true R-offset and of radius R is positive or zero. Lemma 4.1.7 allows us to restate this last condition as the power of any of the points of the sets $\mathfrak{S}$ with respect to any of the circles $\mathcal{D}$ must be positive or zero.

By Definition 4.1.3, the point q on the generalised R-offset curve $\mathcal{O}$ can be constructed from a non-singular point $p = (x, y)$ on C as the intersection of the normal to C at p and the circle centred on p, and of radius R (see Figure 4.1.4). This circle is the variety $V(d)$, where

$$d(x, y, u, v) = (u - x)^2 + (v - y)^2 - R^2,$$

whereas the normal is the variety $V(n)$, where

$$n(x, y, u, v) = -f_y \cdot (u - x) + f_x \cdot (v - y).$$

We are ready to write the equations of the offset.

Let us consider the map $\pi : K^6 \to K^2$ defined by $\pi((x, y, u, v, \alpha, \beta)) = (u, v)$.

The generalised R-offset $\mathcal{O}$ is the Zariski closure of the image by π of the variety of K^6 defined by the following system of equations and inequalities:

$$\begin{cases} f(x, y) = n(x, y, u, v) = d(x, y, u, v) = 0 \\ f_x(x, y) \neq 0 \text{ or } f_y(x, y) \neq 0 \end{cases}.$$

The first line in the preceding system of equations and inequalities contains the algebraic equations defining the point p on C and the point q on the generalised

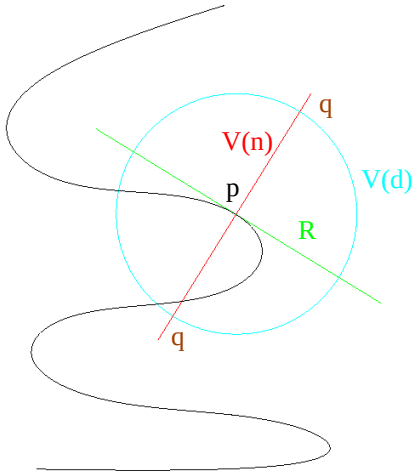

Figure 4.1.4: The construction of a point of the generalised offset

offset to C. The second line is a necessary and sufficient condition for p not being singular.

The true R-offset is obtained as the difference of the generalised R-offset $\mathcal{O}$ and the union of each one of the images by π of the sets defined by the following system of equations and inequalities for each point $m = (\alpha, \beta)$ of $\mathfrak{S}$:

$$\begin{cases} f(\alpha,\beta) = n(\alpha,\beta,u,v) = 0 \\ (u-\alpha)^2 + (v-\beta)^2 - R^2 < 0 \end{cases}.$$

We will now ennounce some fundamental properties of the true offset that are central to the design of the Delaunay graph conflict locator.

Proposition 4.1.8. *The true offset to an algebraic curve is a semi-algebraic set.*

Proof. The true R-offset is defined as the difference of two sets. The second set is a finite union of sets (since $\mathfrak{S}$ is finite), each one of them being the projection of a set defined by a system of equations and inequalities. Such a set is a semi-algebraic set. The projection of a semi-algebraic set is semi-algebraic [BCR98, Thm.2.2.1]. A finite union or difference of semi-algebraic sets is semi-algebraic (see Note following

Definition 2.1.4 in [BCR98]). Thus, the second set is a semi-algebraic set. The first set is also semi-algebraic, as the image of an algebraic set by a projection. □

Proposition 4.1.9. *The true offset to an algebraic curve is not necessarily algebraic.*

Proof. We have proved that the true offset is a difference of two sets, where the first one is the projection of an algebraic set, and the second one is a semi-algebraic set. If the second set is different from the empty set, then the true offset cannot be algebraic. Let's consider the strophoid, which is the affine algebraic variety $V\left(y^2 - x^2 - x^3\right)$ (see Figure 4.1.2). The true offset curve to a strophoid differs from its generalised offset because around the origin of the coordinate system, the generalised offset has two branches which intersect the strophoid. Indeed, the intersection points of the strophoid and its generalised offset cannot be part of a true offset (with positive offset value) to the strophoid, because their distance to the strophoid is zero. Thus, any positive true offset to the strophoid is not algebraic. □

The fact the true offset is not an algebraic set implies that the true Voronoi vertices (as intersections of 3 true offsets) are not algebraic, and thus, they cannot be defined by a system of algebraic equations. The fact the true offset is a semi-algebraic set implies that the true Voronoi vertices are semi-algebraic sets (as finite intersection of semi-algebraic sets). The algebraic variety closest to the true Voronoi vertex in the plane is the intersections of three generalised offsets, that we will call a generalised Voronoi vertex. This notion will be formally defined in Chapter 5, and it will be used as a central tool for the Delaunay graph conflict locator for conics.

4.2 The degree of the generalised offset curve

In this section, we will prove a general formula for the degree of the generalised offset to an algebraic curve. Let us start with some notations. Consider polynomials in $K[x, y, u, v]$ and the projective space $\mathbb{P}^4$, in which K^4 is embedded. The

homogenisation variable will be denoted as t. We consider the point $q = (u, v)$ on the generalised offset curve $\mathcal{O}$ constructed from an arbitrary point $p = (x, y)$ on $C = V(f) \subset K^2$. In this section, the varieties will be considered in different underlying spaces. We will use the notation $V(f)$ extensively hereafter. When necessary, we will precise the underlying space by an inclusion, e.g. $V(f) \subset K^4$ or $V(f) \subset K^2$.

Now let us define the different affine varieties that allow one to define the generalised offset. We will consider these affine varieties in K^4 since the polynomials involved belong to $K[x, y, u, v]$. Let $B = V(f) \subset K^4$ be the image of C by the inclusion map from K^2 to K^4 defined by $(x, y) \mapsto (x, y, u, v)$. Let $\mathcal{N} = V(n)$, $D = V(d)$ and $V(f_x, f_y)$ be considered in K^4 in the same way. Now, $V(n)$ is a proper subset of K^4 provided that p is not singular. If f is not *square-free* (i.e. f admits a factorisation with square factors), there is an infinity of singular points. The generalised offset curve $\mathcal{O}$ is the image of $(V(f) \cap V(n) \cap V(d)) \setminus V(f_x, f_y)$ by the canonical projection $\pi : K^4 \to K^2$ onto the (u, v)-plane.

We will determine the degree of the generalised offset considered in K^4. Recall that the definition of degree of a projective variety we gave at the beginning of Section 4.1 depends on the dimension of the projective variety. Thus, we need to determine the dimension of the generalised offset. For this purpose, we will consider the *projective completion* (i.e. the smallest projective variety containing it) of $V(f, n, d) \subset K^4$. The motivation for the consideration of the projective completion instead of the affine variety lies in the conditions of the theorem (Theorem 4.2.4) we will use in order to determine the dimension of the generalised offset. Then, we will decompose this projective completion as the union of its *component at infinity* (i.e. the points with homogeneous coordinate equal to zero), its *singular component* (points induced by singular points p on C), and the generalised offset considered in K^4 (i.e. the affine variety $V(f, n, d) \setminus V(f_x, f_y) \subset K^4$). We will thus obtain the generalised offset considered in K^4 as a difference of projective varieties, and we will determine their dimensions and degrees in order to determine the degree

of the generalised offset. Now, we need some notations related to homogenisation, projective completion and component at infinity.

Notation 4.2.1. Let

- $\overline{f}$ denote the *homogenisation* (i.e. the replacement of each monomial m in f by $mt^{\deg(f)-\deg(m)}$ where t is the homogenisation variable and deg denotes the degree) of the polynomial f,
- $\overline{V}$ denote the *projective completion* of the variety V,
- f^T denote the polynomial defining the *component at infinity* of $\overline{V(f)}$.

The homogenisation of a polynomial f defines the projective completion of the variety $V(f)$: $\overline{V(f)} = V\left(\overline{f}\right)$; see [CLO97, Sec.8.4]. Clearly, $V\left(\overline{f}\right), V(\overline{n})$ and $V\left(\overline{d}\right)$ are all subsets of $\mathbb{P}^4$. Let $W := \overline{V(f)} \cap \overline{V(n)} \cap \overline{V(d)}$. Thus, $W = V\left(\overline{f}, \overline{n}, \overline{d}\right) \subset \mathbb{P}^4$. Let W_a be the subset of W defined as $W \setminus \left(\overline{V(f_x, f_y)} \cup V(t)\right) \subset \mathbb{P}^4$. W_a is the generalised offset considered in K^4. W_a is a quasi-projective variety since $\overline{V(f)} \cap \overline{V(n)} \cap \overline{V(d)}$ and $\overline{V(f_x, f_y)} \cup V(t) \subset \mathbb{P}^4$ are projective varieties. Let us define the projective variety $W_S := W \cap \overline{V(f_x, f_y)}$ and the projective variety $W_\infty := W \cap V(t)$. Thus, the generalised offset considered in K^4 is the following affine variety: $W_a = W \setminus (W_\infty \bigcup W_S)$.

We will now determine the dimension of the component at infinity W_∞ of W. Since $W = V\left(\overline{f}, \overline{n}, \overline{d}\right) \subset \mathbb{P}^4$ and $W_\infty := W \cap V(t)$, $W_\infty = V\left(f^T, n^T, d^T, t\right)$. Thus, in order to determine the dimension of W_∞, we will examine how the dimension is changed when going from $V\left(f^T\right)$ to $V\left(f^T, d^T\right)$, and from $V\left(f^T, d^T\right)$ to $V\left(f^T, d^T, t\right)$, and finally from $V\left(f^T, d^T, t\right)$ to $V\left(f^T, d^T, t, n^T\right) = W_\infty$. For this purpose we use the following theorem on the dimension of an hypersurface. We will now introduce regular functions, hypersurfaces and irreducible closed sets, in order to recall the theorem on the dimension of a hypersurface. Hypersurfaces will also be

encountered in the proofs of Lemma 4.2.15 on the dimension of W_∞ and of Theorem 4.2.18 on the degree of the generalised offset.

Definition 4.2.2. (Regular function on an affine closed set) Let X be a closed set in the affine space K^N. A function f given on X is called *regular* if there exists a polynomial F with coefficients in k such that $f(x) = F(x)$ for all points $x \in X$ [Sha94].

If X is closed in $\mathbb{P}^N$ and $F \neq 0$ is a complex valued regular function on X, then we denote by X_F the closed subset of X, known as a *projective hypersurface*, defined by $F = 0$.

Definition 4.2.3. (Irreducible closed set [Sha94]) A closed set X is called *reducible* if there exist closed subsets $X_1 \subset X$, $X_2 \subset X$, $X_1 \neq X$, $X_2 \neq X$, such that $X = X_1 \cup X_2$. Otherwise, X is called *irreducible*.

Thus, a closed set $X = V(f)$ is irreducible if, and only if, f cannot be written as $f = h_1 h_2$ with $h_1, h_2 \in K[x, y, u, v]$ and h_1 and h_2 are not constant polynomials. In such case, we call the polynomial f irreducible.

The following Theorem on the dimension of a hypersurface can be found as [Sha94, Thm.4,Ch.1,Sec.6] or [CLO97, Cor.4,p.459].

Theorem 4.2.4. *(Theorem on the dimension of a hypersurface) If a complex valued regular function F does not vanish on an irreducible projective variety X, then* $\dim X_F = \dim X - 1$.

In order to apply the theorem on the dimension of an hypersurface on a projective variety $X = V(f)$ whose irreducibility is not known, we determine if none of the irreducible components of X is contained in an irreducible component of $V(F)$. Indeed, if an irreducible component $V(h_i)$ is contained in an irreducible component $V(F_i)$ of $V(F)$, then F_i and F vanish on $V(h_i)$, and the dimension of $V(h_i) \bigcap X_F$ is the same as the dimension of $V(h_i)$. In Lemma 4.2.6, we determine that none of

the irreducible components of $V\left(f^T\right) \subset \mathbb{P}^4$ is contained in an irreducible component of $V\left(d^T\right) \subset \mathbb{P}^4$. In Lemma 4.2.10, we determine that none of the irreducible components of $V\left(f^T, d^T\right) \subset \mathbb{P}^4$ is contained in an irreducible component of $V(t) \subset \mathbb{P}^4$. Finally in Lemma 4.2.11, we determine when none of the irreducible components of $V\left(f^T, d^T, t\right) \subset \mathbb{P}^4$ is contained in an irreducible component of $V\left(n^T\right) \subset \mathbb{P}^4$. Lemma 4.2.15 will conclude the determination of the degree of the one-dimensional component of W_∞.

Here are the notations and definitions needed to ennounce and prove Lemma 4.2.6.

Let $V\left(f^T\right) = V(h_1) \cup V(h_2) \cup \cdots \cup V(h_s)$ be a minimal (i.e. such as $V(h_i) \not\subset V(h_j)$ for any $j \neq i$) decomposition of $V\left(f^T\right)$ into irreducible closed sets. Such a minimal decomposition of $V\left(d^T\right)$ into irreducible closed sets contains at most two components since the degree of $d^T = (u-x)^2 + (v-y)^2$ is 2. $V\left(d^T\right) = V(d_1) \cup V(d_2)$, where $d_1 := -\iota(u-x) + (v-y)$, $d_2 := \iota(u-x) + (v-y)$ and ι is a root of the equation $x^2 + 1 = 0$ in the algebraically closed field K, is a minimal decomposition of $V\left(d^T\right)$ into irreducible closed sets.

Notation 4.2.5. For $f_1, \ldots, f_s \in K[x_1, ..., x_N]$, let $\langle f_1, ..., f_s \rangle$ denote the *ideal* generated by $f_1, ..., f_s$ in $K[x_1, ..., x_N]$. Let $\sqrt{\ }$ denote the *radical* of the ideal [Sha94, CLO97]. The radical $\sqrt{I}$ of an ideal I of a ring A is $\sqrt{I} = \left\{a \in A | \exists N \in \mathbb{N} : a^N \in I\right\}$.

We can now state and prove Lemma 4.2.6.

Lemma 4.2.6. *None of the irreducible components of* $V\left(f^T\right) \subset \mathbb{P}^4$ *is contained in an irreducible component of* $V\left(d^T\right) \subset \mathbb{P}^4$.

Proof. We will prove it by contradiction. Let us assume that $V(h_i) \subset V\left(d^T\right)$, i.e., $d^T \in I(V(h_i)) \subset K[x, y, u, v, t]$ for some $i \in \{1, 2, ..., s\}$. By Hilbert's Nullstellensatz [Sha94], $I(V(h_i)) = \sqrt{\langle h_i \rangle}$. Since h_i is irreducible, $\sqrt{\langle h_i \rangle} = \langle h_i \rangle$. Thus, there exists $g \in K[x, y, u, v, t]$ such that $d^T = g \cdot h_i$. The sum of the degrees of g

and of h_i equals 2. Now, either the degree of g is 0 and the degree of h_i is 2, or the degrees of g and of h_i are both 1, or the degree of g is 2 and the degree of h_i is 0. The first case is not possible, because if the degree of h_i is 2, then h_i is reducible. In the second case, $d^T = g \cdot (\alpha x + \beta y + \gamma)$, where α, β and γ are coefficients of K. This would imply that the terms in u^2 and the terms in v^2 of d^T must come from g. These terms induce terms in u^2x, u^2y, v^2x, and v^2y which cannot be cancelled. In the last case, h_i reduces to a constant, which is impossible since $V(h_i) \neq \emptyset$. □

In order to present Lemmas 4.2.10 and 4.2.11, we need to recall some definitions and results about regular functions and regular mappings on projective varieties.

Definition 4.2.7. (Regular mapping of affine closed sets) Let X be a closed set of K^N and Y be a closed set of K^N. A mapping $f : X \to Y$ is called *regular* if there exists N regular functions $f_1, ..., f_N$ on X such that $f(x) = (f_1(x), .., f_N(x))$ for all $x \in X$ [Sha94].

Let $\mathbb{P}^N$ denote the $N-$dimensional projective space, so that a point $\xi \in \mathbb{P}^N$ is given by $N+1$ elements $(\xi_0 : ... : \xi_N)$ of K and not all the ξ_i are 0 [Sha94]. Let $\mathbb{A}_i^N$ be the subset of $\mathbb{P}^N$ consisting of all the points for which $\xi_i \neq 0$.

Definition 4.2.8. (Regular mapping of quasi-projective varieties) Let $f : X \to Y$ be a *mapping of quasi projective varieties* and $Y \subset \mathbb{P}^N$. This mapping is called *regular* if for every point $x \in X$ and every open affine set $\mathbb{A}_i^N$ containing the point $f(x)$ there exists a neighbourhood U of x such that $f(U) \subset \mathbb{A}_i^N$, and the mapping $f : U \to \mathbb{A}_i^N$ is regular [Sha94].

Theorem 4.2.9. *([Sha94, Thm.8, Sect.1.6]) Let $f : X \to Y$ be a regular map between projective varieties with $f(X) = Y$. Suppose that Y is irreducible and that all the fibres $f^{-1}(y)$ for $y \in Y$ are irreducible and of the same dimension, then X is irreducible.*

We are now ready to state and prove Lemmas 4.2.10 and 4.2.11.

Lemma 4.2.10. *None of the irreducible components of* $V\left(f^T, d^T\right) \subset \mathbb{P}^4$ *is contained in an irreducible component of* $V(t) \subset \mathbb{P}^4$.

Proof. Notice that $V\left(f^T\right)$ and $V(h_i)$ are not contained in the hyperplane at infinity $V(t)$ and that the irreducible components of $V\left(f^T, d^T\right)$ are the $V(h_i, d_j)$ for all $i = 1, \ldots, s$, $j = 1, 2$. Consider the following sequence of projections:

$$\pi_{ij} : \begin{array}{ccc} V(h_i, d_j) \subset \mathbb{P}^4 & \rightarrow & V(h_i) \subset \mathbb{P}^2 \\ (t : x : y : u : v) & \mapsto & (t : x : y) \end{array}.$$

Clearly, these π_{ij} are regular mappings of projective varieties. Clearly also, each fibre of these mappings is irreducible. The fibres $\pi_{ij}^{-1}(\omega)$ for $\omega \in V(h_i)$ have dimension 1 since the points on these fibres have fixed x, y and t coordinates (those of ω), and their other coordinates u and v are related by the equation of d_j. Thus, all the fibres have the same dimension. Then, we can apply Theorem 4.2.9, and conclude that the $V(h_i, d_j)$ are irreducible.

We will show that none of these $V(h_i, d_j)$ is contained in an irreducible component of $V(t)$. Let's suppose $t \in I(V(h_i, d_j))$ for some $i \in \{1, 2, ..., s\}$ and $j \in \{1, 2\}$. By Hilbert's Nullstellensatz [Sha94], $I(V(h_i, d_j)) = \sqrt{\langle h_i, d_j \rangle}$. Since the $V(h_i, d_j)$ are irreducible, $\sqrt{\langle h_i, d_j \rangle} = \langle h_i, d_j \rangle$. Then there exists $a, b \in K[x, y, u, v, t]$ such that $t = ah_i + bd_j$. Since h_i and d_j don't have monomials with t nor constant terms, t in $ah_i + bd_j$ must come from a or b or both. Since h_i and d_j don't have constant terms, the monomial of least total degree containing t in $ah_i + bd_j$ must have a degree greater than or equal to 1 in the other variables. This is a contradiction. □

Lemma 4.2.11. *None of the irreducible components of* $V\left(f^T, d^T, t\right) \subset \mathbb{P}^4$ *is contained in an irreducible component of* $V\left(n^T\right) \subset \mathbb{P}^4$ *if, and only if,* $\left(f_x^T\right) + \iota\left(f_y^T\right) \notin \langle h_i \rangle$ *and* $\left(f_x^T\right) - \iota\left(f_y^T\right) \notin \langle h_i \rangle$ *for* $i = 1, 2, ..., s$.

Proof. Now, consider the following sequence of mappings:

$$\pi_{ij} : \begin{array}{ccc} V(h_i, d_j, t) \subset \mathbb{P}^4 & \rightarrow & V(h_i) \subset \mathbb{P}^2 \\ (t : x : y : u : v) & \mapsto & (t : x : y) \end{array}.$$

Notice that $V\left(f^T, d^T, t\right)$ is contained in the hyperplane at infinity $V(t)$, and thus does $V\left(h_i, d_j, t\right)$. Clearly, these mappings are regular mappings of projective varieties and $\pi_{ij}\left(V\left(h_i, d_j, t\right)\right) = V\left(h_i, t\right)$. Clearly also, each fibre of these mappings is irreducible, and $V\left(h_i, t\right)$ is also irreducible. The fibres $\pi_{ij}^{-1}\left(\omega\right)$ for $\omega \in V\left(h_i, t\right)$ have dimension 1 since the points on these fibres have fixed x, y coordinates (those of ω), their t coordinate equals 0, and their other coordinate v is related to u by the equation of d_j, and is therefore also fixed. Thus, all the fibres have the same dimension. Then, we can apply Theorem 4.2.9, and conclude that the $V\left(h_i, d_j, t\right)$ are irreducible.

We will prove the contrapositive of the conditional statements. Let's suppose $\left(f_x^T\right) + \iota\left(f_y^T\right) \in \langle h_i \rangle$ for some $i \in \{1, 2, ..., s\}$. We have to show that $n^T = 0$ on $V\left(h_i, d_1, t\right)$. Since $\left(f_x^T\right) + \iota\left(f_y^T\right) \in \langle h_i \rangle$, there exists $g \in K\left[x, y\right]$ such as $\left(f_x^T\right) + \iota\left(f_y^T\right) = g \cdot h_i$. But, $h_i = 0$ on $V\left(h_i, d_1, t\right)$. Thus, $\left(f_x^T\right) + \iota\left(f_y^T\right) = 0$, i.e., $f_y^T = \iota f_x^T$. Then, replacing in n^T, we get $n^T = -\iota f_x^T\left(u - x\right) + f_x^T\left(v - y\right) = \left(-\iota\left(u - x\right) + \left(v - y\right)\right) f_x^T$ on $V\left(h_i, d_1, t\right)$. Since $d_1 = 0$ on $V\left(h_i, d_1, t\right)$, $-\iota\left(u - x\right) + \left(v - y\right) = 0$, and therefore, $n^T = 0$ on $V\left(h_i, d_1, t\right)$. We can prove in the same way that if $\left(f_x^T\right) - \iota\left(f_y^T\right) \in \langle h_i \rangle$ then $n^T = 0$ on $V\left(h_i, d_2\right)$.

Reciprocally, let us suppose $n^T = 0$ on $V\left(h_i, d_1, t\right)$ for some $i \in \{1, 2, \ldots, s\}$. Since none of n^T, h_i, d_1 depend on t and $n^T = 0$ on $V\left(h_i, d_1, t\right)$, $n^T = 0$ on $V\left(h_i, d_1\right)$. Since $d_1 = 0$ on $V\left(h_i, d_1\right)$, then $-\iota\left(u - x\right) + \left(v - y\right) = 0$, and

$$\begin{cases} n^T = -f_y^T\left(u - x\right) + \iota f_x^T\left(u - x\right) = 0 \\ n^T = \iota f_y^T\left(v - y\right) + f_x^T\left(v - y\right) = 0 \end{cases}$$

on $V\left(h_i, d_1\right)$. Since $\left(-f_y^T + \iota f_x^T\right) = \iota\left(\iota f_y^T + f_x^T\right)$, we can rewrite the last system of equations as

$$\begin{cases} \left(-f_y^T + \iota f_x^T\right)\left(u - x\right) = 0 \\ \left(-f_y^T + \iota f_x^T\right)\left(v - y\right) = 0 \end{cases}$$

on $V\left(h_i, d_1\right)$.

The subvariety $V(h_i, d_1, u-x, v-y)$ of $V(h_i, d_1)$ is a one-dimensional variety in the two-dimensional variety $V(h_i, d_1)$. The open set $V(h_i, d_1) \backslash V(h_i, d_1, u-x, v-y)$ is a dense open subset of $V(h_i, d_1)$. From the last system, we know that $\left(-f_y^T + \iota f_x^T\right)$ vanishes on this dense open subset. Now, we consider $V\left(h_i, d_1, -f_y^T + \iota f_x^T\right)$. This is a closed projective set. We know that it contains the open set $V(h_i, d_1) \setminus V(h_i, d_1, u-x, v-y)$. Thus, it contains also its Zariski closure, i.e., $V(h_i, d_1)$. Thus, $\left(-f_y^T + \iota f_x^T\right)$ vanishes on $V(h_i, d_1)$. Therefore, there exists $a, b \in K[x, y, u, v]$ such that $-f_y^T + \iota f_x^T = ah_i + bd_1$, i.e., $-f_y^T + \iota f_x^T = ah_i + b(-\iota(u-x) + v - y)$.

Let a_{uv} be the sum of all the terms of a containing the variables u or v. Since $\left(-f_y^T + \iota f_x^T\right)$ does not depend on any of the variables u and v, $a_{uv}h_i + b(-\iota u + v) = 0$. Thus, $a_{uv}h_i = b(\iota u - v)$. Since the left hand side of this equality has terms in x or y, b must also have terms in x or y. The last two facts imply that there exists a polynomial $e \in K[x, y, u, v]$ such that $b = eh_i$ and $a_{uv} = e(\iota u - v)$. Since $b = eh_i$, $b \in \langle h_i \rangle$. Thus, $-f_y^T + \iota f_x^T \in \langle h_i \rangle$. Thus, $f_x^T + \iota f_y^T = -\iota\left(-f_y^T + \iota f_x^T\right) \in \langle h_i \rangle$. In the same way, we can prove that if $n^T = 0$ on $V(h_i, d_2, t) \subset \mathbb{P}^4$ then $f_x^T - \iota f_y^T \in \langle h_i \rangle$. □

We are now going to analyse the dimension of W_∞.

Notation 4.2.12. Let $l = 1$ if $f_x^T + \iota f_y^T \in \langle h_i \rangle$ or $f_x^T - \iota f_y^T \in \langle h_i \rangle$ for some $i \in \{1, 2, ..., s\}$, and $l = 0$ otherwise.

Lemma 4.2.13. *The dimension of W_∞ is equal to l.*

Proof. We can see W_∞ in the following two equivalent ways $W_\infty = \overline{B} \cap \overline{\mathcal{N}} \cap \overline{D} \cap V(t) \subset \mathbb{P}^4$, or $W_\infty = V\left(f^T, n^T, d^T, t\right) \subset \mathbb{P}^4$. Considering the last expression, by Lemmas 4.2.6, 4.2.10 and three repeated applications of Theorem 4.2.4, we get that $V\left(f^T, d^T, t\right) \subset \mathbb{P}^4$ has dimension $4 - 3 = 1$. Thus, the dimension of W_∞ is 0 or 1. By Theorem 4.2.4, the dimension of W_∞ is 0 if, and only if, $l = 0$ by Lemma 4.2.11. □

The following Lemmas 4.2.15 and 4.2.16 give the degrees of the one-dimensional component at infinity and of the one dimensional singular component of the projective variety W. They will allow us to conclude with the degree of the generalised offset in Theorem 4.2.18.

We recall the definition of *localisation* (see [GP02, Definition 1.4.4]) at a *prime ideal* (see [GP02, Definition 1.3.10]) $\mathfrak{P} \subset K[t_1, ..., t_m]$, where K is a field. We denote by $K[x_1, ..., x_m]_{\mathfrak{P}}$ the set of all *rational functions* f/g such that $f, g \in K[x_1, ..., x_m]$ where $g \notin \mathfrak{P}$.

Definition 4.2.14. (Intersection multiplicity, adapted from [GP02, Def.A.8.16, p.480]) Let $f, g \in K[x, y]$, $p = (p_1, p_2) \in V(f) \bigcap V(g) \subset K^2$, and let $\mathcal{M}_p = \langle x - p_1, y - p_2 \rangle$ be the maximal ideal of p. We define $\mu_p(f, g) := \dim_k \left(K[x, y]_{\mathcal{M}_p} / \langle f, g \rangle \right)$, and call it *the intersection multiplicity* of f and g at p.

Now let $F, G \in K[z, x, y]$ be homogeneous polynomials, let $p = (p_0 : p_1 : p_2) \in V(F) \bigcap V(G) \subset \mathbb{P}^2$, and let $\mathcal{M}_p = \langle p_0 x - p_1 z, p_0 y - p_2 z \rangle$ be the homogeneous ideal of p. Assume that $p_0 \neq 0$ then, for the intersection multiplicity $\mu_p(F, G) := \dim_k \left(K[z, x, y]_{\mathcal{M}_p} / \langle F, G \rangle \right)$, it is easy to see that it equals $\mu_p(f, g)$, where $f = F|_{z=1}$ and $g = G|_{z=1}$.

If C, D are projective curves such as $I(C) = \langle F \rangle$, and $I(D) = \langle G \rangle$, then $\mu_p(C, D) := \mu_p(F, G) = \mu_p(V(F), V(G))$ is the intersection multiplicity of C, D at p.

Let C_∞ be the component at infinity of the projective completion of C, i.e. $C = V\left(f^T, t\right) \subset K^2$.

Lemma 4.2.15. *The degree of the one-dimensional component of W_∞ is:*

$$2l \sum_{p \in C_\infty} \mu_{p'} \left(V\left(f^T\right), V\left(n^T\right) \right),$$

where p' is an arbitrary point of W_∞ whose projection on the projective (t, x, y)-plane (or equivalently on C_∞) is p.

Proof. Lemma 4.2.13 implies that W_∞ has a one-dimensional component if, and only if, $l = 1$. In this case, there exists an irreducible component $V(h_i)$ such that $n^T = 0$ on $V(h_i, d_j, t)$. Thus, the points of W_∞ are the same as the ones of $V\left(\sqrt{\langle d^T, f^T, t\rangle}\right)$, but their multiplicities differ.

The one-dimensional component of $V\left(\sqrt{\langle d^T, f^T, t\rangle}\right)$ is constituted of all the points p' whose projection on the projective (t, x, y)-plane is a point p of $C_\infty \subset \mathbb{P}^2$ and whose projection to the affine (u, v)-plane is the circle $V\left(d^T\right)$. There are also the two isolated cyclic points $(0 : 0 : 0 : 1 : \pm\iota)$ in W_∞, but they do not belong to the one-dimensional component of W_∞.

The degree of the one-dimensional component of $V\left(\sqrt{\langle d^T, f^T, t\rangle}\right)$ equals the product of the degree of d^T (which is 2) by the number of isolated points in C_∞. The degree of the one-dimensional component of W_∞ is twice the sum of the multiplicities of $V\left(f^T, n^T\right) \subset \mathbb{P}^4$ at the points p' for all the points p of C_∞. □

Let C_S be the affine subvariety of C composed of all its singular points, i.e. $C_S = V(f_x, f_y, f) \subset K^2$.

Lemma 4.2.16. *The degree of the one-dimensional component of W_S is:*

$$2 \sum_{q \in C_S} \mu_{q'}\left(V\left(\overline{f}\right), V\left(\overline{n}\right)\right),$$

where q' is an arbitrary point of W_S whose projection on the affine (x, y)-plane (or equivalently on C_S) is q.

Proof. Each point q of C_S induces a trivial equation n. Thus, at the level of W_S, it induces a one-dimensional variety that consists of all the points q' whose projection on the affine (x, y)-plane is q and whose projection on the projective (t, u, v)-plane is a projective circle centred at q and of radius R. It follows that W_S does not have a one-dimensional component at infinity. The only component at infinity of W_S are the points $(0 : x_0 : y_0 : 1 : y_0 \pm \iota(1 - x_0))$, where (x_0, y_0) is a common root of f^T, $f_x^T(x, y)$, and $f_y^T(x, y)$).

The points of W_S are the same as the points of $V\left(\sqrt{\langle \overline{f_x}, \overline{f_y}, \overline{f}, \overline{d}\rangle}\right)$, but their multiplicities differ. The degree of the one-dimensional component of $V\left(\sqrt{\langle \overline{f_x}, \overline{f_y}, \overline{f}, \overline{d}\rangle}\right)$ equals the product of the degree of $\overline{d}$ (which is 2) by the number of isolated points in C_S. The degree of the one-dimensional component of W_S is twice the sum of the multiplicities of $V\left(\overline{f}, \overline{n}\right) \subset \mathbb{P}^4$ at the points q' for all the points q of C_S. □

Remark 4.2.17. We could think that since the only variables in common to $\overline{f}$ and $\overline{n}$ are x and y, and these are the only variables of $\overline{f}$, the intersection multiplicity of $V\left(\overline{f}\right) \subset \mathbb{P}^4$ and $V\left(\overline{n}\right) \subset \mathbb{P}^4$ at q' should equal the intersection multiplicity of $V\left(\overline{f}\right) \subset K^2$ and $V\left(\overline{n}\right) \subset K^2$ at q. In fact this is not necessarily true. Indeed, by the projection from the four-dimensional projective space to a projective plane, the intersection multiplicity may be lowered.

An example is $\overline{f} = x^2y - y^3$. The intersection multiplicity of $V\left(\overline{f}\right) \subset \mathbb{P}^4$ and $V\left(\overline{n}\right) \subset \mathbb{P}^4$ at $q' = (1:0:0:0:0)$ is 9 while the intersection multiplicity of $V\left(\overline{f}\right) \subset K^2$ and $V\left(\overline{n}\right) \subset K^2$ at $q = (0,0)$ is 6. The degree of W_S is 18, which corresponds to twice the intersection multiplicity of $V\left(\overline{f}\right) \subset \mathbb{P}^4$ and $V\left(\overline{n}\right) \subset \mathbb{P}^4$ at q'.

The point q' was taken on the 0−generalised offset to the curve defined by $f = x^2y - y^3$: it satisfies $\overline{d} = (u-x)^2 + (v-y)^2 - R^2t^2 = 0$ with $R = 0$. However, when $R = 0$, W_S is 0-dimensional, but its degree corresponds to that announced by Lemma 4.2.16. In this case, W_∞ is 0-dimensional (the three points $(0:1:0:1:0)$, $(0:1:1:1:1)$ and $(0:1:-1:1:-1)$), and $l = 0$. Finally, the degree of W is 18.

However, in this case the degree of the canonical projection $\pi : \mathbb{P}^4 \backslash \{(1:0:0:0:0)\} \rightarrow K^2$ is not any more 1, but 6. Indeed, any point (x, y) of the generalised offset is the image of possibly 3 double points $(t:x:y:u:v)$ of W_a by π. This justifies that the degree of the 0−generalised offset is 3.

Theorem 4.2.18. *The degree of the generalised offset to an algebraic curve* $V(f) \subset$

K^2 of degree m, such as f is square-free is :

$$2m^2 - 2l \sum_{p \in C_\infty} \mu_{p'} \left(V\left(f^T\right), V\left(n^T\right)\right) - 2 \sum_{q \in C_S} \mu_{q'} \left(V\left(\overline{f}\right), V\left(\overline{n}\right)\right),$$

where p' and q' are defined as in Lemmas 4.2.15 and 4.2.16.

Proof. Bézout's Theorem [Sha94] tells us that for projective varieties, the degree of the intersection of two such varieties is generically equal to the product of the degrees of these varieties. Therefore, the degree of $W = \overline{B} \cap \overline{\mathcal{N}} \cap \overline{D}$ is generically $2m^2$. The quasi-projective varieties W, W_a, W_∞, and W_S are related by $W_a = W \backslash (W_\infty \cup W_S)$. Since W is defined by three polynomials, its dimension is at least 1 according to Theorem 4.2.4. Since n is a polynomial in two more variables than f, n cannot identically vanish on $\overline{B}$. Since d is a polynomial whose coefficients are polynomials in R, $R \neq 0$ and the coefficients of f and d do not depend on R, d cannot identically vanish on $\overline{B} \cap \overline{\mathcal{N}}$, and the dimension of W is exactly one. Since W is one-dimensional, the degree of W is the sum of the degrees of its one-dimensional components. Thus, the degree of W_a equals the degree of W minus the degree of the one-dimensional component of $W_\infty \cup W_S$.

Since W_S has no one-dimensional component at infinity (see proof of Lemma 4.2.16), the one-dimensional components of W_∞ and of W_S are disjoint, and the degree of the one-dimensional component of $W_\infty \cup W_S$ is the sum of the degrees of the one-dimensional components of W_∞ and of W_S. By Lemmas 4.2.15 and 4.2.16, the degree of W_a is

$$2m^2 - 2l \sum_{p \in C_\infty} \mu_{p'} \left(V\left(f^T\right), V\left(n^T\right)\right) - 2 \sum_{q \in C_S} \mu_{q'} \left(V\left(\overline{f}\right), V\left(\overline{n}\right)\right). \tag{4.2.1}$$

By definition, t never vanishes on W_a. Finally, the generalised offset $\mathcal{O}$ is the image of W_a by the canonical projection $\pi : \mathbb{P}^4 \backslash \{(1:0:0:0:0)\} \to K^2 \ : \ (t:x:y:u:v) \mapsto \left(\frac{u}{t}, \frac{v}{t}\right)$ with degree 1. Thus, the degree of $\mathcal{O}$ is (4.2.1). □

The multiplicity of the extraneous variety corresponding to a singular point is at least the product of the valuations of the polynomials f and n at a generic point

of W_S. The valuation of f corresponds to the multiplicity m_p of the singular point of C. The valuation of n is the valuation of f minus 1 since n involves the partial derivatives of f. Thus, the multiplicity of the extraneous variety corresponding to a singular point is at least $m_p\,(m_p-1)$, and the degree of the corresponding extraneous factor is at least $2m_p\,(m_p-1)$.

4.3 The degree of the generalised offset to a conic

In this section, we use the general formula for the degree of the generalised offset to a conic developed at the preceding section in order to compute the degree of the generalised offset to the different conics. In this section, we deal with real affine curves.

Proposition 4.3.1. *The degree of the generalised offset to a circle is 4.*

Proof. The projective completion of the circle is the projective variety $V\left((x-at)^2+(y-bt)^2-r^2t^2\right)$. Replacing t by 0 in the previous polynomial, we get $f^T=x^2+y^2=(x+\iota y)\,(x-\iota y)=h_1h_2$ By taking the partial derivatives, we get $f_x^T=2x$, $f_y^T=2y$. Thus, $f_x^T+\iota f_y^T=2\,(x+\iota y)\in\langle h_1\rangle$. Thus, the dimension of the component at infinity of the projective completion of the generalised offset is 1. Since the component at infinity of a circle is the two cyclic points $(0:1:\pm\iota)$, and $V\left(f^T\right)$ and $V\left(n^T\right)$ do not meet tangentially, $\sum_{p\in C_\infty}\mu_{p'}\left(V\left(f^T\right),V\left(n^T\right)\right)=2$, and the degree of W_∞ is 4. Since a circle does not have singular points, $C_S=\emptyset$. Thus, the degree of the generalised offset to the circle is $2\cdot 2^2-4-0=4$. □

Proposition 4.3.2. *The degree of the generalised offset to an ellipse or a hyperbola is* 8.

Proof. The ellipse can be considered in some coordinate system as the affine variety $V\left(\frac{x^2}{a^2}+\frac{y^2}{b^2}-1\right)$. The projective completion is the projective variety $V\left(b^2x^2+a^2y^2-a^2b^2t^2\right)$. Replacing t by 0 in the previous polynomial, we get

$f^T = b^2x^2 + a^2y^2 = h_1 \cdot h_2$, where $h_1 = bx + \iota ay$ and $h_2 = bx - \iota ay$. By taking the partial derivatives we get $f_x^T = 2b^2x$, $f_y^T = 2a^2y$. Thus, $f_x^T + \iota f_y^T = 2\left(b^2x + \iota a^2y\right) \notin \langle h_1 \rangle$, and $f_x^T - \iota f_y^T = 2\left(b^2x - \iota a^2y\right) \notin \langle h_2 \rangle$ unless $a^2 = b^2$. Thus, the dimension of the component at infinity of the projective completion of the generalised offset equals 0 unless $a = b$, in which case the conic is a circle. Thus, $l = 0$. The ellipse (or the hyperbola) does not have singular points, thus $C_S = \emptyset$. Thus, the degree of the generalised offset to the ellipse is $2 \cdot 2^2 = 8$. The same reasoning allows one to conclude with the same degree for the hyperbola: the difference with the case of the ellipse is that the polynomial $\frac{x^2}{a^2} + \frac{y^2}{b^2} - 1$ is replaced by $\frac{x^2}{a^2} - \frac{y^2}{b^2} - 1$. □

Proposition 4.3.3. *The degree of the generalised offset to a parabola is* 6.

Proof. The parabola can be considered in some coordinate system as the affine variety $V\left(y^2 - 2px\right)$. The projective completion is the projective variety $V\left(y^2 - 2pxt\right)$. The projective completion of the normal at the point (x, y) is the projective variety $V\left(-2yu + 2xy - 2ptv + 2pty\right)$. Replacing t by 0, we get $f^T = y^2 = h_1^2$, where $h_1 = y$. By taking the partial derivatives, we get $f_x^T = 0$, $f_y^T = 2y$. Thus, $f_x^T + \iota f_y^T = 2\iota y \in \langle h_1 \rangle$. Thus, the dimension of the component at infinity of the projective completion of the generalised offset equals 1. The component at infinity is given by $f^T = y^2 = 0$, hence $(0 : 1 : 0)$. Since $V\left(f^T\right)$ and $V\left(n^T\right)$ do not meet tangentially, $\sum_{p \in C_\infty} \mu_{p'}\left(V\left(f^T\right), V\left(n^T\right)\right) = 1$, and the degree of W_∞ is 2. Since a parabola does not have singular points, $C_S = \emptyset$. Thus, the degree of the generalised offset to the parabola is: $2 \cdot 2^2 - 2 \cdot 1 \cdot 1 = 8 - 2 = 6$. □

Lastly, we consider the generalised offset curve of two straight lines. The two straight lines are the following affine variety $V\left((ax + by + c)(dx + ey + f)\right)$. It is the union of $V\left(ax + by + c\right)$ and of $V\left(dx + ey + f\right)$. The projective completion of the two straight lines variety is the affine variety $V\left((ax + by + ct)(dx + ey + ft)\right)$. Replacing t by 0 in the previous polynomial, we

get $f^T = (ax+by)(dx+ey) = h_1 \cdot h_2$, where $h_1 = ax+by$ and $h_2 = dx+ey$. By taking the partial derivatives of the previous polynomial, we get $f_x^T = a(dx+ey) + d(ax+by)$, $f_y^T = b(dx+ey) + e(ax+by)$. Thus, $f_x^T + \iota f_y^T = (a(dx+ey) + d(ax+by)) + \iota(b(dx+ey) + e(ax+by))$ and $f_x^T - \iota f_y^T = (a(dx+ey) + d(ax+by)) - \iota(b(dx+ey) + e(ax+by))$.

Lemma 4.3.4. *$l = 1 \Leftrightarrow ae - bd = 0$ or $a \pm \iota b = 0$ or $d \pm \iota e = 0$.*

Proof. $\Rightarrow$: There must exist a polynomial g of $K[x,y]$ such as $(a(dx+ey) + d(ax+by)) \pm \iota(b(dx+ey) + e(ax+by)) = g(ax+by)$ or $(a(dx+ey) + d(ax+by)) \pm \iota(b(dx+ey) + e(ax+by)) = g(dx+ey)$.

Thus,

$$\begin{cases} (ax+by)(d \pm \iota e) + (dx+ey)(a \pm \iota b) = g(ax+by) \\ or \\ (ax+by)(d \pm \iota e) + (dx+ey)(a \pm \iota b) = g(dx+ey) \end{cases}.$$

The first equality implies that $ae - bd = 0$ or $a \pm \iota b = 0$. The second equality implies that $ae - bd = 0$ or $d \pm \iota e = 0$. For conics with real coefficients, this implies that $ae - bd = 0$.

$\Leftarrow$: If $ae - bd = 0$ then there exists $z \in k$ such that $dx + ey = z(ax+by)$. Thus, $f_x^T \pm \iota f_y^T = (ax+by)(d \pm \iota e) + (dx+ey)(a \pm \iota b) = (ax+by)(d \pm \iota e + z(a \pm \iota b))$, and $f_x^T \pm \iota f_y^T \in \langle h_1 \rangle$.

If $a \pm \iota b = 0$, $f_x^T \pm \iota f_y^T = (ax+by)(d \pm \iota e) + (dx+ey)(a \pm \iota b) = (d \pm \iota e)(ax+by)$. Thus, $f_x^T \pm \iota f_y^T \in \langle h_1 \rangle$.

If $d \pm \iota e = 0$, then $f_x^T \pm \iota f_y^T = (ax+by)(d \pm \iota e) + (dx+ey)(a \pm \iota b) = (a \pm \iota b)(dx+ey)$. Thus, $f_x^T \pm \iota f_y^T \in \langle h_2 \rangle$. In all these cases, $l = 1$. □

Proposition 4.3.5. *The degree of the generalised offset curve of two distinct straight lines is* 4.

Proof. According to Lemma 4.3.4, $l = 1 \Leftrightarrow ae - bd = 0 or a \pm \iota b = 0 or d \pm \iota e = 0$. However, for real straight lines (i.e. defined by polynomials with real coefficients), the

dimension of the component at infinity of the projective completion of the generalised offset equals 1 if, and only if $ae-bd=0$, since $a\pm\iota b=0 or d\pm\iota e=0$ is impossible with a,b,d,e being all real coefficients. Thus, the dimension of the component at infinity of the projective completion of the generalised offset equals 1 if, and only if, the two straight lines are parallel and distinct (if they were not f would not be square-free). The component at infinity of two parallel and distinct straight lines is one-dimensional and it is the zero set of $f^T = \left((a+d)^2+(b+e)^2\right)(ax+by)^2$, i.e. of $(ax+by)^2$. Thus, since $V\left(f^T\right)$ and $V\left(n^T\right)$ do not meet tangentially, $\sum_{p\in C_\infty}\mu_{p'}\left(V\left(f^T\right),V\left(n^T\right)\right)=2$ and the degree of W_∞ is 4. If the two straight lines are parallel, the variety does not have singular points, and $C_S=\emptyset$. Thus, the degree of the generalised offset to two parallel and distinct straight lines is $2\cdot 2^2-4-0=4$. Otherwise, the two straight lines have a single real intersection, $l=0$, and the variety has exactly one singular point (the intersection point) of multiplicity 2. Thus, since $V\left(\overline{f}\right)$ and $V\left(\overline{n}\right)$ do not meet tangentially, $\sum_{q\in C_S}\mu_{q'}\left(V\left(\overline{f}\right),V\left(\overline{n}\right)\right)=2$, and the degree of the generalised offset to two non-parallel straight lines is $2\cdot 2^2-0-4=4$. □

The results of this section are summarised in the following Table 4.1.

Conic	ellipse/hyperbola	parabola	circle	two lines
Offset degree	8	6	4	4

Table 4.1: The degree of the generalised offset to conics

4.4 An implicit equation of the generalised offset to a conic

In this section, we use the results of the preceding section in order to compute an implicit equation of the generalised offset to a conic as a factor of a sparse resultant. This implicit equation of the generalised offset to a conic will be used in both the algebraic formalisation (in Chapter 5) and the numerical computation (in Section

6.3) of the Delaunay graph conflict locator for conics.

A conic C can be defined implicitly as the variety defined by a second degree formal polynomial: $f(x,y) = \alpha x^2 + \beta xy + \gamma y^2 + \delta x + \epsilon y + \zeta = 0$. We shall compute an implicit equation of its generalised offset. The partial derivatives are $f_x = 2\alpha x + \beta y + \delta$ and $f_y = \beta x + 2\gamma y + \epsilon$. An equation of the normal $\mathcal{N} = V(n) \subset K^4$ to the original conic at the point (x,y) is: $n(x,y,u,v) = -(\beta x + 2\gamma y + \epsilon)(u - x) + (2\alpha x + \beta y + \delta)(v - y) = 0$. If a conic is not degenerate (proper conic different from the union of two lines), then it has no singular points, and $C_S = W_S = \emptyset$.

The generalised offset to a conic is the Zariski closure of the projection of the affine variety $\mathrm{V}(f,n,d) \setminus V(f_x, f_y)$ onto the (u,v) plane. Its implicit equation is thus a factor of the resultant expressing the elimination of the variables x and y from the three polynomials f, n, d. However, if the conic is not degenerate, an implicit equation of the generalised offset is the sparse resultant itself. Since we are using the sparse resultant instead of the "normal" projective resultant, we should pay attention to the fact the sparse resultant is a necessary condition of existence of common solutions in $(\mathbb{C}*)^N$ instead of $\mathbb{P}^N$. The axes of equation $x = 0$ and $y = 0$ can be part of the generalised offset to a conic only if the conic is the degenerate union of two straight lines with one of them being the line of equation $x = \pm r$ or $y = \pm r$, where r is the offset parameter. Since the conic is defined generically, the generalised offset will not generically contain one of the axes of equation $x = 0$ and $y = 0$.

The main objective that has been sought in the computations is to simplify the polynomials. This simplification has to induce a system of equations equivalent to the original system of equations in order to generate the same set of zeroes (i.e. the same variety). In the case of the sparse resultant computation, this has been

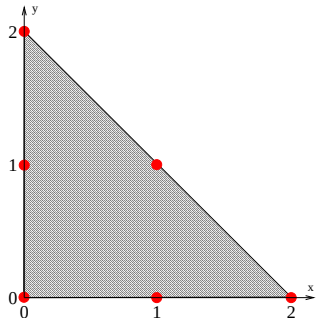

Figure 4.4.1: The Newton polytope of f.

achieved by replacing the original system of algebraic equations by an equivalent system of algebraic equations, where one polynomial is replaced by a linear combination of the polynomials in the system of equations having a Newton polytope strictly inscribed in the Newton polytope of the original polynomial.

The sparse resultants have been computed thanks to the sparse resultant software developed by Emiris [EC95, Emi97]. This software allowed us to compute the sparse resultant matrix. We computed the determinant of this matrix and its factorisation. The degree of the generalised offset to conics has been determined in Section 4.3 (see Table 4.1). This allowed us to identify the factor that corresponds to an implicit equation of the generalised offset.

In the case where α and β are different from 0, we call the conic "generic". If not stated otherwise, we will suppose the conic is generic in all this subsection. The Newton polytope of the polynomial defining $B = V(f) \subset K^4$ is illustrated in Figure 4.4.1.

The polynomial defining $\mathcal{N} = V(n) \subset K^4$ can be rewritten in the following way exhibiting its monomials in the variables x and y, which need to be eliminated in order to get an equation of the generalised offset: $n(x,y,u,v) = \beta x^2 - \beta y^2 + 2(\gamma - \alpha)xy + (-\beta u + \epsilon)x + (\beta v - \delta)y + (-\epsilon u + \delta v) = 0$. The monomial in x^2 can be eliminated if we replace $n(x,y,u,v)$ by $\alpha n(x,y,u,v) - \beta f(x,y)$. The Newton

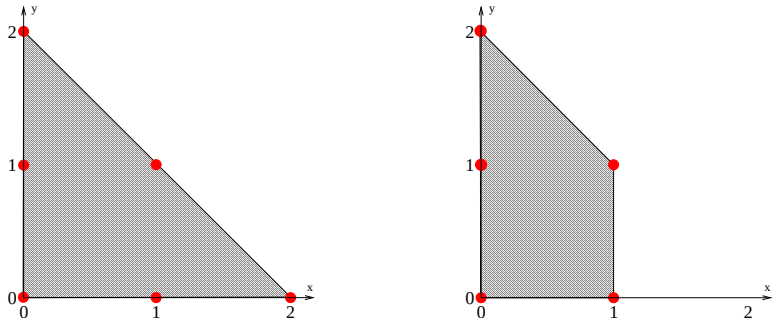

Figure 4.4.2: The Newton polytopes of n and of $\alpha n - \beta f$.

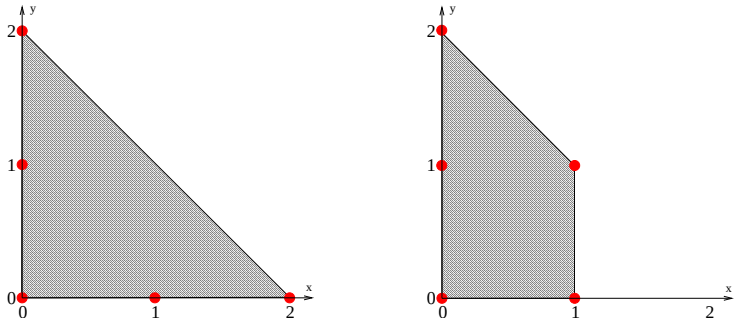

Figure 4.4.3: The Newton polytopes of d and of $f - \alpha d$

polytopes of $n\,(x, y, u, v)$ and of $\alpha n\,(x, y, u, v) - \beta f\,(x, y)$ are shown in Figure 4.4.2. It is easy to see from Figures 4.4.1, 4.4.2 and 4.4.3, that no other monomial can be eliminated in addition to x^2 if we replace n by a linear combination of n, f and d.

If we replace $d\,(x, y, u, v)$ by $f\,(x, y) - \alpha d\,(x, y, u, v)$, the monomial in x^2 disappears. The Newton polytopes of $d\,(x, y, u, v)$ and of $f\,(x, y) - \alpha d\,(x, y, u, v)$ are shown in Figure 4.4.3. It is easy to see from Figures 4.4.1, 4.4.2 and 4.4.3, that no other monomial can be eliminated in addition to x^2 if we replace d by a linear combination of d, f and n.

The mixed volumes of f and $\alpha n - \beta f$ and of f and $f - \alpha d$ are 4 (see Figures 4.4.4 and 4.4.6), and the mixed volume of $\alpha n - \beta f$ and $f - \alpha d$ is 3 (see Figure 4.4.5).

In our search for an equivalent system of algebraic equations, we could have supposed that n or d will be unchanged instead of f. In both cases, we would arrive to the same mixed volume. We get an equation for the generalised offset as the

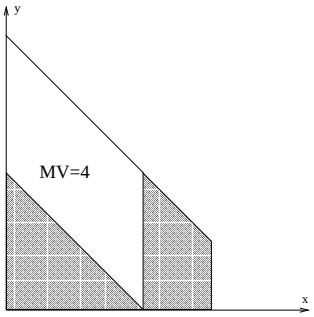

Figure 4.4.4: The mixed volume of f and $\alpha n - \beta f$.

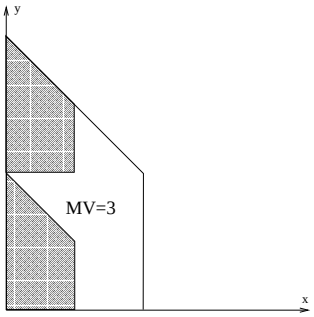

Figure 4.4.5: The mixed volume of $\alpha n - \beta f$ and $f - \alpha d$.

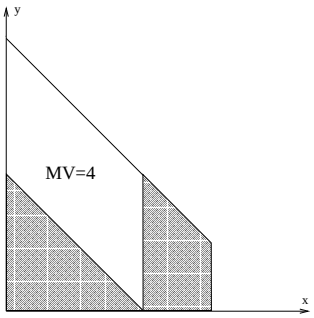

Figure 4.4.6: The mixed volume of f and $f - \alpha d$.

sparse resultant of the three polynomials f, $\alpha n - \beta f$ and $f - \alpha d$. The code written to get this sparse resultant has been placed in Appendices B.1 (Maple code generating the files that will be input to the "Resin" program [Emi97]), and B.2 (Maxima code used to compute the sparse resultant as a factor of the determinant of the matrix returned by the "Resin" program). This implicit equation is publicly available at: http:// www.cs.ubc.ca/˜fanton. However, it is possible to further simplify the equation of a non-degenerate conic by using a different coordinate system. For an ellipse or an hyperbola, a nice coordinate system has its origin at its *centre* (the intersection of its axes of symmetry), and axes the axes of symmetry of the conic. For a parabola, a nice coordinate system has its origin at its *summit* (intersection of the parabola with its axis of symmetry), and one of the axes is the axis of symmetry of the parabola.

In the case of an ellipse or an hyperbola, the equation of the conic in a coordinate system with origin at the centre of the conic, and axes the axes of symmetry of the conic, simplifies to $\frac{x^2}{a^2} \pm \frac{y^2}{b^2} - 1 = 0$, assuming both a and b are different from 0. By multiplying this equation by a^2, we get an equivalent equation of the form $x^2 \pm \frac{a^2}{b^2} y^2 - a^2 = 0$. Then, by replacing $\pm\frac{a^2}{b^2}$ by c, and $-a^2$ by e, we get an equivalent equation of the form $x^2 + cy^2 + e = 0$ where $e \pm b^2 c = 0$. Let $f = x^2 + cy^2 + e$.

The Newton polytope of the polynomial defining $B = V(f) \subset K^4$ is illustrated in Figure 4.4.7.

The polynomial defining $\mathcal{N} = V(n) \subset K^4$ can be rewritten in the following way exhibiting its monomials in the variables x and y, which need to be eliminated in order to get an equation of the generalised offset: $n(x, y, u, v) = -2cy(u - x) + 2x(v - y) = 0$. The Newton polytope of $n(x, y, u, v)$ is shown in Figure 4.4.8. It is easy to see from Figures 4.4.7 and 4.4.9 that it is not possible to remove monomials from n without adding new monomials by replacing n by any linear combination of f, n and d.

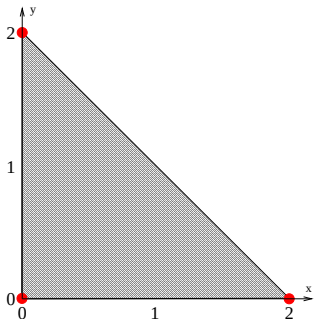

Figure 4.4.7: The Newton polytope of f for ellipses and hyperbolas.

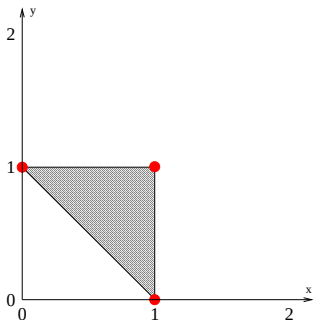

Figure 4.4.8: The Newton polytope of n for ellipses and hyperbolas.

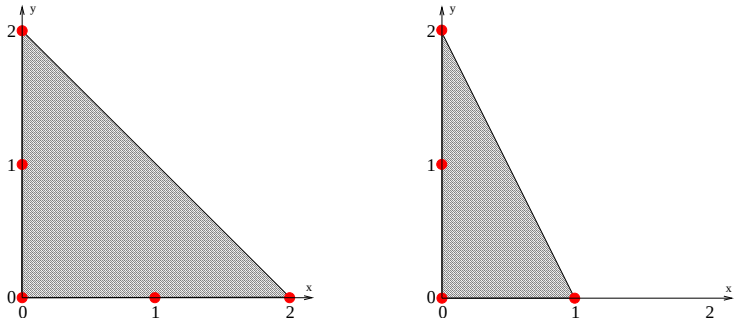

Figure 4.4.9: The Newton polytopes of d and of $ad - f$ for ellipses and hyperbolas.

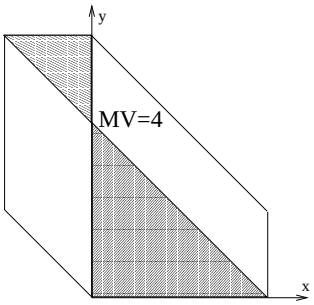

Figure 4.4.10: The mixed volume of f and n for ellipses and hyperbolas.

If we replace $d(x, y, u, v) = (u - x)^2 + (v - y)^2 - R^2 = 0$ by $f(x, y) - d(x, y, u, v)$, the monomial in x^2 disappears. The Newton polytopes of $d(x, y, u, v)$ and of $f(x, y) - d(x, y, u, v)$ are shown in Figure 4.4.9. As with that equation in the case of a generic conic, no other monomial can be eliminated in addition to x^2 if we replace d by a linear combination of d, f and n (see Figures 4.4.7, 4.4.8 and 4.4.9.

The mixed volumes of f and n and of f and $f - d$ are 4 (see Figures 4.4.10 and 4.4.12), and the mixed volume of n and $f - d$ is 3 (see Figure 4.4.11).

As before, in our search for an equivalent system of algebraic equations, we could have supposed that n or d will be unchanged instead of f. In both cases, we would arrive to greater mixed volumes. The sparse resultant of $f, n, f - d$ in the variables u and v in the case where the equation f of the ellipse or hyperbola is expressed in a coordinate system centred on its centre and with axes its axis of

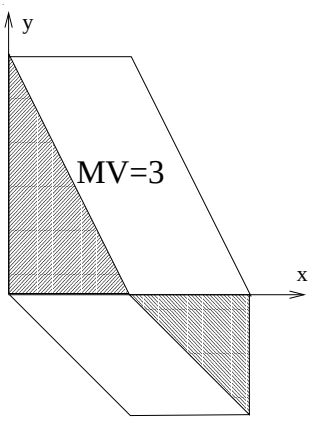

Figure 4.4.11: The mixed volume of n and $f - d$ for ellipses and hyperbolas.

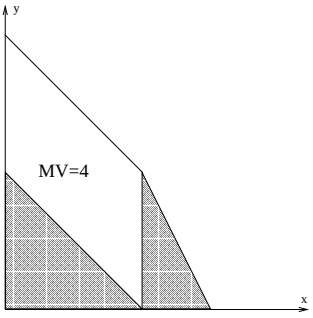

Figure 4.4.12: The mixed volume of f and $f - d$ for ellipses and hyperbolas.

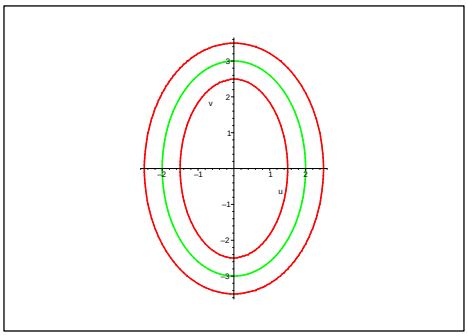

Figure 4.4.13: An ellipse and its 0.5−generalised offset

symmetry has been placed in Appendix B.4.

The degree (8) of this equation matches the degree of the generalised offset to an ellipse or hyperbola obtained in Section 4.3. By a simple change of coordinate system, we can obtain easily the general equation of the generalised offset to an ellipse or an hyperbola. As a means of verification of these symbolic computations, if we replace the formal coefficients of the equation of an ellipse or an hyperbola by their numerical values, we get exactly the same equation of the generalised offset when we use the general equation of the generalised offset to a conic or the general equation of the generalised offset to an ellipse or an hyperbola. Two examples of generalised offsets to an ellipse computed using the preceding equation are shown in Figures 4.4.13 and 4.4.14. Two examples of generalised offsets to a hyperbola computed using the preceding equation are shown in Figures 4.4.15 and 4.4.16.

In the case of a parabola, the equation of the conic in a coordinate system with origin at the summit of the parabola, and one of the axes being the axis of the parabola, simplifies to $y^2 - 2px = 0$.

The Newton polytope of the polynomial defining $B = V(f) \subset K^4$ is illustrated in Figure 4.4.17.

The polynomial defining $\mathcal{N} = V(n) \subset K^4$ can be rewritten in the following way exhibiting its monomials in the variables x and y, which need to be eliminated in

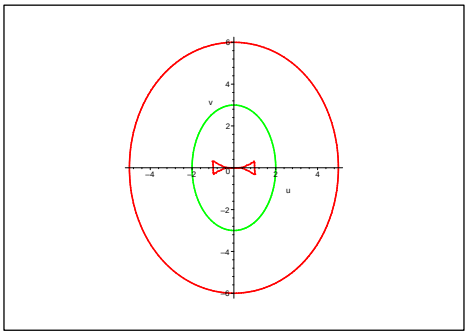

Figure 4.4.14: The same ellipse and its 3−generalised offset

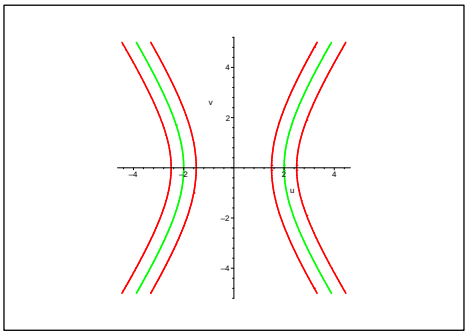

Figure 4.4.15: An hyperbola and its 0.5−generalised offset

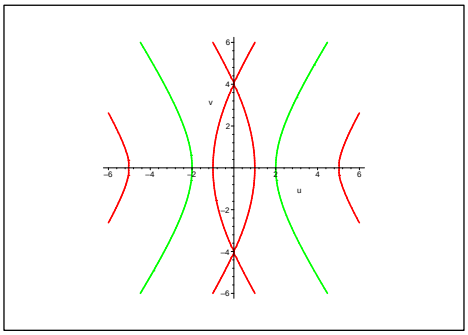

Figure 4.4.16: The same hyperbola and its 3−generalised offset

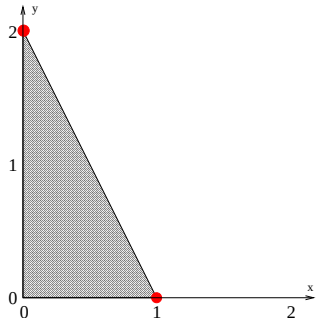

Figure 4.4.17: The Newton polytope of f for parabolas.

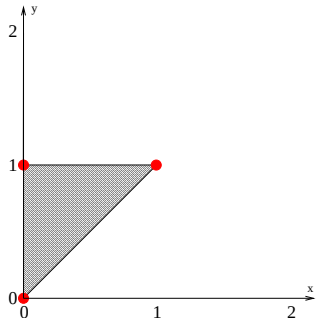

Figure 4.4.18: The Newton polytope of n for parabolas.

order to get an equation of the generalised offset: $n(x, y, u, v) = -2yu + 2xy - 2pv + 2py = 0$. This polynomial has monomials in xy, y, and a constant term. It is easy to see from Figures 4.4.17 and 4.4.19 that it is not possible to remove monomials from n without adding new monomials by replacing n by any linear combination of f, n and d. The Newton polytope of $n(x, y, u, v)$ is shown in Figure 4.4.18.

If we replace $d(x, y, u, v) = (u - x)^2 + (v - y)^2 - R^2 = 0$ by $f(x, y) - d(x, y, u, v)$, the monomial in y^2 disappears. The Newton polytopes of $d(x, y, u, v)$ and of $f(x, y) - d(x, y, u, v)$ are shown in Figure 4.4.19. It is easy to see from Figures 4.4.17, 4.4.18 and 4.4.19 that no other monomial can be eliminated in addition to y^2 if we replace d by a linear combination of d, f and n.

The mixed volumes of f and n and of n and $f - d$ are 3 (see Figures 4.4.20

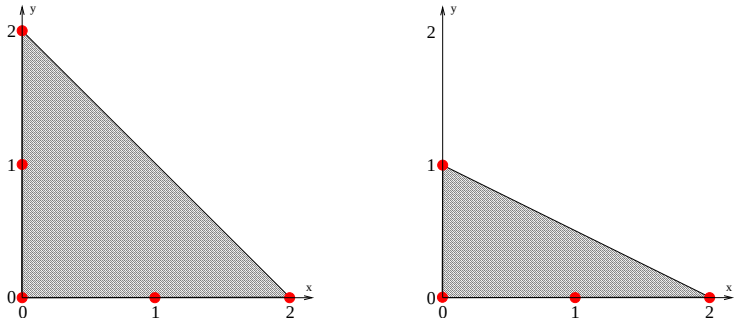

Figure 4.4.19: The Newton polytopes of d and of $f - d$ for parabolas.

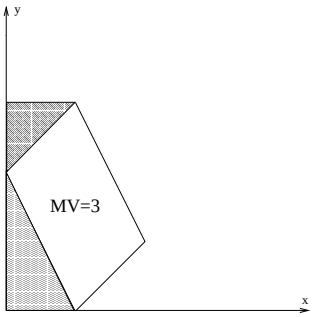

Figure 4.4.20: The mixed volume of f and n for parabolas.

and 4.4.21), and the mixed volume of f and $f - d$ is 4 (see Figure 4.4.22).

The sparse resultant of $f, n, f - d$ in the variables u and v in the case where the equation f of the parabola is expressed in a coordinate system centred on its summit, and one axis of the coordinate system is its axis of symmetry has been placed in Appendix B.3.

The degree (6) of this equation matches the degree of the generalised offset to a parabola obtained in Section 4.3. By a simple change of coordinate system, we can obtain easily the general equation of the generalised offset to a parabola. As before, as a means of verification of these symbolic computations, if we replace the formal coefficients of the equation of a parabola by their numerical values, we get exactly the same equation of the generalised offset when we use the general equation of the generalised offset to a conic or the general equation of the generalised offset to a parabola. Two examples of generalised offsets to a parabola computed using

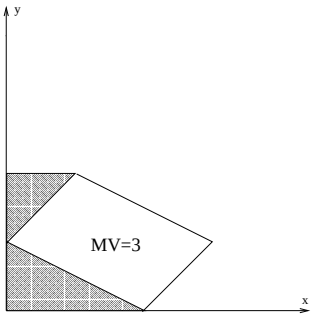

Figure 4.4.21: The mixed volume of n and $f - d$ for parabolas.

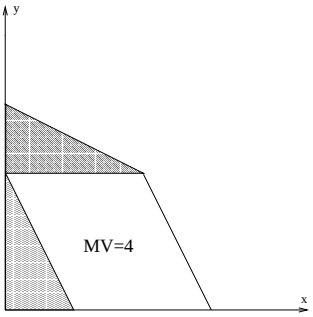

Figure 4.4.22: The mixed volume of f and $f - d$ for parabolas.

the preceding equation are shown in Figures 4.4.23 and 4.4.24.

4.5 Conclusions

We have obtained a general formula for the degree of the generalised offset to an algebraic curve defined in its most general setting: an implicit equation with coefficients in a zero characteristic algebraically closed field. We have applied it to compute the degree of the generalised offset to conics. We have obtained an implicit equation of the generalised offset to a conic defined by a formal polynomial. We have also obtained simplified equations in two cases: in the first case, the conic is a circle, an ellipse or an hyperbola defined by a formal polynomial; and in the second case, the conic is a parabola defined by a formal polynomial. The same simplific-

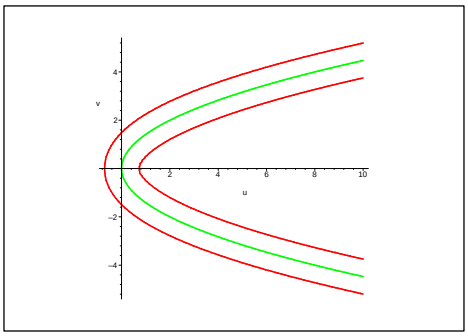

Figure 4.4.23: A parabola and its 0.5−generalised offset

ation can be obtained in the case of a degenerate conic (i.e. two straight lines). This implicit equation of the generalised offset to a conic has been used in order to get algebraic descriptions for the generalised Voronoi vertex of three conics and the algebraic Delaunay graph conflict locator for conics.

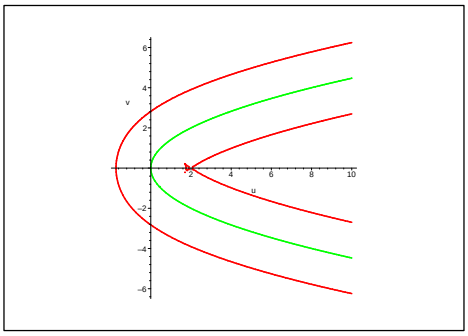

Figure 4.4.24: The same parabola and its 3−generalised offset

Chapter 5

The Delaunay graph conflict locator for conics

In this chapter, we present an algebraic approach to the computation of the Delaunay graph conflict locator in the case of conics.

In Section 5.1, we will show how the definitions related to generalised Voronoi diagrams presented in Chapter 1 adapt to the case where the sites are conics. In Section 5.2, we will present the formalisation of the Delaunay graph conflict locator for conics. In Section 5.3, we present the simplification of the equations defining the Delaunay graph conflict locator for conics. In Section 5.4, we present how these simplications allowed us to compute the matrix of the sparse resultant used for the computation of the Delaunay graph conflict locator. Finally, in Section 5.5, we present the numerical computation of the Delaunay graph conflict locator.

5.1 Preliminaries

We consider now $M = \mathbb{R}^2$. Let $\mathcal{S} = \{C_1, ..., C_m\} \subset M, m \geq 2$ be a set of m different conics in M. Let us define the distance $d(p, s_i)$ between a point p and a conic C_i as $d(p, s_i) = \inf \delta(p, q) \,|q \in s_i$, where δ denote the Euclidean distance between points. The definitions of influence zone, bisector, Voronoi region and Voronoi diagram we

presented in Chapter 1 hold, and in this context they are the definitions of influence zone, bisector, Voronoi region and Voronoi diagram of conics.

5.2 Formalisation of the Delaunay graph conflict locator

In this section, we consider the maintenance of the Delaunay graph for conics in an incremental way: we check the validity of the triangles of the Delaunay graph of the set of sites $\mathcal{S}$ formed by a given triple of conics with respect to a newly inserted conic. The reason of the simultaneous treatment of all the triangles of the Delaunay graph formed by a given triple is algebraic: we cannot treat portions of curves because they are semi-algebraic sets instead of being algebraic sets. We need to treat whole algebraic curves. Thus, four conics C_1, C_2, C_3 and C_4 are given: the first three are supposed to be the vertices of one or more triangles in the Delaunay graph, and the last one is the newly inserted conic. Our approach is similar to the one we used in [AKM02]. We consider now the notion of generalised offset of a conic (see Section 4.1), which can be seen as an expansion/shrinking of conics.

We introduce the notion of generalised Voronoi vertex:

Definition 5.2.1. (generalised Voronoi vertex) A *generalised Voronoi vertex* of three conics C_1, C_2, and C_3 is a point of intersection of the generalised offsets of C_1, C_2, and C_3 with the same offset parameter (see Example on Figure 5.2.1).

The generalised Voronoi vertex shown in Figure 5.2.1 is not a true Voronoi vertex because the circle centred on that vertex and tangent to the three conics has points of one of the conics in its interior. In contrast, the generalised Voronoi vertex shown in Figure 5.2.2 is a true Voronoi vertex: the circle centred on that vertex and tangent to the three conics has no points of any of the conics in its interior.

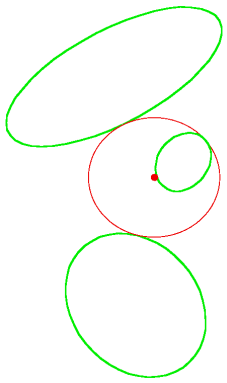

Figure 5.2.1: A generalised Voronoi vertex (dot) of three conics (thick lines)

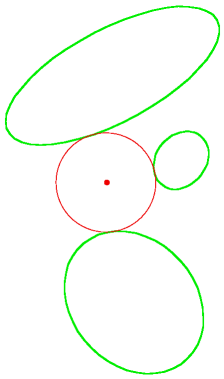

Figure 5.2.2: A true Voronoi vertex of three conics (thick lines)

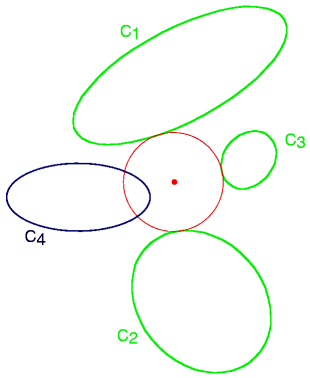

Figure 5.2.3: The insertion of C_4 induces a conflict with the triangle $C_1C_2C_3$

The objective of the Delaunay graph conflict locator is to determine whether or not the insertion of a new conic C_4 would change each one of the triangles of the Delaunay graph $DG(\mathcal{S})$ of $\mathcal{S}$ corresponding to the triple C_1, C_2, C_3. There are two possible outcomes to the conflict locator: either the triangles $C_1C_2C_3$ are valid with respect to C_4 and the triangles remain in the new Delaunay graph, or some triangles are not valid with respect to C_4 and these triangles will not be present in the Delaunay graph (and in the quad-edge data structure storing it) any longer. We can see an example of the later case in Figure 5.2.3. One of the two triangles $C_1C_2C_3$ is not valid with respect to the conic C_4, thus that triangle will not belong any more to the Delaunay graph (see Figure 5.2.4).

The Delaunay graph conflict locator consists of determining which of the true Voronoi vertices of the conics C_1, C_2, and C_3 are at a distance (denoted as R) with respect to C_4 lower than the distance (denoted as r) with respect to C_1, C_2, and C_3 (see Figure 5.2.5).

Let (x, y) be the coordinates of a generalised Voronoi vertex of C_1, C_2, and C_3. Let r be the local distance between the generalised Voronoi vertex of C_1, C_2, and C_3 and C_1. Let R be the local distance between the generalised Voronoi vertex of C_1, C_2, and C_3 and C_4. Let X be the set of all possible values of (x, y, r, R) that

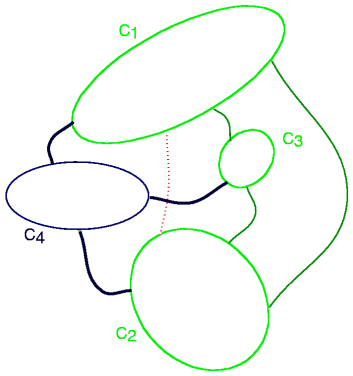

Figure 5.2.4: The new Delaunay graph after insertion of C_4: the edges in plain thin lines remain, those in dashed lines disappear, and those in plain thick lines appear

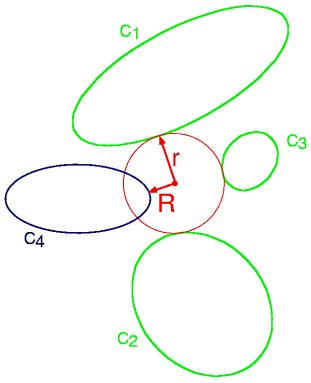

Figure 5.2.5: The Delaunay graph conflict locator for conics

are solutions to the system of four equations composed of the implicit equations of the three $r-$generalised offsets to C_1, C_2, and C_3 and the $R-$generalised offset to C_4 in the four unknowns x, y, r, R, such that (x, y) is a true Voronoi vertex of C_1, C_2, and C_3.

The Delaunay graph conflict locator is computed in two phases by evaluating the sign of the polynomial $R - r$ among the solutions (x, y, r, R) of the system composed of the three $r-$generalised offsets to C_1, C_2, and C_3 and the $R-$generalised offset to C_4 in the four unknowns x, y, r, R, and computing the x, y, r coordinates of the solutions for which $R - r$ has the desired sign, in order to isolate them. The computation of the x, y, r coordinates is not used as an intermediary computation of the Delaunay graph conflict locator computation. It is used in order to check that the coordinates of a solution of the preceding system of equations is a true Voronoi vertex (this is done in a second phase which is explained below). If one of the points of X is such that $R - r < 0$, then there is a change in the Delaunay graph. The true Voronoi vertices are the generalised Voronoi vertices whose empty circle does not conflict with any of the first three (defining) conics. This is checked in the second phase by evaluating the sign of R-r on the solutions of each one of the systems composed of the three $r-$generalised offsets to C_1, C_2, and C_3 and the $R-$generalised offset to a fourth conic, where the fourth conic is alternatively each one of the first three conics, and isolating the x, y, r coordinates of the solutions for which $R - r \geq 0$. The true Voronoi vertices (x, y) are those for which $R - r \geq 0$ for any solution (x, y, r, R) of any system composed of the three $r-$generalised offsets to C_1, C_2, and C_3 and the $R-$generalised offset to C_4 in the four unknowns x, y, r, R, where C_4 is alternatively C_1, C_2, and C_3. We can summarise this paragraph with the following theorem:

Theorem 5.2.2. *The Delaunay graph conflict locator output list is empty if, and only if, $R - r$ does not take a negative real value on the points of X.*

5.3 The simplification of the Delaunay graph conflict locator

In this section, we will present the simplification of the algebraic equations specifying the generalised Voronoi vertices of C_1, C_2 and C_3 and their local distances r to C_1 or C_2 or C_3, and R to C_4. This simplification allowed us to compute the sparse resultant needed for the algebraic computation of the Delaunay graph conflict locator.

Since the offsets to ellipses or hyperbolas have degree 8 while the offsets to parabolas have degree 6 (see Table 4.1), we will call C_1 a conic whose offset has maximum degree among C_1, C_2, and C_3: one of the ellipses or hyperbolas among C_1, C_2, and C_3 if there is one, or a parabola if there are no ellipses nor hyperbolas among C_1, C_2, and C_3. We will suppose without loss of generality that we can make a change of coordinate system such that the x-axis of our new coordinate system is parallel to the main axis of symmetry of C_1, and that the origin of the new coordinate system is the summit of C_1 if C_1 is a parabola or its centre if C_1 is a conic with centre (ellipse or hyperbola). Thus in this new system of coordinates, the equation of C_1 is either $\frac{x^2}{a^2} \pm \frac{y^2}{b^2} - 1 = 0$ or $y^2 - 2p_1 x = 0$. The equations of the conic $C_i, i \neq 1$ can be obtained from the generic equation of a conic in a system centered on the summit/centre of C_i and with x-axis the main axis of symmetry of C_i by applying an affine transformation given by: $x = \alpha_i X - \beta_i Y + \gamma_i$ and $y = \beta_i X + \alpha_i Y + \delta_i$. In the same way we can obtain an implicit equation of the $r-$generalised offset to C_i from the generic equation of the $r-$generalised offset to C_i in a system centered on the summit/centre of C_i and with x-axis the main axis of symmetry of C_i by applying the affine transformation given by: $x = \alpha_i X - \beta_i Y + \gamma_i$ and $y = \beta_i X + \alpha_i Y + \delta_i$. We have already studied and obtained an implicit equation of the generalised offset to the different types of conics in Section 4.4. Observe that the variable r appears

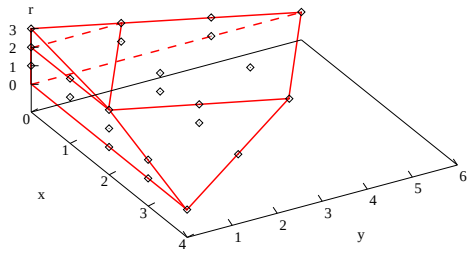

Figure 5.3.1: The Newton polytope of the implicit equation of the $r-$generalised offset to a parabola in a system centred on its summit and with x-axis the axis of symmetry of the parabola

always as powers of r^2 since the only term in which it occurs is the term r^2 of the polynomials $d_i\,(x, y, u, v)$ expressing the distance between the point on the curve C_i and the point on the generalised offset. Therefore, we can consider the following change of variable $r^2 -> r$ for all the conics. Although, we have to eliminate the four variables x, y, r, R, and thus we need to consider the Newton polytopes in K^4, we represented all the Newton polytopes of this section in K^3 since the variable R only appears in the implicit equation of the $R-$generalised offset to the fourth conic.

An implicit equation of the $r-$generalised offset to a parabola of parameter p in a system centred on its summit and with x-axis its axis of symmetry has been placed in Appendix B.3.

The Newton polytope of the implicit equation of the $r-$generalised offset to a parabola in a system centred on its summit and with x-axis the axis of symmetry of the parabola is shown on Figure 5.3.1.

The Newton polytope of the implicit equation of the $r-$generalised offset to a parabola in an arbitrary system is shown on Figure 5.3.2.

However, if we subtract the implicit equation of the $r-$generalised offset

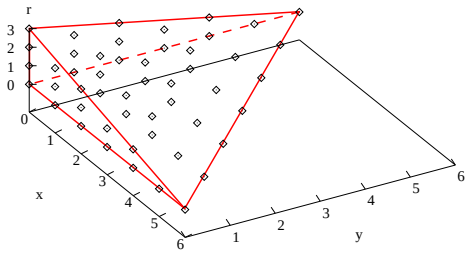

Figure 5.3.2: The Newton polytope of the implicit equation of the $r-$generalised offset to a parabola in an arbitrary system

to a parabola in a nice system centred on its summit and with x-axis its axis of symmetry from this equation, the term in r^3 disappears, and the corresponding Newton polytope is shown on Figure 5.3.3.

An implicit equation of the generalised offset to an ellipse or hyperbola of big axis a and small axis b in a system centred on the centre of the conic and with axes the axes of symmetry of the conic has been placed in Appendix B.4.

The Newton polytope of the implicit equation of the $r-$generalised offset to an ellipse or hyperbola in a system centred on the centre of the conic and with axes the axes of symmetry of the conic is shown on Figure 5.3.4.

The Newton polytope of the implicit equation of the $r-$generalised offset to an ellipse or an hyperbola in an arbitrary system is shown on Figure 5.3.5.

However, in a linear combination of the implicit equations of the $r-$generalised offset to a parabola in a nice system centred on the centre of the conic and with axes the axes of symmetry of the conic and in an arbitrary system, the term in r^4 disappears, and the corresponding Newton polytope is shown on Figure 5.3.6.

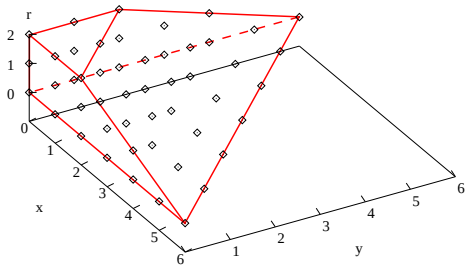

Figure 5.3.3: The Newton polytope of the difference of the implicit equations of the generalised offset to a parabola in a nice system and in an arbitrary system

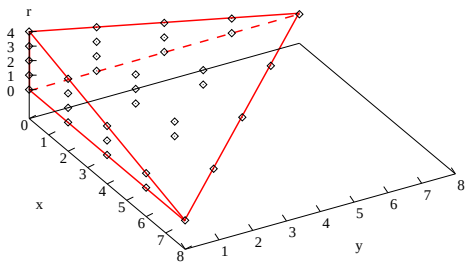

Figure 5.3.4: The Newton polytope of the implicit equation of the $r-$generalised offset to an ellipse or an hyperbola in a system centred on the centre of the conic and with axes the axes of symmetry of the conic

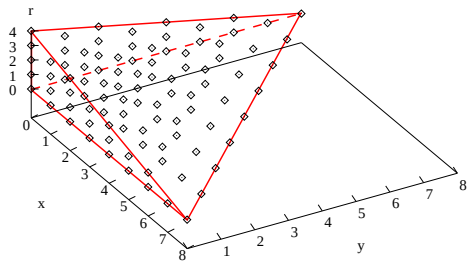

Figure 5.3.5: The Newton polytope of the implicit equation of the $r-$generalised offset to an ellipse or an hyperbola in an arbitrary system

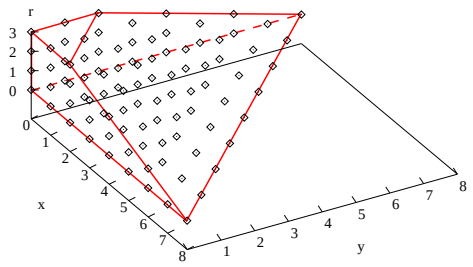

Figure 5.3.6: The Newton polytope of a linear combination of the implicit equations of the generalised offset to an ellipse or an hyperbola in a nice system and in an arbitrary system

5.4 The algebraic precomputations

In this section, we present how the simplifications described in the preceding section allowed us to compute the matrix of the sparse resultant needed for the algebraic computation of the Delaunay graph conflict locator of conics.

Let o_i denote the polynomial expressing the implicit equation of the generalised offset of the conic C_i in the coordinate system based on C_1.

In order to evaluate the sign of the polynomial $R - r$ among the solutions (x, y, r, R) of the system composed of the three $r-$generalised offsets to C_1, C_2, and C_3 and the $R-$generalised offset to C_4 in the four unknowns x, y, r, R, we evaluate the sparse resultant matrix (i.e. the Newton matrix) of the polynomials $R - r$, o_1, o_2, o_3, and o_4. Indeed, the matrix of the multiplication map by $R - r$ in the quotient algebra $K[x, y, r, R]/ < o_1, o_2, o_3, o_4 >$ is the transpose of the Schur complement of the submatrix M_{11} of the sparse resultant matrix of the polynomials $R - r$, o_1, o_2, o_3, and o_4 (see Section 3.3.1). The sparse resultant matrix is a square matrix of size up to 7995 (in the case of four ellipses/hyperbolas). The computation of this Schur complement involves the computation of the inverse of the matrix M_{11}, which is a square matrix of up to $7995 - 1024 = 6971$ rows (in the case of four ellipses/hyperbolas). Even with a UNIX workstation equipped with 4 Gb of RAM (cosimo or ginevra machines at Medicis [CNR]), that computation on a symbolic matrix halts due to a lack of memory. Thus, the Schur complement computation must be done after having replaced formal coefficients by their numerical values (fractions). The different possible configurations and the corresponding mixed volumes and the degree of the sparse resultant are shown in Table 5.1. The sparse resultant computations were done using Singular [GPS01], on both an Apple PowerBook G4 with 1Gb of RAM running under Mac OS X and the cosimo machine at Medicis [CNR] (a Compaq DS20E - Alpha EV 6 with 4Gb of RAM running under OSF/1 4.0f). We have given the Singular [GPS01] code for the cases of four

C_1	C_2	C_3	C_4	MV_{-1}	MV_{-2}	MV_{-3}	MV_{-4}	MV_{-0}	$deg\,(R_{\mathcal{A}})$
P	P	P	P	108	108	108	108	324	756
P	P	P	Eh	144	144	144	108	432	972
Eh	P	P	P	108	144	144	144	432	972
Eh	Eh	P	P	144	144	192	192	576	1248
Eh	Eh	Eh	P	192	192	192	256	768	1600
Eh	Eh	P	P	144	192	192	144	576	1248
Eh	Eh	P	Eh	192	192	256	192	768	1600
Eh	Eh	Eh	Eh	256	256	256	256	1024	2048

Table 5.1: The mixed volumes and sparse resultant degrees of the different configurations of conics (Eh stands for ellipse or hyperbola, P stands for parabola)

parabolas and four ellipses/circles/hyperbolas (see Appendix C).

5.5 The numerical computation of the Delaunay graph conflict locator

Observe that the sparse resultant matrix of the Delaunay graph conflict locator is very sparse: for the case of four ellipses/circles/hyperbolas, the sparse resultant matrix is a 7995×7995 matrix with only 589610 non-zero entries, i.e. a density of 0.92%! The computation of the Schur complement and of the eigenvalues should be done using methods for sparse matrices. Such computations using standard matrix methods can take several days on Singular [GPS01]. The computation of eigenvalues with ARPACK takes $O\,(N)$ time where N is the number of lines of the square matrix. The computation of the x, y, r coordinates can be done in theory by simultaneous orthogonalisation of the matrix of the multiplication operator (see Section 3.3.1). The positions (row or column number) of solutions for which $R - r$ has the desired sign are first stored. Then, the coordinates corresponding to $R-r$ having the desired sign can be read on the diagonal of the corresponding orthogonalised multiplication operator matrix. In practice however, the computation of the Schur complement of the matrix M_{11} involves the computation of the inverse of the matrix M_{11}. In the

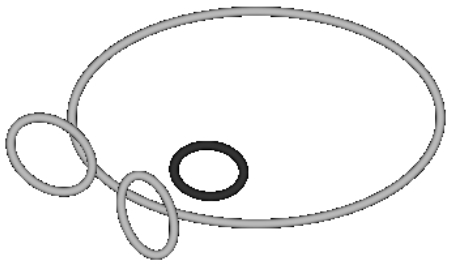

Figure 5.5.1: The test case with four ellipses

case of four ellipses/circles/hyperbolas, the size of M_{11} is $7995 - 1024 = 6971$. We have chosen a common case of four ellipses for the computation of the Delaunay graph conflict locator (see Figure 5.5.1). The computation of the Schur complement takes 1 minute and half, and the computation of the eigenvalues takes also 1 minute and half.

The main problem in the computation of the Schur complement is that the matrix M_{11} is badly conditioned due to the fact the numerical coefficients of this matrix are very huge. This induces a precision problem in Matlab [LM90]. This is due to the fact Matlab does all the computations considering matrix coefficients as doubles. The alternative is to do the computation of the Schur complement of the Newton matrix using a Computer Algebra System (like Maple [CGGL92], Singular [GPS01], or Maxima [GG82]) that does exact computations on fractional numbers, and then use a method of certification (like [Krä92]) that computes tight bounds on the intervals taken by the eigenvalues. The problem with this computation is that it takes far too much time for being a practical implementation within the context of a randomised incremental algorithm for the construction of the Voronoi diagram of conics. Another alternative is to compute the intervals taken by the entries of the Schur complement, and then certify the computation of the eigenvalues. However, we can't apply the certification method for computing eigenvalues of matrices with exact entries [Krä92]. This is due to the fact the Schur complement is not a continuous

function of the matrix entries. We already observed this problem in the simultaneous orthogonalisation of the matrices of the multiplication operators in Section 3.2.

Thus, we can conclude that even though we could compute the Delaunay graph conflict locator "exactly" by computing the Schur complement of a submatrix of the sparse resultant (or Newton) matrix with a Computer Algebra System that does exact computations on fractions, this method would not lead to a practical implementation of the conflict locator that is fast enough for the randomised incremental construction of the Voronoi diagram of conics. The main problem if we compute the Schur complement already mentioned above using floating point computations is that we must certify the results using interval analysis on the Schur complement computation, and then we should do the same thing (using interval analysis) on the computation of the eigenvalues.

Chapter 6

The Delaunay graph conflict locator for semi-algebraic sets

In this chapter, we will first present a Delaunay graph conflict locator for sites being semi-algebraic sets. This conflict locator checks whether the addition of a given semi-algebraic set would remove or not each facet of the Delaunay graph $DG(\mathcal{S})$ of a set $\mathcal{S}$ of semi-algebraic sets whose vertices are a given $(N+1)-$tuple of semi-algebraic sets. Thus, the input is an $(N+2)-$tuple of semi-algebraic sets, where the first $N+1$ semi-algebraic sets define one or more facets of the Delaunay graph, and the $(N+2)^{\text{th}}$ semi-algebraic set is the semi-algebraic set being added. This conflict locator will be based on the ALIAS library [Mer00], whose important features used here have been presented in Section 3.3.3. We will first present the system of equations and inequalities that must be satisfied if the conflict locator outcome is positive (the addition of the semi-algebraic set would remove one or more triangles) in Section 1. We will then present in Section 2 how we check the existence of solutions to the preceding system of algebraic equations and inequalities using ALIAS (with different solving techniques and corresponding parameters). We will then present in Section 3 how the implicit equation of the generalised offset to a conic (see Section 4.4) can be used in order to accelerate the computation of the

conflict locator in the case the semi-algebraic sets are conics. This involves both symbolic algebraic precomputations and scientific computations.

6.1 The algebraic equations and inequalities of the Delaunay graph conflict locator

The definition of a semi-algebraic set has already been presented in Definition 1.0.1 in Chapter 1. Let $X_1, ..., X_{N+2}$, be semi-algebraic sets. Let us suppose that each semi-algebraic set X_i is defined as $\bigcup_{j=1}^{s_i} \bigcap_{k=1}^{r_{i,j}} \left\{x \in \mathbb{R}^N | f_{i,j,k} \star_{i,j,k} 0\right\}$,
where $f_{i,j,k} \in \mathbb{R}\left[x_{i_1}, ..., x_{i_N}\right]$ and $\star_{i,j,k}$ is either $<$ or $=$, for $i = 1, 2, 3, 4$, $j = 1, ..., s_i$ and $k = 1, ..., r_{i,j}$.

Let us assume without loss of generality that each $\bigcap_{k=1}^{r_{i,j}} \left\{x \in \mathbb{R}^N | f_{i,j,k} \star_{i,j,k} 0\right\}$ for each X_i is defined by at least one non-trivial algebraic equation (i.e. different from the zero polynomial). This is equivalent to each component of each semi-algebraic set being defined as the restriction of a proper closed set in the Zariski topology sense. Thus each component of each semi-algebraic set has a codimension greater or equal to 1. If our starting assumption is not valid in the case we treat, we can make it valid by adding the equations corresponding to $f_{i,j,k} = 0$ for each (i, j, k) such that j is the index of a component that is not defined as in the assumption and i is the index of the semi-algebraic set to which the component belongs. This realises the (real) topological closure of the set of solutions of the $f_{i,j,k} \star_{i,j,k} 0$ such that j is the index of a component that is not defined as in the assumption and i is the index of the semi-algebraic set to which the component belongs. Let us denote V_i as the intersection of all the $V\left(f_{i,j,k}\right)$ such that $\star_{i,j,k}$ is $=$ for each $i = 1, 2, 3, 4$. Let $\mathcal{N}_i$ be the normal space to V_i at the point $x_i = \left(x_{i_1}, ..., x_{i_N}\right)$. Each $f_{i,j,k}$ defining V_i induces $N - 1$ polynomials $n_{i,j,k,l}$ with $l = 1, ..., N - 1$ that are the equations defining the normal to $V\left(f_{i,j,k}\right)$ at x_i. A point $q = (y_1, ..., y_N)$ belongs to $\mathcal{N}_i$ if its coordinates satisfy each one of the equations of the normal spaces to $V\left(f_{i,j,k}\right)$ at x_i

such that $\star_{i,j,k}$ is $=$.

For a given $q = (y_1, ..., y_N)$, let $\mathcal{M}_i$ be the the set of points $m_i = (z_{i_1}, ..., z_{i_N}) \in X_i$ such that q belongs to the normal space to V_i at the point m_i.

In the general case, each set $\mathcal{M}_i$ is a finite set of points. However, if V_i contains a portion of hypersphere $PHS(q, \rho)$ centered on q, then $\mathcal{M}_i$ contains that portion of hypersphere. To get in all cases a finite set of points m_i of V_i, we use $\mathfrak{S}_i = \mathcal{M}_i$ when $\mathcal{M}_i$ is finite, and $\mathfrak{S}_i \bigcap PHS(q, \rho) = \{w_i\}$ for an arbitrary point w_i of $PHS(q, \rho)$ when V_i contains a portion of hypersphere $PHS(q, \rho)$ centered on q.

We are now able to write the system of algebraic equations and inequalities that define the outcome of the Delaunay graph conflict locator. Let us consider the map $\pi : K^{3N} \to K^N$ defined by $\pi(x_i, q, m_i) = q$.

The point q is at the distance r from the point x_i if, and only if, the distance between q and x_i is r. This is expressed algebraically by the equation $d_i(q, x_i) = (y_1 - x_{i_1})^2 + ... + (y_N - x_{i_N})^2 - r^2 = 0$.

The generalised r-offset $\mathcal{O}_i$ to X_i is the image by π of the points of K^{3N} defined by the following system of equations and inequalities:

$$\left\{ \begin{array}{l} \exists j \in [1, s_i], \forall k \in [1, r_{i,j}], \\ \quad \left\{ \begin{array}{l} f_{i,j,k}(x_i) \star_{i,j,k} 0 \\ \text{if } \star_{i,j,k} \text{ is } \text{“}=\text{”}, \\ \quad \left\{ \begin{array}{l} d_i(x_i, q) = 0 \\ \forall l = 1, .., N-1, n_{i,j,k,l}(x_i, q) = 0 \\ f_{x_{i_1}}(x_i) \neq 0 \text{ or } \ldots \text{ or } f_{x_{i_N}}(x_i) \neq 0 \end{array} \right. \end{array} \right. \end{array} \right.$$

The true r-offset to X_i is obtained as the difference of the generalised r-offset $\mathcal{O}_i$ to X_i and the union of each one of the images by π of the semi-algebraic sets defined

by the following system of equations and inequalities for each point m_i of $\mathfrak{S}_i$:

$$\left\{\begin{array}{l} \exists j \in [1, s_i], \forall k \in [1, r_{i,j}], \\ \quad \left\{\begin{array}{l} f_{i,j,k}(m_i) \star_{i,j,k} 0 \\ \text{if } \star_{i,j,k} \text{ is } \text{“}=\text{”}, \\ \quad \left\{\begin{array}{l} f(m_i) = 0 \\ \forall l = 1, .., N-1, n_{i,j,k,l}(m_i, q) = 0 \\ d(m_i, q) < 0 \end{array}\right. \end{array}\right. \end{array}\right.$$

It is obvious that a true Voronoi vertex of $X_1, ..., X_{N+1}$ is a point of intersection of the true $r-$offsets to $X_1, ..., X_{N+1}$ respectively. Now, what is left to write is first, that the true Voronoi vertices of $X_1, ..., X_{N+1}$ are at the distance R from X_{N+2}, or alternatively, that the true Voronoi vertices of $X_1, ..., X_{N+1}$ belong to the true $R-$offset to X_{N+2}, and finally to evaluate the signs of the (real) values of $R-r$.

Consider the $(N+2)-$dimensional points whose first N coordinates are the coordinates of a true Voronoi vertex of $X_1, ..., X_{N+1}$, and the remaining two are the distances r between that true Voronoi vertex and $X_1, ..., X_{N+1}$, and R between that true Voronoi vertex and X_{N+2}. The Delaunay graph conflict locator output list will not be empty if, and only if, $R-r<0$ for one of these $(N+2)-$dimensional points.

6.2 The formulation of the Delaunay graph conflict locator with ALIAS

All these equations and the final inequality can be written in a Maple [CGGL92] file (see Maple program in Appendix D). We have tested different solving procedures as well as different ALIAS parameters for computing the Delaunay graph conflict locator for semi-algebraic sets on conics. We first observed better results with the 3B consistency method (ALIAS/3B=1) than without (ALIAS/3B=0), both in terms of running time and in terms of number of searched boxes. Consequently, we tested only 3B based searching techniques. We can either apply the 3B consist-

ency method on all the equations and/or inequalities, or we can specify a subset of equations and/or inequalities on which the 3B consistency method will be applied (the ALIAS/subeq3B list variable). Typically, one chooses the algebraic equations and inequalities with lowest degree for the SubEq3B subset. We can also select a maximal interval range (ALIAS/Max3B), that is the maximal range that the variables intervals should have in order for the 3B method to start being applied. By setting this maximum interval range to the maximum variable interval range, we can force the 3B method to be applied from the starting variables intervals. We can also select the tolerance interval range for the 3B method, by setting the ALIAS/Delta3B variable. Some testing of these ALIAS bisection modes was done on the same example of ellipses as in Chapter 5 (see Figure 5.5.1). The results are summarised in Table 6.1. The executable was run always on the same machine at off-peak hours (in the evening). The significant results is that single bisection is much faster than mixed bisection, which in turn is faster than full bisection. These running times illustrate the impact of the exponential growth of the number of boxes with the number of variables on the running times. The 3B consistency method improves the results obtained by single bisection. There is a trade-off between the additional time required by the additional 3B interval evaluations and the reduction of the variable intervals by the 3B method. The optimum in the example of Table 6.1 is to start the 3B method after the variable intervals have been reduced by 4 by the normal single bisection process.

We have tested the full system of equations and inequalities corresponding to the Delaunay graph conflict locator as well as the partial system for identifying the generalised Voronoi vertices that are true or computing the conflict locator assuming we know which generalised Voronoi vertices are true Voronoi vertices. Note that we need to execute the program $N + 2$ times instead of one if we use the partial system ($N+1$ times for identifying the true Voronoi vertices and once for computing the Delaunay graph conflict locator assuming the true Voronoi vertices have been

Bisection process	Running time for optimised GradientSolve
Full bisection	2 h 34 min 42 s
Mixed bisection (half the variables)	26 min 26 s
Single bisection + 3B	2 min 26 s
+ 3B with half Max3B	2 min 8 s
+ 3B with one quarter Max3B	2 min 2 s
+ 3B with one sixth Max3B	4 min 49 s
+ 3B with one eight Max3B	4 min 49 s

Table 6.1: Some running time results with different ALIAS parameters on the system with the generalised offsets (see Table 6.4)

Running time	without subEq3B	with subEq3B
optimised General Solve	6 min 38 s	6 min 26 s
non-optimised General Solve		
optimised Gradient Solve	2 h 56 min 10 s	2 h 55 min 4 s
non-optimised Gradient Solve		
optimised Hessian Solve	20 h 17 min 42 s	20 h 19 min 33 s
non-optimised Hessian Solve		

Table 6.2: Some running time results for ellipses with the equations of the original curves

identified). The results are summarised in Tables 6.2 and 6.3.

While there is a time disadvantage in running the solver on the full system with respect to running the solver (four times) on the partial systems for the Gradient and Hessian based solvers, running the General solver on the full system is less time consuming. What is important to observe at this point is that if the full system is not used, in addition to running four times the solver, we need to check that the coordinates and local distance to the defining semi-algebraic sets of each generalised Voronoi vertex corresponding to the solutions correspond to a true Voronoi vertex. The full system has all the constraints (inequalities in it). This is likely why the full system can be solved faster than the four partial systems. Moreover, the General solver is the fastest one on the full and partial systems based on the equations of the original curves. Finally, specifying a subset of equations and/or inequalities

Running time	without subEq3B	with subEq3B
optimised General Solve	2 min 51 s	2 min 38 s
non-optimised General Solve		
optimised Gradient Solve	26 min 52 s	27 min 8 s
non-optimised Gradient Solve		
optimised Hessian Solve	1 h 43 min 42 s	1 h 45 min 38 s
non-optimised Hessian Solve		1 h 49 min 9 s

Table 6.3: Some running time results for ellipses with the equations of the original curves for the partial system

(ALIAS/subeq3B) on which the 3B bisection process will be applied improves the running time of the general solver while increases those of the gradient and hessian based solvers.

The general solver seems to be more efficient on the full system based on the equations specifying the semi-algebraic sets because the computations of the gradient and of the hessian are more time consuming when the number of variables increases. A way to improve the running time is to improve the computations of the intervals. What we have done is to use the parser to convert the Maple [CGGL92] expressions into C++ code that uses the ALIAS C++ library in order to do interval computations and identify intervals with possible solutions. The interval computations resulting from this automatised process may not be optimal. This is true because the computation of the interval taken by a function depends on the way this function is written. We have tried to optimise the way functions are written for the interval computations on the general solver code. However, this did not lead to any significant improvement on the running time of the optimised code. It looks like there is a trade-off between the running time improvement owed to the change of expression of the function and the optimisation done by the C++ compiler (g++ version 2.95.2).

Running time	without subEq3B	with subEq3B
optimised General Solve	12 min 37 s	12 min 56 s
non-optimised General Solve		
optimised Gradient Solve	2 min 26 s	4 min 57 s
non-optimised Gradient Solve		
optimised Hessian Solve	3 min 41 s	3 min 42 s
non-optimised Hessian Solve		

Table 6.4: Some running time results for ellipses with the equations of the generalised offsets

Ratios (without/with generalised offset)	without subEq3B	with subEq3B
optimised General Solve	0.526	0.497
non-optimised General Solve		
optimised Gradient Solve	72.40	35.37
non-optimised Gradient Solve		
optimised Hessian Solve	330.6	329.6
non-optimised Hessian Solve		

Table 6.5: The ratios of the running time results without/with the equations of the generalised offsets

6.3 The hybrid symbolic/scientific computation of the Delaunay graph conflict locator for conics

In the previous section, we have presented the computation of the Delaunay graph conflict locator for semi-algebraic sets. In this section, we will present how we can use the implicit equation of the generalised offset to a conic in the ALIAS computations, and show some results that illustrate the considerable reduction of running time realised by using the equations of the generalised offsets instead of the equations of the original curves. Table 6.4 shows some results on the ellipses that were used for the tests of the previous section. These results show a decrease of running time except for the general solver with ratios varying between 35 and 331 (see Table 6.5).

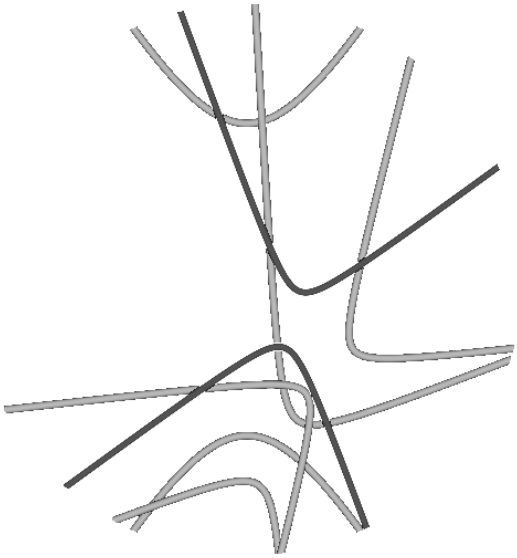

Figure 6.3.1: The test case with four hyperbolas

In order to see if the same result could be achieved with hyperbolas, we also tested the Delaunay graph conflict locator on hyperbolas. The test case we used is illustrated on Figure 6.3.1. The running times obtained using the original implicit equations of the hyperbolas are summarised in Table 6.6, while the running times obtained using the implicit equations of the generalised offsets to the hyperbolas are summarised in Table 6.7. The general solver and 3B consistency method with specification of a subset of equations to be evaluated in the 3B method on the system with the implicit equations of the generalised offsets gives the best running time with 6 minutes and 20 seconds.

Another possibility of expression of the equations of conics is the replacement of the implicit equations of conics by their parametric equations in polar coordinates[1]. This did not lead to any running time improvement with any of the solving

[1]The general polar equation of a non degenerate conic with respect to one of its foci is: $r = \frac{p}{1-e\cos\theta}$, where e is the eccentricity of the conic and p is that eccentricity times the distance from the foci to the directrix.

Running time	without subEq3B	with subEq3B
optimised General Solve	17 min 23 s	15 min 40 s
non-optimised General Solve		
optimised Gradient Solve	1h 10 min 33 s	1h 07 min 28 s
non-optimised Gradient Solve		
optimised Hessian Solve	1h 45 min 11 s	
non-optimised Hessian Solve		1h 36 min 0 s

Table 6.6: Some running time results for hyperbolas with the original equations of the hyperbolas

Running time	without subEq3B	with subEq3B
optimised General Solve	10 min 8 s	6 min 20 s
non-optimised General Solve		
optimised Gradient Solve	1 h 2 min 2 s	35 min 28 s
non-optimised Gradient Solve		
optimised Hessian Solve	2 h 12 min 17 s	
non-optimised Hessian Solve		1 h 49 min 9 s

Table 6.7: Some running time results for hyperbolas with the equations of the generalised offsets

techniques used earlier.

Chapter 7

Conclusions

The theoretical purpose of this thesis as defined in Chapter 1 is the elucidation of the basic algebraic and geometric properties of the generalised offset to a curve that are central to the Delaunay graph conflict locator. One of the practical objectives (defined in Chapter 1) is the computation of the Delaunay graph conflict locator for algebraic varieties in the case of conics. The other practical purpose of this thesis is the computation of the Delaunay graph conflict locator for low degree (semi-) algebraic sets embedded in the Euclidean plane.

We will briefly review the achievements obtained in this thesis. With respect to the theoretical objectives, we have obtained what we fixed as objectives. A general formula for the degree of the generalised offset to an algebraic curve has been presented in Section 4.2. The degree of the generalised offset to a conic and the degree of the Delaunay graph conflict locator for conics have been computed (see Sections 4.3 and 5.4 respectively). We have also reduced the problem of the computation of the Delaunay graph conflict locator from a semi-algebraic problem to a linear algebra problem (computing the eigenvalues of a matrix, as described in Chapter 5).

With respect to the practical objectives of this thesis, we have obtained what we fixed as objectives with algebraic precomputations limited to the im-

plicit equation of the generalised offset to a conic and the numerical computations done with the interval analysis based solver ALIAS. The implicit equation of the generalised offset of a conic defined by a polynomial with formal coefficients ($ax^2+by^2+cxy+dx+ey+f=0$) has been computed symbolically (see Section 4.4). The exact Delaunay graph of the Voronoi diagram for circles has been computed symbolically (see Section 3.1.3). This allows one to construct the Voronoi diagram of circles exactly. The matrix of the sparse resultant of the polynomials specifying the Delaunay graph conflict locator for conics was computed symbolically (see Section 5.4). The Schur complement of a submatrix of this matrix is the multiplication operator matrix whose eigenvalues allow one to answer the Delaunay graph conflict locator. Its computation has not been done symbolically, but numerically.

The main problem with this computation is that it takes far too much time to have a practical interest in the randomised incremental construction of the Voronoi diagram for conics. The multiplication operator matrix and of its eigenvalues were computed using Matlab [LM90]. However, this computation cannot be done with enough precision due to the bad conditioning of the matrix of the sparse resultant. The certified Delaunay graph of algebraic curves and of semi-algebraic sets defined by numeric polynomials has been computed using the interval analysis and consistency based solver "ALIAS" (see Chapter 6).

The certified computation of the Delaunay graph of conics using interval analysis gradient and hessian based solvers can benefit from the use of the implicit equation of the generalised offset to a conic (from 35 to 331 times faster, 2 min 26 s for ellipses, 6 min 20 s for hyperbolas). This result confirms the idea that knowing the structure of the set of solutions may help finding the solutions.

Even though there is no known theoretical lower nor upper bound for the interval analysis based solvers, in practice those solvers can compute the Delaunay graph conflict locator much faster than the computation of the eigenvalues of the Schur complement of a submatrix of the sparse resultant matrix. The first time

we encountered a result similar to this one was when we found in Section 3.2 that the same computation of the Delaunay graph conflict locator was taking 11 minutes with ALIAS and 12 hours using GB/RS [Fau, Rou] (by computing the Gröbner basis and the eigenvalues of the multiplication operator matrices).

Moreover, the computation of the Delaunay graph conflict locator by interval analysis based solvers is more direct than the computation of the conflict locator through eigenvalues of the Schur complement of a submatrix of the sparse resultant matrix. Indeed, we can test whether the generalised Voronoi vertices are true Voronoi vertices and the Delaunay graph conflict locator simultaneously using ALIAS. We do not have to evaluate first the conflict locator assuming the generalised Voronoi vertices are true Voronoi vertices and then test which generalised Voronoi vertices invalid with respect to the fourth conic were true Voronoi vertices of the first three conics.

Finally the computation of the Delaunay graph conflict locator by interval analysis based solvers can be used for semi-algebraic sets of arbitrary degree, which is not possible for algebraic techniques because we have reached with conics the limit on the memory that can be used in current systems (4 Gb of RAM and 6 Gb of virtual memory on Medicis machines [CNR]). This is due to the exponential complexity of the size of the matrices involved in the computations. Moreover, the computation of the Delaunay graph conflict locator by interval analysis based solvers can be easily generalised to general regular curves, which is absolutely impossible for algebraic techniques.

This thesis has presented what we believe is the first computation of the Delaunay graph conflict locator for semi-algebraic sets (and in particular conics), and its application to a semi-dynamic algorithm for the construction of the Voronoi diagram of semi-algebraic sets.

The computations involved in the certified Delaunay graph conflict locator may be required only in almost degenerate cases where the semi-algebraic set being

added touches or almost touches one or more Delaunay graph empty circles. This situation is not the most probable one, at least statistically. The algorithms presented in this thesis could be combined with simpler algorithms when the degenerate cases can be filtered.

7.1 Limitations of this research

The implicit equation of the generalised offset to a conic is quite long and complex. We could have tried to use invariants in order to try to simplify it after we obtained the implicit equation. A simplification of this implicit equation of the generalised offset to a conic would imply simpler polynomials (i.e. with fewer monomials and maybe a lower degree). Simpler implicit equations for the generalised offsets would imply that the sparse resultant matrix of the Delaunay graph conflict locator would be smaller, provided the new variables (the invariants used) allow one to identify unambiguously the generalised Voronoi vertices and their local distances to the first three conics and to the fourth conic. The problem here is to find such a set of new variables (the invariants used). The invariants of the special orthogonal group for the generalised Voronoi vertex look like the only invariants that allow one to identify unambiguously the generalised Voronoi vertices and their local distances to the first three conics and to the fourth conic.

With respect to the Computer Algebra Systems, I tested a large number of them: all the public domain ones (CoCoA [CNR00], Macaulay 2 [GS], Maxima [GG82], and Singular [GPS01]), and the most commonly commercial one used for Gröbner bases computations (Magma), as well as general Computer Algebra Systems such as Maple [CGGL92] or Mathematica [Wol99].

With regard to numerical computations, I have tried only interval analysis based methods. Other methods such as homotopy continuation [Li03] allow one to compute the solutions of systems of polynomial equations. With respect to the interval analysis based solvers, I tested only ALIAS for two reasons. The first one is

the availability of the designers of ALIAS at INRIA Sophia-Antipolis, where I made a visit of one year and half. The second one is that ALIAS is an attempt to bring together in a single library tools from interval analysis and consistency methods. There are a lot of interval analysis solvers available either on the public domain or commercially. Other alternatives could have been tested. The libraries used at the lower levels in order to perform interval evaluations for functions are however more limited in number. ALIAS uses the PROFIL/BIAS [Knü94] library for performing interval evaluations.

7.2 Future research

The future research should try to go beyond the limitations of this research as outlined in the previous section. An interesting direct generalisation worth to explore is the case of regular curves rather than semi-algebraic sets. However, the practical applications of such a generalisation does not seem to be more important than the case of semi-algebraic sets, because regular curves can be approximated to any geometrical tolerance by semi-algebraic sets. What is critical is that the Voronoi diagram and the Delaunay graph are very sensitive to the continuity of the first order and second order derivatives at contact points. This is the reason for which approximation of curves by line segments does not guarantee the exactness of the Delaunay graph. It would be useful to explore the differences in running times of the Delaunay graph conflict locator computation using regular curves or such an approximation of regular curves by semi-algebraic sets (and especially conics).

If the implicit equation of the generalised offset to a conic could be simplified by using new variables such as invariants, it would be worth trying to compute the Gröbner basis of the ideal generated by the three $r-$generalised offsets to three conics C_1, C_2 and C_3 and the $R-$generalised offset to a fourth conic C_4. Knowing the Gröbner basis of that ideal, two alternative approaches may be used in order to evaluate the conflict locator: either the linear algebra approach in the quotient

algebra (as developed in Section 3.1.3 or in Chapter 5), or the triangular set approach (as implemented in the triang library of Singular [GPS01]).

Bibliography

[ABMY02] Francçois Anton, Jean-Daniel Boissonnat, Darka Mioc, and Mariette Yvinec. An exact predicate for the optimal construction of the Additively Weighted Voronoi diagram. In *Proceedings of the European Workshop on Computational Geometry 2002, Warsaw, Poland*, 2002.

[AK00] Franz Aurenhammer and Rolf Klein. Voronoi diagrams. In *Handbook of computational geometry*, pages 201–290. North-Holland, Amsterdam, 2000.

[AKM02] François Anton, David Kirkpatrick, and Darka Mioc. An exact algebraic predicate for the maintenance of the topology of the additively weighted voronoi diagram. In *The Fourteenth Canadian Conference on Computational Geometry*, pages 72–76, Lethbridge, Alberta, Canada, 2002.

[AMG98] François Anton, Darka Mioc, and Christopher M. Gold. Dynamic Additively Weighted Voronoi diagram made easy. In *Proceedings of the 10th Canadian Conference on Computational Geometry, August 1998, Montréal, Canada*, pages 92–93, 1998.

[AS95] Helmut Alt and Otfried Schwarzkopf. The Voronoi diagram of curved objects. In *Proc. 11th ACM Symp. Computational Geometry*, pages 89–97. ACM Press, 5–7 June 1995.

[ASS99] Enrique Arrondo, Juana Sendra, and J. Rafael Sendra. Genus formula for generalized offset curves. *J. Pure Appl. Algebra*, 136(3):199–209, 1999.

[Aur87] Franz Aurenhammer. Power diagrams: properties, algorithms and applications. *SIAM J. Comput.*, 16(1):78–96, 1987.

[Baj94] Chandrajit L. Bajaj. Some applications of constructive real algebraic geometry. In *Algebraic geometry and its applications (West Lafayette, IN, 1990)*, pages 393–405. Springer, New York, 1994.

[BB96] John A. Beachy and William D. Blair. *Abstract Algebra*. Waveland Press Inc., 1996.

[BCK88] Bruno Buchberger, George E. Collins, and Bernhard Kutzler. Algebraic methods for geometric reasoning. In *Annual review of computer science, Vol. 3*, pages 85–119. Annual Reviews, Palo Alto, CA, 1988.

[BCR98] Jacek Bochnak, Michel Coste, and Marie-Françoise Roy. *Real algebraic geometry*. Springer-Verlag, Berlin, 1998. Translated from the 1987 French original, Revised by the authors.

[BCSS98] Lenore Blum, Felipe Cucker, Michael Shub, and Steve Smale. *Complexity and real computation*. Springer-Verlag, New York, 1998. With a foreword by Richard M. Karp.

[BR90] Riccardo Benedetti and Jean-Jacques Risler. *Real algebraic and semi-algebraic sets*. Hermann, Paris, 1990.

[Buc70] Bruno Buchberger. Ein algorithmisches Kriterium für die Lösbarkeit eines algebraischen Gleichungssystems. *Aequationes Math.*, 4:374–383, 1970.

[Buc79] Bruno Buchberger. A criterion for detecting unnecessary reductions in the construction of Gröbner-bases. In *Symbolic and algebraic computation (EUROSAM '79, Internat. Sympos., Marseille, 1979)*, volume 72 of *Lecture Notes in Comput. Sci.*, pages 3–21. Springer, Berlin, 1979.

[Buc88] Bruno Buchberger. Applications of Gröbner bases in nonlinear computational geometry. In *Mathematical aspects of scientific software (Minneapolis, Minn., 1986/87)*, volume 14 of *IMA Vol. Math. Appl.*, pages 59–87. Springer, New York, 1988.

[Buc92] Bruno Buchberger. Gröbner bases: an introduction. In *Automata, languages and programming (Vienna, 1992)*, volume 623 of *Lecture Notes in Comput. Sci.*, pages 378–379. Springer, Berlin, 1992.

[Buc98] Bruno Buchberger. Introduction to Gröbner bases. In *Gröbner bases and applications (Linz, 1998)*, volume 251 of *London Math. Soc. Lecture Note Ser.*, pages 3–31. Cambridge Univ. Press, Cambridge, 1998.

[CCM97] Hyeong In Choi, Sung Woo Choi, and Hwan Pyo Moon. Mathematical theory of medial axis transform. *Pacific J. Math.*, 181(1):57–88, 1997.

[CD00] Tzu-Yi Chen and James W. Demmel. Balancing sparse matrices for computing eigenvalues. In *Proceedings of the International Workshop on Accurate Solution of Eigenvalue Problems (University Park, PA, 1998)*, volume 309, pages 261–287, 2000.

[CE00] John F. Canny and Ioannis Z. Emiris. A subdivision-based algorithm for the sparse resultant. *J. ACM*, 47(3):417–451, 2000.

[CGGL92] Bruce W. Char, Keith O. Geddes, Gaston H. Gonnet, and Benton L. Leoung. *Maple V.* Springer, Berlin, 1992.

[CLO97] David Cox, John Little, and Donal O'Shea. *Ideals, varieties, and algorithms*. Springer-Verlag, New York, second edition, 1997. An introduction to computational algebraic geometry and commutative algebra.

[CLO98] David Cox, John Little, and Donal O'Shea. *Using algebraic geometry*. Springer-Verlag, New York, 1998.

[CNR] CNRS/Ecole Polytechnique, http://www.medicis.polytechnique.fr. *Unité Mixte de Services Medicis*.

[CNR00] Antonio Capani, Gianfranco Niesi, and Lorenzo Robbiano. *CoCoA, a system for doing Computations in Commutative Algebra*. Available via anonymous ftp from `cocoa.dima.unige.it`, 4.0 edition, 2000.

[CPX02] Zhenming Chen, Evanthia Papadopoulou, and Jinhui Xu. Robust algorithm for k-gon voronoi diagram construction. In *Abstracts for the Fourteenth Canadian Conference on Computational Geometry CCCG '02*, pages 77–81, Lethbridge, Alberta, Canada, August 2002. University of Lethbridge.

[Des03] Alexis Deschamps. *Handbook of Aluminum*, chapter Analytical Techniques for Aluminium Alloys, vol. 2, Alloy Production and Materials manufacturing, pages 155–192. Marcel Dekker, Inc., New York, USA, 2003.

[DMT92] Olivier Devillers, Stefan Meiser, and Monique Teillaud. The space of spheres, a geometric tool to unify duality results on Voronoi diagrams. Rapport de recherche 1620, INRIA, 1992.

[EC95] Ioannis Z. Emiris and John F. Canny. Efficient incremental algorithms for the sparse resultant and the mixed volume. *J. Symbolic Comput.*, 20(2):117–149, 1995.

[Emi96] Ioannis Z. Emiris. On the complexity of sparse elimination. *J. Complexity*, 12(2):134–166, 1996.

[Emi97] Ioannis Z. Emiris. A general solver based on sparse resultants: Numerical issues and kinematic applications. Technical Report 3110, SAFIR, 1997.

[Fau] Jean-Charles Faugère. *Gb*. Available from `http://www-calfor.lip6.fr/ jcf/index.html`.

[FJ94a] R. T. Farouki and J. K. Johnstone. Computing point/curve and curve/curve bisectors. In *Design and application of curves and surfaces (Edinburgh, 1992)*, pages 327–354. Oxford Univ. Press, New York, 1994.

[FJ94b] Rida T. Farouki and John K. Johnstone. The bisector of a point and a plane parametric curve. *Comput. Aided Geom. Design*, 11(2):117–151, 1994.

[FN90a] Rida T. Farouki and C. Andrew Neff. Algebraic properties of plane offset curves. *Comput. Aided Geom. Design*, 7(1-4):101–127, 1990. Curves and surfaces in CAGD '89 (Oberwolfach, 1989).

[FN90b] Rida T. Farouki and C. Andrew Neff. Analytic properties of plane offset curves. *Comput. Aided Geom. Design*, 7(1-4):83–99, 1990. Curves and surfaces in CAGD '89 (Oberwolfach, 1989).

[FR98a] Rida T. Farouki and Rajesh Ramamurthy. Degenerate point/curve and curve/curve bisectors arising in medial axis computations for planar domains with curved boundaries. *Comput. Aided Geom. Design*, 15(6):615–635, 1998.

[FR98b] Rida T. Farouki and Rajesh Ramamurthy. Specified-precision computation of curve/curve bisectors. *Internat. J. Comput. Geom. Appl.*, 8(5-6):599–617, 1998.

[GG82] V. Ellen Golden and Mathlab Group. *Introductory MACSYMA documentation: a collection of papers: (a) An introduction to ITS for the Macsyma user, (b) ITS easy once ITS explained, and (c) Macsyma primer*. Massachusetts Institute of Technology, Laboratory for Computer Science, Cambridge, MA, USA, revised edition, 1982.

[Giu84] Marc Giusti. Some effectivity problems in polynomial ideal theory. In *EUROSAM 84 (Cambridge, 1984)*, volume 174 of *Lecture Notes in Comput. Sci.*, pages 159–171. Springer, Berlin, 1984.

[GP02] Gert-Martin Greuel and Gerhard Pfister. *A* **Singular** *introduction to commutative algebra*. Springer-Verlag, Berlin, 2002. With contributions by Olaf Bachmann, Christoph Lossen and Hans Schönemann, With 1 CD-ROM (Windows, Macintosh, and UNIX).

[GPS01] Gert-Martin Greuel, Gerhard Pfister, and Hans Schönemann. SINGULAR 2.0. A Computer Algebra System for Polynomial Computations, Centre for Computer Algebra, University of Kaiserslautern, 2001. `http://www.singular.uni-kl.de`.

[Grö39] Wolfgang Gröbner. Über die algebraischen eigenschaften der integrale von linearen differentialgleichungen mit konstanten koeffizienten. *Monatsh. der Math.*, 1939.

[GS] Daniel R. Grayson and Michael E. Stillman. Macaulay 2, a software system for research in algebraic geometry. Available at http://www.math.uiuc.edu/Macaulay2/.

[GS85] Leonidas Guibas and Jorge Stolfi. Primitives for the manipulation of general subdivisions and the computation of voronoi. *ACM Transactions on Graphics (TOG)*, 4(2):74–123, 1985.

[Han92] Eldon Hansen. *Global optimization using interval analysis*, volume 165 of *Monographs and Textbooks in Pure and Applied Mathematics*. Marcel Dekker Inc., New York, 1992.

[Hea02] Michael T. Heath. *Scientific Computing : An Introductory Survey*. McGraw-Hill Higher Education, second edition edition, 2002.

[Hof90] Christoph M. Hoffmann. Algebraic and numerical techniques for offsets and blends. In *Computation of curves and surfaces (Puerto de la Cruz, 1989)*, pages 499–528. Kluwer Acad. Publ., Dordrecht, 1990.

[Huỳ86] Dũng T. Huỳnh. A superexponential lower bound for Gröbner bases and Church-Rosser commutative Thue systems. *Inform. and Control*, 68(1-3):196–206, 1986.

[HV91] Christoph M. Hoffmann and Pamela J. Vermeer. Eliminating extraneous solutions in curve and surface operations. *Internat. J. Comput. Geom. Appl.*, 1(1):47–66, 1991.

[Kan57] Leonid Vitalyevich Kantorovitch. On some further applications of the Newton approximation method. *Vestnik Leningrad. Univ. Ser. Mat. Meh. Astr.*, 12(7):68–103, 1957.

[KE02] Menelaos I. Karavelas and Ioannis Z. Emiris. Predicates for the Planar Additively Weighted Voronoi Diagram. ECG Technical Report ECG-TR-122201-01, INRIA, 2002.

[KKS00] Deok-Soo Kim, Donguk Kim, and Kokichi Sugihara. Voronoi diagram of a circle set constructed from Voronoi diagram of a point set. In *Al-*

gorithms and computation (Taipei, 2000), volume 1969 of *Lecture Notes in Comput. Sci.*, pages 432–443. Springer, Berlin, 2000.

[KKS01a] Deok-Soo Kim, Donguk Kim, and Kokichi Sugihara. Voronoi diagram of a circle set from Voronoi diagram of a point set. I. Topology. *Comput. Aided Geom. Design*, 18(6):541–562, 2001.

[KKS01b] Deok-Soo Kim, Donguk Kim, and Kokichi Sugihara. Voronoi diagram of a circle set from Voronoi diagram of a point set. II. Geometry. *Comput. Aided Geom. Design*, 18(6):563–585, 2001.

[Kle89] Rolf Klein. *Concrete and abstract Voronoï diagrams*. Springer-Verlag, Berlin, 1989.

[Knü94] Olaf Knüppel. PROFIL/BIAS—a fast interval library. *Computing*, 53(3-4):277–287, 1994. International Symposium on Scientific Computing, Computer Arithmetic and Validated Numerics (Vienna, 1993).

[Kol37] Andrei Nikolaevich Kolmogorov. A statistical theory for the recrystallization of metals. *Akad. nauk SSSR, Izv., Ser. Matem.*, 1(3):355–359, 1937.

[Krä92] Walter Krämer. Verified solution of eigenvalue problems with sparse matrices. In *Computational and applied mathematics, I (Dublin, 1991)*, pages 277–287. North-Holland, Amsterdam, 1992.

[Lan02] Serge Lang. *Algebra*, volume 211 of *Graduate Texts in Mathematics*. Springer-Verlag, New York, third edition, 2002.

[Li03] Tien-Yien Li. Numerical solution of polynomial systems by homotopy continuation methods. In *Handbook of numerical analysis, Vol. XI*, Handb. Numer. Anal., XI, pages 209–304. North-Holland, Amsterdam, 2003.

[LM90] Jack N. Little and Cleve B. Moler. *Matlab — User's Guide*. MathWorks, Inc., Cochinate Place, 24 Prime Park Way, Natick, MA 01760, January 1990.

[LSY98] Rich B. Lehoucq, Danny C. Sorensen, and Chao Yang. *ARPACK users' guide*. Software, Environments, and Tools. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 1998. Solution of large-scale eigenvalue problems with implicitly restarted Arnoldi methods.

[MB97] François Mercier and Olivier Baujard. Voronoi diagrams to model forest dynamics in french guiana. In *Proceedings of GeoComputation '97 & Sirc '97*, pages 161–171, University of Otago, New Zeland, August 1997.

[Mer96] Jean-Pierre Merlet. Some algebraic geometry problems arising in the field of mechanism theory. In *Algorithms in algebraic geometry and applications (Santander, 1994)*, pages 271–283. Birkhäuser, Basel, 1996.

[Mer00] Jean-Pierre Merlet. Alias: an interval analysis based library for solving and analyzing system of equations. In *SEA*, Toulouse, France, 14–16 June 2000.

[Mer01] Jean-Pierre Merlet. A parser for the interval evaluation of analytical functions and its application to engineering problems. *J. Symbolic Comput.*, 31(4):475–486, 2001.

[Mio02] Darka Mioc. *The Voronoi spatio-temporal data structure*. PhD thesis, Université Laval, 2002.

[MM84] H. Michael Möller and Ferdinando Mora. Upper and lower bounds for the degree of Groebner bases. In *EUROSAM 84 (Cambridge, 1984)*, volume 174 of *Lecture Notes in Comput. Sci.*, pages 172–183. Springer, Berlin, 1984.

[Moo77] Ramon E. Moore. A test for existence of solutions to nonlinear systems. *SIAM J. Numer. Anal.*, 14(4):611–615, 1977.

[Mou96] Bernard Mourrain. Enumeration problems in geometry, robotics and vision. In *Algorithms in algebraic geometry and applications (Santander, 1994)*, pages 285–306. Birkhäuser, Basel, 1996.

[Nik00] Jorgen L. Nikolajsen. An improved Laguerre eigensolver for unsymmetric matrices. *SIAM J. Sci. Comput.*, 22(3):822–834 (electronic), 2000.

[OBS92] Atsuyuki Okabe, Barry Boots, and Kōkichi Sugihara. *Spatial tessellations: concepts and applications of Voronoï diagrams*. John Wiley & Sons Ltd., Chichester, 1992. With a foreword by D. G. Kendall.

[ÓSY86] Colm Ó'Dúnlaing, Micha Sharir, and Chee-K. Yap. Generalized Voronoĭ diagrams for moving a ladder. I. Topological analysis. *Comm. Pure Appl. Math.*, 39(4):423–483, 1986.

[ÓSY87] Colm Ó'Dúnlaing, Micha Sharir, and Chee Yap. Generalized Voronoĭ diagrams for a ladder. II. Efficient construction of the diagram. *Algorithmica*, 2(1):27–59, 1987.

[ÓY85] Colm Ó'Dúnlaing and Chee-K. Yap. A "retraction" method for planning the motion of a disc. *J. Algorithms*, 6(1):104–111, 1985.

[Pin02] Giorgio Pini. Leftmost eigenvalue of real and complex sparse matrices on parallel computer using approximate inverse preconditioning. *Parallel Algorithms Appl.*, 17(1):41–58, 2002.

[RF99a] Rajesh Ramamurthy and Rida T. Farouki. Voronoi diagram and medial axis algorithm for planar domains with curved boundaries. I. Theoretical foundations. *J. Comput. Appl. Math.*, 102(1):119–141, 1999. Special issue: computational methods in computer graphics.

[RF99b] Rajesh Ramamurthy and Rida T. Farouki. Voronoi diagram and medial axis algorithm for planar domains with curved boundaries. II. Detailed algorithm description. *J. Comput. Appl. Math.*, 102(2):253–277, 1999.

[Rou] Fabrice Rouillier. *RS RealSolving.* Available from `http://spaces.lip6.fr/ rouillie/Softs/RS/index.php`.

[Sha94] Igor R. Shafarevich. *Basic algebraic geometry. 1.* Springer-Verlag, Berlin, second edition, 1994. Varieties in projective space, Translated from the 1988 Russian edition and with notes by Miles Reid.

[VC90] Christine Voiron-Canicio. *Analyse spatiale et analyse d'images par la morphologie mathématique.* RECLUS, 1990.

[vdV99] Henk van der Vorst. Subspace iteration methods with preconditioning for eigenvalues of very large matrices. In *Recent advances in numerical methods and applications, II (Sofia, 1998)*, pages 124–134. World Sci. Publishing, River Edge, NJ, 1999.

[Vor07] Georgiï Feodosevich Voronoï. Nouvelles applications des paramètres continus à la théorie des formes quadratiques. premier mémoire. sur quelques propriétés des formes quadratiques positives parfaites. *Journal für die reine und angewandte Mathematik*, 133:97–178, 1907.

[Vor08] Georgiï Feodosevich Voronoï. Nouvelles applications des paramètres continus à la théorie des formes quadratiques. deuxième mémoire. recherches sur les paralléloèdres primitifs. première partie. partition uniforme de l'espace analytique à n dimensions à l'aide des translations d'un même polyèdre convexe. *Journal für die reine und angewandte Mathematik*, 134:198–287, 1908.

[Vor10] Georgiï Feodosevich Voronoï. Nouvelles applications des paramètres continus à la théorie des formes quadratiques. deuxième mémoire.

recherches sur les parallélоèdres primitifs. seconde partie. domaines de formes quadratiques correspondant aux différents types de parallélоèdres primitifs. *Journal für die reine und angewandte Mathematik*, 136:67–181, 1910.

[Wol99] Stephen Wolfram. *The Mathematica book*. Cambridge University Press and Wolfram Research, Inc., New York, NY, USA and 100 Trade Center Drive, Champaign, IL 61820-7237, USA, fourth edition, 1999.

[Yap87] Chee-K. Yap. An $O(n \log n)$ algorithm for the Voronoĭ diagram of a set of simple curve segments. *Discrete Comput. Geom.*, 2(4):365–393, 1987.

Appendix A

The Macaulay 2 program for the exact Delaunay graph conflict locator for the Additively Weighted Voronoi diagram

```
gbTrace 4
dim FractionField  :=   F  ->  0
P  =  frac(QQ[a,b,c,d,e,f,g,h,i,j,k,l])
R = P[x,y,t]
cercle1 = (x-a)^2+(y-b)^2-(c+t)^2
cercle2 = (x-d)^2+(y-e)^2-(f+t)^2
cercle3 = (x-g)^2+(y-h)^2-(i+t)^2
emptycircle  =  ideal(cercle1,cercle2,cercle3)
ecgb  =  gb  emptycircle
print  ecgb
eckb  =  basis  cokernel  gens  ecgb
print  eckb
```

```
 kl  =  sort(flatten(entries(eckb)))
 kmind  =  splice {0..#kl - 1}
scan(kl,entry->print ring entry);
hashlist  =  pack(2,mingle(kl,kmind));
 feetmon  =  applyKeys(hashTable hashlist, key->toString(key));
compmat = f -> (htl=apply(kl,be->
hashTable(pack(2,mingle(apply(flatten(entries((coefficients((f*be)
%ecgb))#0)),
item  ->  feetmon#(toString(item))),flatten(entries((coefficients
((f*be)%ecgb))#1))))));
matrix(table(#kl,#kl,(i,j)->if  (htl#i)#?j  then  (htl#i)#j  else
0)));
matp2  =  compmat((x-j)^2+(y-k)^2-(t+l)^2);
m00  =  matp2_(0,0)
m01  =  matp2_(0,1)
m10  =  matp2_(1,0)
m11  =  matp2_(1,1)
cm00  =  coefficients m00
cm000  =  cm00#0
cm001  =  cm00#1
cm01  =  coefficients m01
cm010  =  cm01#0
cm011  =  cm01#1
cm10  =  coefficients m10
cm100  =  cm10#0
cm101  =  cm10#1
cm11  =  coefficients m11
cm110  =  cm11#0
```

```
cm111  =  cm11#1
```

Appendix B

An implicit equation of the generalised offset to a conic

B.1 The Maple program for calling Resin for obtaining the matrix of the sparse resultant

```
> restart;conique:=a*x^2+b*x*y+c*y^2+d*x+e*y+f;
```

$$conique := a\,x^2 + b\,x\,y + c\,y^2 + d\,x + e\,y + f$$

```
> normale:=expand(-diff(conique,y)*(u-x)+diff(conique,x)*(v-y));
```

$$normale := -b\,x\,u + b\,x^2 - 2\,c\,y\,u + 2\,c\,y\,x - e\,u + e\,x + 2\,a\,x\,v - 2\,a\,x\,y + b\,y\,v - b\,y^2 + d\,v - d\,y$$

```
> distance:=(u-x)^2+(v-y)^2-r^2;
```

$$distance := (u-x)^2 + (v-y)^2 - r^2$$

```
> collect(normale,[x,y]);
> collect(expand(X*conique+Y*distance),[x,y]);
> coeffs(collect(expand(X*conique+Y*distance),[x,y]),x,'a1');a1;
> alpha:=coeffs(collect(expand(X*conique+Y*distance),[x,y]),x)[3]
> ;coeffs(collect(expand(X*conique+Y*distance),[y,x]),y,'a2');a2;
> beta:=coeffs(collect(expand(X*conique+Y*distance),[y,x]),y)[3];
> solve({alpha=b,beta=-b},{X,Y});simplify(expand(subs(solve(
> {alpha=b,beta=-b},{X,Y}),X*conique+Y*distance))); b1:=
> denom(simplify(expand(subs(solve({alpha=b,beta=-b},
> {X,Y}),X*conique+Y*distance))))
> numer(simplify(expand(subs(solve({alpha=b,beta=-b},
> {X,Y}),X*conique+Y*distance))))\
> simplify(expand(b1*normale-numer(simplify(expand(subs(
> solve({alpha=b,beta=-b},{X,Y}),X*conique+Y*distance))))));
```

$$b\,x^2+((2\,c-2\,a)\,y+e-b\,u+2\,a\,v)\,x-b\,y^2+(-2\,c\,u-d+b\,v)\,y-e\,u+d\,v$$

$$(X\,a+Y)\,x^2+(X\,d+X\,b\,y-2\,Y\,u)\,x+(X\,c+Y)\,y^2+(-2\,Y\,v+X\,e)\,y-Y\,r^2+X\,f+Y\,u^2+Y\,v^2$$

$$(X\,c+Y)\,y^2+(-2\,Y\,v+X\,e)\,y-Y\,r^2+X\,f+Y\,u^2+Y\,v^2,\ X\,d+X\,b\,y-2\,Y\,u,\ X\,a+Y$$

$$1,\ x,\ x^2$$

$$\alpha := X\,a+Y$$

$$(X\,a+Y)\,x^2+(X\,d-2\,Y\,u)\,x-Y\,r^2+X\,f+Y\,u^2+Y\,v^2,\ -2\,Y\,v+X\,b\,x+X\,e,\ X\,c+Y$$

$$1,\ y,\ y^2$$

$$\beta := X\,c+Y$$

$$\{Y=-\frac{b\,(a+c)}{a-c},\ X=2\,\frac{b}{a-c}\}$$

$$b(a\,x^2+2\,b\,x\,y+c\,y^2+2\,d\,x+2\,e\,y+2\,f-a\,u^2+2\,a\,u\,x-a\,v^2+2\,a\,v\,y-a\,y^2+a\,r^2-c\,u^2+2\,c\,u\,x-c\,x^2-c\,v^2+2\,c\,v\,y+c\,r^2)/(a-c)$$

$$b1 := a-c$$

$$b(a\,x^2+2\,b\,x\,y+c\,y^2+2\,d\,x+2\,e\,y+2\,f-a\,u^2+2\,a\,u\,x-a\,v^2+2\,a\,v\,y-a\,y^2+a\,r^2-c\,u^2+2\,c\,u\,x-c\,x^2-c\,v^2+2\,c\,v\,y+c\,r^2)$$

$$-3\,b\,c\,v\,y-b\,c\,u\,x+b\,c\,v^2-b\,c\,r^2-b\,a\,v\,y-3\,b\,a\,u\,x-2\,b\,e\,y+b\,a\,u^2+b\,a\,v^2-b\,a\,r^2+b\,c\,u^2-2\,a\,c\,y\,u+4\,a\,c\,y\,x-2\,c\,a\,x\,v+2\,a^2\,x\,v-2\,b^2\,x\,y-2\,b\,f-c\,e\,x-c\,d\,v+c\,d\,y+2\,c^2\,y\,u-a\,e\,u-2\,c^2\,y\,x+a\,e\,x-2\,a^2\,x\,y+a\,d\,v-a\,d\,y+c\,e\,u-2\,b\,d\,x$$

```
> collect(distance,[x,y]);
> collect(expand(X2*conique+Y2*normale),[x,y]);
> coeffs(collect(expand(X2*conique+Y2*normale),[x,y]),x,'a1');a1;
> alpha:=coeffs(collect(expand(X2*conique+Y2*normale),[x,y]),x)[3
> ];coeffs(collect(expand(X2*conique+Y2*normale),[y,x]),y,'a2');
> a2;beta:=coeffs(collect(expand(X2*conique+Y2*normale),[y,x]),y)
> [3];solve({alpha=1,beta=1},{X2,Y2});simplify(expand(subs(solve(
> {alpha=1,beta=1},{X2,Y2}),X2*conique+Y2*normale)));
> b2:=denom(simplify(expand(subs(solve({alpha=1,beta=1},
> {X2,Y2}),X2*conique+Y2*normale))));
> numer(simplify(expand(subs(solve({alpha=1,beta=1},{X2,Y2}),
> X2*conique+Y2*normale))));
> simplify(expand(b2*distance-numer(simplify(expand(subs(solve(
> {alpha=1,beta=1},{X2,Y2}),X2*conique+Y2*normale))))));
```

$$u^2-2\,u\,x+x^2+v^2-2\,v\,y+y^2-r^2$$

$$(Y2\,b+X2\,a)\,x^2+((-2\,Y2\,a+2\,Y2\,c+X2\,b)\,y-Y2\,b\,u+Y2\,e+2\,Y2\,a\,v+X2\,d)\,x+(X2\,c-Y2\,b)\,y^2+(X2\,e-2\,Y2\,c\,u+Y2\,b\,v-Y2\,d)\,y+X2\,f-Y2\,e\,u+Y2\,d\,v$$

$$(X2\,c-Y2\,b)\,y^2+(X2\,e-2\,Y2\,c\,u+Y2\,b\,v-Y2\,d)\,y+X2\,f-Y2\,e\,u+Y2\,d\,v,\ (-2\,Y2\,a+2\,Y2\,c+X2\,b)\,y-Y2\,b\,u+Y2\,e+2\,Y2\,a\,v+X2\,d,\ Y2\,b+X2\,a$$

$$1,\ x,\ x^2$$

$$\alpha:=Y2\,b+X2\,a$$

$$(Y2\,b + X2\,a)\,x^2 + (-Y2\,b\,u + Y2\,e + 2\,Y2\,a\,v + X2\,d)\,x + X2\,f - Y2\,e\,u + Y2\,d\,v,$$
$$(-2\,Y2\,a + 2\,Y2\,c + X2\,b)\,x + X2\,e - 2\,Y2\,c\,u + Y2\,b\,v - Y2\,d,\ X2\,c - Y2\,b$$

$$1,\ y,\ y^2$$

$$\beta := X2\,c - Y2\,b$$

$$\{Y2 = -\frac{a-c}{b\,(a+c)},\ X2 = 2\,\frac{1}{a+c}\}$$

$$(b\,c\,v\,y - b\,c\,u\,x + b\,c\,x^2 - b\,a\,v\,y + b\,a\,u\,x + 2\,b\,e\,y + b\,a\,y^2 + 2\,a\,c\,y\,u - 4\,a\,c\,y\,x + 2\,c\,a\,x\,v - 2\,a^2\,x\,v + b\,c\,y^2 + 2\,b^2\,x\,y + 2\,b\,f + c\,e\,x + c\,d\,v - c\,d\,y - 2\,c^2\,y\,u + a\,e\,u + 2\,c^2\,y\,x - a\,e\,x + 2\,a^2\,x\,y - a\,d\,v + a\,d\,y - c\,e\,u + b\,a\,x^2 + 2\,b\,d\,x)/(b\,(a+c))$$

$$b2 := b\,(a+c)$$

$$b\,c\,v\,y - b\,c\,u\,x + b\,c\,x^2 - b\,a\,v\,y + b\,a\,u\,x + 2\,b\,e\,y + b\,a\,y^2 + 2\,a\,c\,y\,u - 4\,a\,c\,y\,x + 2\,c\,a\,x\,v - 2\,a^2\,x\,v + b\,c\,y^2 + 2\,b^2\,x\,y + 2\,b\,f + c\,e\,x + c\,d\,v - c\,d\,y - 2\,c^2\,y\,u + a\,e\,u + 2\,c^2\,y\,x - a\,e\,x + 2\,a^2\,x\,y - a\,d\,v + a\,d\,y - c\,e\,u + b\,a\,x^2 + 2\,b\,d\,x$$

$$-3\,b\,c\,v\,y - b\,c\,u\,x + b\,c\,v^2 - b\,c\,r^2 - b\,a\,v\,y - 3\,b\,a\,u\,x - 2\,b\,e\,y + b\,a\,u^2 + b\,a\,v^2 - b\,a\,r^2 + b\,c\,u^2 - 2\,a\,c\,y\,u + 4\,a\,c\,y\,x - 2\,c\,a\,x\,v + 2\,a^2\,x\,v - 2\,b^2\,x\,y - 2\,b\,f - c\,e\,x - c\,d\,v + c\,d\,y + 2\,c^2\,y\,u - a\,e\,u - 2\,c^2\,y\,x + a\,e\,x - 2\,a^2\,x\,y + a\,d\,v - a\,d\,y + c\,e\,u - 2\,b\,d\,x$$

```
> read("/cs/beta/People/Anton/ag/toric/mapl2form");
> read("/cs/beta/People/Anton/ag/toric/maplib");
```

```
Warning, the protected names norm and trace have been redefined and
unprotected
```

```
type1Array := proc(tarray, ttype)
local i, size;
   if not evalb(type(tarray, array)) then RETURN(false) end if;
   if not type(tarray, vector) then RETURN(true) end if;
   size := nops(convert(tarray, list));
   for i to size do if not evalb(type(tarray_i, ttype)) then RETURN(false)
      end if
   end do;
   RETURN(true)
end proc
```

$typeMatrix := \mathbf{proc}(tmatrix,\ ttype)$
local i;
if not evalb(type($tmatrix$, $matrix$)) **then** RETURN($false$) **end if**;
for i **to** rowdim($tmatrix$) **do**
if not evalb(type1Array(row($tmatrix$, i), $ttype$)) **then** RETURN($false$)
end if
end do;
RETURN($true$)
end proc

```
> PList:=[expand(conique),
> simplify(expand(b1*normale-numer(simplify(expand(subs(
> solve({alpha=b,beta=-b},{X,Y}),X*conique+Y*distance))))))),
> simplify(expand(b2*distance-numer(simplify(expand(subs(
> solve({alpha=1,beta=1},{X2,Y2}),
> X2*conique+Y2*normale))))))];VList:=[x,y];
```

$$PList := [a\,x^2 + b\,x\,y + c\,y^2 + d\,x + e\,y + f, -3\,b\,c\,v\,y - b\,c\,u\,x + b\,c\,v^2 - b\,c\,r^2 - b\,a\,v\,y - 3\,b\,a\,u\,x - 2\,b\,e\,y + b\,a\,u^2 + b\,a\,v^2 - b\,a\,r^2 + b\,c\,u^2 - 2\,a\,c\,y\,u + 4\,a\,c\,y\,x - 2\,c\,a\,x\,v + 2\,a^2\,x\,v - 2\,b^2\,x\,y - 2\,b\,f - c\,e\,x - c\,d\,v + c\,d\,y + 2\,c^2\,y\,u - a\,e\,u - 2\,c^2\,y\,x + a\,e\,x - 2\,a^2\,x\,y + a\,d\,v - a\,d\,y + c\,e\,u - 2\,b\,d\,x, -2\,b\,c\,v\,y - 2\,b\,c\,u\,x + b\,c\,v^2 - b\,c\,r^2 - 2\,b\,a\,v\,y - 2\,b\,a\,u\,x + b\,a\,u^2 + b\,a\,v^2 - b\,a\,r^2 + b\,c\,u^2 - Y2\,e\,x + a\,b\,x^2 - X2\,b\,x\,y + Y2\,b\,x\,u + c\,b\,y^2 + c\,b\,x^2 + a\,b\,y^2 - X2\,f + 2\,Y2\,c\,y\,u - 2\,Y2\,c\,y\,x - 2\,Y2\,a\,x\,v + 2\,Y2\,a\,x\,y - Y2\,b\,y\,v - X2\,e\,y - X2\,d\,x - X2\,c\,y^2 - X2\,a\,x^2 + Y2\,d\,y - Y2\,d\,v + Y2\,b\,y^2 + Y2\,e\,u - Y2\,b\,x^2]$$

$$VList := [x,\ y]$$

```
> sys_forC(PList,VList,u,
> `/cs/beta/People/Anton/ag/toric/offsetgeneralemtplus`,2);

writing for C program: 5 arguments; returns symb.coeffs

Writing exponents of 3 polys in 2 vars to
`/cs/beta/People/Anton/ag/toric/offsetgeneralemtplus.exps`.

written all exponents

Writing coeffile
`/cs/beta/People/Anton/ag/toric/offsetgeneralemtplus.coef` with hidden
```

```
degree=2

3 polyns in 2 variables [x, y] varslist and hidden u

wrote monoms, coeffile; return symbolic coeffs
```

$$\{c3x1 = -X2\,f + b\,c\,u^2 + b\,c\,v^2 - b\,c\,r^2 - b\,a\,r^2 + Y2\,e\,u + b\,a\,u^2 + b\,a\,v^2 - Y2\,d\,v,$$
$$c2x4 = -2\,a^2 + 4\,a\,c - 2\,c^2 - 2\,b^2,$$
$$c2x3 = -3\,b\,c\,v - 2\,b\,e + 2\,c^2\,u - b\,a\,v - a\,d - 2\,a\,c\,u + c\,d,\ c1x6 = b, c2x1 = b\,a\,v^2 + c\,e\,u + b\,c\,v^2 - b\,c\,r^2 + a\,d\,v - c\,d\,v + b\,c\,u^2 + b\,a\,u^2 - 2\,b\,f - b\,a\,r^2 - a\,e\,u,\ c2x2 = -b\,c\,u - 2\,b\,d - 2\,c\,a\,v - 3\,b\,a\,u + a\,e - c\,e + 2\,a^2\,v,\ c1x4 = c,$$
$$c1x5 = a,\ c1x2 = d,\ c1x3 = e,\ c1x1 = f,\ c3x5 = a\,b - Y2\,b + c\,b - X2\,a,$$
$$c3x6 = 2\,Y2\,a - X2\,b - 2\,Y2\,c,\ c3x4 = -X2\,c + a\,b + Y2\,b + c\,b,$$
$$c3x2 = Y2\,b\,u - 2\,b\,c\,u - Y2\,e - 2\,Y2\,a\,v - 2\,b\,a\,u - X2\,d,$$
$$c3x3 = Y2\,d - 2\,b\,c\,v + 2\,Y2\,c\,u - X2\,e - Y2\,b\,v - 2\,b\,a\,v\}$$

B.2 The Maxima program for computing the sparse resultant

```
c3x2 : a-c;

c3x3 : -a*r^2+a*u^2+a*v^2-f;

c3x4 : -2*a*u-d;

c3x5 : -2*a*v-e;

c1x2 : a;

c1x3 : c;

c1x4 : f;

c1x5 : d;

c1x6 : e;

c2x1 : b^2-2*a*c+2*a^2;

c2x2 : b*c+a*b;

c2x3 : -a*d*v+b*f+a*e*u;

c2x4 : b*d+a*b*u-2*a^2*v-a*e;
```

```
c2x5 : b*e-a*b*v+2*a*c*u+a*d;
c3x1 : -b;
c1x1 : b;
M : matrix(
[ c1x1, c1x2, c1x3, c1x4, c1x5, c1x6, 0, 0, 0, 0, 0, 0, 0],
[ c1x2, 0, c1x1, 0, 0, c1x5, 0, 0, 0, c1x3, c1x4, c1x6, 0],
[ c2x1, 0, c2x2, c2x3, c2x4, c2x5, 0, 0, 0, 0, 0, 0, 0],
[ 0, 0, 0, c1x6, c1x1, c1x3, c1x2, c1x4, c1x5, 0, 0, 0, 0],
[ c2x2, c2x1, 0, 0, c2x5, 0, c2x4, 0, c2x3, 0, 0, 0, 0],
[ 0, 0, 0, c2x5, c2x1, c2x2, 0, c2x3, c2x4, 0, 0, 0, 0],
[ 0, 0, c2x1, 0, 0, c2x4, 0, 0, 0, c2x2, c2x3, c2x5, 0],
[ c3x1, 0, c3x2, c3x3, c3x4, c3x5, 0, 0, 0, 0, 0, 0, 0],
[ c3x2, c3x1, 0, 0, c3x5, 0, c3x4, 0, c3x3, 0, 0, 0, 0],
[ 0, 0, 0, c3x5, c3x1, c3x2, 0, c3x3, c3x4, 0, 0, 0, 0],
[ 0, 0, c3x1, 0, 0, c3x4, 0, 0, 0, c3x2, c3x3, c3x5, 0],
[ 0, 0, 0, c1x5, c1x2, c1x1, 0, 0, 0, 0, c1x6, c1x3, c1x4],
[ 0, 0, 0, c2x4, 0, c2x1, 0, 0, 0, 0, c2x5, c2x2, c2x3]
);
p:determinant(M);
r:factor(p);
```

B.3 An implicit equation for parabolas

An implicit equation of a parabola in a system centred on its summit (the intersection of its axis of symmetry with itself) and the x axis being its axis of symmetry is: $y^2 - 2px = 0$.

An implicit equation of the $r-$generalised offset to a parabola is: $4*y^6+4*x^2*y^4-20*p*x*y^4-(12*r^2-p^2)*y^4-16*p*x^3*y^2-8*(r^2-4*p^2)*x^2*y^2+$

$4*(p*r^2-p^3)*x*y^2+2*(6*r^4-10*p^2*r^2)*y^2+16*p^2*x^4-16*(p*r^2+p^3)*x^3+4*(r^4-2*p^2*r^2+p^4)*x^2+16*(p*r^4+p^3*r^2)*x-(4*r^6+8*p^2*r^4+4*p^4*r^2).$

B.4 An implicit equation for ellipses, circles or hyperbolas

An implicit equation of the conic is: $\frac{x^2}{a^2}+\pm\frac{y^2}{b^2}-1=0$ where $\pm$ stands for $+$ in the case of an ellipse or a circle and $-$ in the case of an hyperbola.

An implicit equation of the $r-$generalised offset to an ellipse, a circle or an hyperbola is: $c^4*v^8+2*(c^4+c^3)*u^2*v^6-2*((2*c^4-c^3)*r^2+(c^4-2*c^3)*e)*v^6+(c^4+4*c^3+c^2)*u^4*v^4-2*((3*c^4-c^3+c^2)*r^2-(c^4-c^3+3*c^2)*e)*u^2*v^4+((6*c^4-6*c^3+c^2)*r^4+2*(3*c^4-5*c^3+3*c^2)*e*r^2+(c^4-6*c^3+6*c^2)*e^2)*v^4+2*(c^3+c^2)*u^6*v^2-2*((c^4-c^3+3*c^2)*r^2-(3*c^3-c^2+c)*e)*u^4*v^2+2*((3*c^4-5*c^3+3*c^2)*r^4-(2*c^4-3*c^3-3*c^2+2*c)*e*r^2+(3*c^3-5*c^2+3*c)*e^2)*u^2*v^2-2*((2*c^4-3*c^3+c^2)*r^6+(3*c^4-4*c^3+2*c^2-c)*e*r^4+(c^4-2*c^3+4*c^2-3*c)*e^2*r^2-(c^3-3*c^2+2*c)*e^3)*v^2+c^2*u^8+2*((c^3-2*c^2)*r^2+(2*c^2-c)*e)*u^6+((c^4-6*c^3+6*c^2)*r^4+2*(3*c^3-5*c^2+3*c)*e*r^2+(6*c^2-6*c+1)*e^2)*u^4-2*((c^4-3*c^3+2*c^2)*r^6-(c^4-2*c^3+4*c^2-3*c)*e*r^4-(3*c^3-4*c^2+2*c-1)*e^2*r^2-(2*c^2-3*c+1)*e^3)*u^2+((c^4-2*c^3+c^2)*r^8+2*(c^4-c^3-c^2+c)*e*r^6+(c^4+2*c^3-6*c^2+2*c+1)*e^2*r^4+2*(c^3-c^2-c+1)*e^3*r^2+(c^2-2*c+1)*e^4)$

where $c=\pm\frac{a^2}{b^2}$ and $e=-a^2$ and $\pm$ stands for $+$ in the case of an ellipse or a circle, and $-$ in the case of an hyperbola.

Appendix C

The Singular program for obtaining the matrix of the sparse resultant of the Delaunay graph conflict locator for conics

C.1 The case of parabolas

```
ring predicatr=(0,a,b,c,d,e,f,g,h,i,j,k,l,m,n,o,p,u0,u1,u2,u3,u4),
(x,y,r,s),dp;
poly offset1=4*y^6+4*x^2*y^4-20*p*x*y^4-12*r*y^4+p^2*y^4
-16*p*x^3*y^2-8*r*x^2*y^2+32*p^2*x^2*y^2+4*p*r*x*y^2
-4*p^3*x*y^2+12*r^2*y^2-20*p^2*r*y^2+16*p^2*x^4-16*
p*r*x^3-16*p^3*x^3+4*r^2*x^2-8*p^2*r*x^2+4*p^4*x^2
+16*p*r^2*x+16*p^3*r*x-4*r^3-8*p^2*r^2-4*p^4*r;
poly offset2=4*(b*x+a*y+d)^6+4*(a*x-b*y+c)^2*(b*x+a*y+d)^4
-20*o*(a*x-b*y+c)*(b*x+a*y+d)^4-12*r*(b*x+a*y+d)^4+o^2*
(b*x+a*y+d)^4-16*o*(a*x-b*y+c)^3*(b*x+a*y+d)^2
```

```
-8*r*(a*x-b*y+c)^2*(b*x+a*y+d)^2+32*o^2*(a*x-b*y+c)^2*
(b*x+a*y+d)^2+4*o*r*(a*x-b*y+c)*(b*x+a*y+d)^2
-4*o^3*(a*x-b*y+c)*(b*x+a*y+d)^2+12*r^2*(b*x+a*y+d)^2
-20*o^2*r*(b*x+a*y+d)^2+16*o^2*(a*x-b*y+c)^4
-16*o*r*(a*x-b*y+c)^3-16*o^3*(a*x-b*y+c)^3
+4*r^2*(a*x-b*y+c)^2-8*o^2*r*(a*x-b*y+c)^2
+4*o^4*(a*x-b*y+c)^2+16*o*r^2*(a*x-b*y+c)+
16*o^3*r*(a*x-b*y+c)-4*r^3-8*o^2*r^2-4*o^4*r;
poly offset3=4*(f*x+e*y+h)^6+4*(e*x-f*y+g)^2*(f*x+e*y+h)^4
-20*n*(e*x-f*y+g)*(f*x+e*y+h)^4-12*r*(f*x+e*y+h)^4
+n^2*(f*x+e*y+h)^4-16*n*(e*x-f*y+g)^3*(f*x+e*y+h)^2
-8*r*(e*x-f*y+g)^2*(f*x+e*y+h)^2+32*n^2*(e*x-f*y+g)^2*
(f*x+e*y+h)^2+4*n*r*(e*x-f*y+g)*(f*x+e*y+h)^2
-4*n^3*(e*x-f*y+g)*(f*x+e*y+h)^2+12*r^2*(f*x+e*y+h)^2
-20*n^2*r*(f*x+e*y+h)^2+16*n^2*(e*x-f*y+g)^4
-16*n*r*(e*x-f*y+g)^3-16*n^3*(e*x-f*y+g)^3
+4*r^2*(e*x-f*y+g)^2-8*n^2*r*(e*x-f*y+g)^2
+4*n^4*(e*x-f*y+g)^2+16*n*r^2*(e*x-f*y+g)+
16*n^3*r*(e*x-f*y+g)-4*r^3-8*n^2*r^2-4*n^4*r;
poly offset4=4*(j*x+i*y+l)^6+4*(i*x-j*y+k)^2*(j*x+i*y+l)^4
-20*m*(i*x-j*y+k)*(j*x+i*y+l)^4-12*s*(j*x+i*y+l)^4
+m^2*(j*x+i*y+l)^4-16*m*(i*x-j*y+k)^3*(j*x+i*y+l)^2
-8*s*(i*x-j*y+k)^2*(j*x+i*y+l)^2+32*m^2*(i*x-j*y+k)^2*
(j*x+i*y+l)^2+4*m*s*(i*x-j*y+k)*(j*x+i*y+l)^2
-4*m^3*(i*x-j*y+k)*(j*x+i*y+l)^2+12*s^2*(j*x+i*y+l)^2
-20*m^2*s*(j*x+i*y+l)^2+16*m^2*(i*x-j*y+k)^4
-16*m*s*(i*x-j*y+k)^3-16*m^3*(i*x-j*y+k)^3
+4*s^2*(i*x-j*y+k)^2-8*m^2*s*(i*x-j*y+k)^2
```

```
+4*m^4*(i*x-j*y+k)^2+16*m*s^2*(i*x-j*y+k)+
16*m^3*s*(i*x-j*y+k)-4*s^3-8*m^2*s^2-4*m^4*s;
ideal predicati=u0+u1*x+u2*y+u3*r+u4*s,offset1,offset2-offset1,
offset3-offset1,
offset4-(i^6+j^2*i^4)*offset1;
module predicatm=mpresmat(predicati,0);
```

C.2 The case of ellipses and/or hyperbolas

```
ring predicatr = (0,a,b,c,d,e,f,g,h,i,j,k,l,m,n,o,p,q,t,u,u0,u1,u2,
u3,u4,v),
(x,y,r,s),(dp(4),C);
poly offset1=a^4*y^8+2*(a^4+a^3)*x^2*y^6-2*((2*a^4-a^3)*r
+(a^4-2*a^3)*b)*y^6+(a^4+4*a^3+a^2)*x^4*y^4
-2*((3*a^4-a^3+a^2)*r-(a^4-a^3+3*a^2)*b)*x^2*y^4
+((6*a^4-6*a^3+a^2)*r^2+2*(3*a^4-5*a^3+3*a^2)*b*r
+(a^4-6*a^3+6*a^2)*b^2)*y^4+2*(a^3+a^2)*x^6*y^2
-2*((a^4-a^3+3*a^2)*r-(3*a^3-a^2+a)*b)*x^4*y^2
+2*((3*a^4-5*a^3+3*a^2)*r^2-(2*a^4-3*a^3-3*a^2+2*a)*b*r
+(3*a^3-5*a^2+3*a)*b^2)*x^2*y^2-2*((2*a^4-3*a^3+a^2)*r^3
+(3*a^4-4*a^3+2*a^2-a)*b*r^2+(a^4-2*a^3+4*a^2-3*a)*b^2*r
-(a^3-3*a^2+2*a)*b^3)*y^2+a^2*x^8+2*((a^3-2*a^2)*r
+(2*a^2-a)*b)*x^6+((a^4-6*a^3+6*a^2)*r^2
+2*(3*a^3-5*a^2+3*a)*b*r+(6*a^2-6*a+1)*b^2)*x^4
-2*((a^4-3*a^3+2*a^2)*r^3-(a^4-2*a^3+4*a^2-3*a)*b*r^2
-(3*a^3-4*a^2+2*a-1)*b^2*r-(2*a^2-3*a+1)*b^3)*x^2
+((a^4-2*a^3+a^2)*r^4+2*(a^4-a^3-a^2+a)*b*r^3
+(a^4+2*a^3-6*a^2+2*a+1)*b^2*r^2+2*(a^3-a^2-a+1)*b^3*r
+(a^2-2*a+1)*b^4);
```

```
poly offset2=c^4*(f*x+e*y+h)^8+2*(c^4+c^3)*(e*x-f*y+g)^2*
(f*x+e*y+h)^6-2*((2*c^4-c^3)*r+(c^4-2*c^3)*d)*(f*x+e*y+h)^6+
(c^4+4*c^3+c^2)*(e*x-f*y+g)^4*(f*x+e*y+h)^4-2*((3*c^4-c^3+c^2)*r-
(c^4-c^3+3*c^2)*d)*(e*x-f*y+g)^2*(f*x+e*y+h)^4+((6*c^4-6*c^3+c^2)*r^2
+2*(3*c^4-5*c^3+3*c^2)*d*r+(c^4-6*c^3+6*c^2)*d^2)*(f*x+e*y+h)^4
+2*(c^3+c^2)*(e*x-f*y+g)^6*(f*x+e*y+h)^2-2*((c^4-c^3+3*c^2)*r
-(3*c^3-c^2+c)*d)*(e*x-f*y+g)^4*(f*x+e*y+h)^2
+2*((3*c^4-5*c^3+3*c^2)*r^2-(2*c^4-3*c^3-3*c^2+2*c)*d*r
+(3*c^3-5*c^2+3*c)*d^2)*(e*x-f*y+g)^2*(f*x+e*y+h)^2
-2*((2*c^4-3*c^3+c^2)*r^3+(3*c^4-4*c^3+2*c^2-c)*d*r^2
+(c^4-2*c^3+4*c^2-3*c)*d^2*r-(c^3-3*c^2+2*c)*d^3)*(f*x+e*y+h)^2
+c^2*(e*x-f*y+g)^8+2*((c^3-2*c^2)*r+(2*c^2-c)*d)*(e*x-f*y+g)^6
+((c^4-6*c^3+6*c^2)*r^2+2*(3*c^3-5*c^2+3*c)*d*r+(6*c^2-6*c+1)*d^2)*
(e*x-f*y+g)^4-2*((c^4-3*c^3+2*c^2)*r^3-(c^4-2*c^3+4*c^2-3*c)*d*r^2
-(3*c^3-4*c^2+2*c-1)*d^2*r-(2*c^2-3*c+1)*d^3)*(e*x-f*y+g)^2
+((c^4-2*c^3+c^2)*r^4+2*(c^4-c^3-c^2+c)*d*r^3+
(c^4+2*c^3-6*c^2+2*c+1)*d^2*r^2+2*(c^3-c^2-c+1)*d^3*r+
(c^2-2*c+1)*d^4);
poly offset3=i^4*(l*x+k*y+n)^8+2*(i^4+i^3)*(k*x-l*y+m)^2*
(l*x+k*y+n)^6-2*((2*i^4-i^3)*r+(i^4-2*i^3)*j)*(l*x+k*y+n)^6
+(i^4+4*i^3+i^2)*(k*x-l*y+m)^4*(l*x+k*y+n)^4
-2*((3*i^4-i^3+i^2)*r-(i^4-i^3+3*i^2)*j)*(k*x-l*y+m)^2*(l*x+k*y+n)^4
+((6*i^4-6*i^3+i^2)*r^2+2*(3*i^4-5*i^3+3*i^2)*j*r
+(i^4-6*i^3+6*i^2)*j^2)*(l*x+k*y+n)^4+2*(i^3+i^2)*(k*x-l*y+m)^6*
(l*x+k*y+n)^2-2*((i^4-i^3+3*i^2)*r-(3*i^3-i^2+i)*j)*(k*x-l*y+m)^4*
(l*x+k*y+n)^2+2*((3*i^4-5*i^3+3*i^2)*r^2-(2*i^4-3*i^3-3*i^2+2*i)*j*r
+(3*i^3-5*i^2+3*i)*j^2)*(k*x-l*y+m)^2*(l*x+k*y+n)^2
-2*((2*i^4-3*i^3+i^2)*r^3+(3*i^4-4*i^3+2*i^2-i)*j*r^2
```

```
+(i^4-2*i^3+4*i^2-3*i)*j^2*r-(i^3-3*i^2+2*i)*j^3)*(l*x+k*y+n)^2
+i^2*(k*x-l*y+m)^8+2*((i^3-2*i^2)*r+(2*i^2-i)*j)*(k*x-l*y+m)^6
+((i^4-6*i^3+6*i^2)*r^2+2*(3*i^3-5*i^2+3*i)*j*r+(6*i^2-6*i+1)*j^2)*
(k*x-l*y+m)^4-2*((i^4-3*i^3+2*i^2)*r^3-(i^4-2*i^3+4*i^2-3*i)*j*r^2
-(3*i^3-4*i^2+2*i-1)*j^2*r-(2*i^2-3*i+1)*j^3)*(k*x-l*y+m)^2
+((i^4-2*i^3+i^2)*r^4+2*(i^4-i^3-i^2+i)*j*r^3
+(i^4+2*i^3-6*i^2+2*i+1)*j^2*r^2+2*(i^3-i^2-i+1)*j^3*r
+(i^2-2*i+1)*j^4);
poly offset4=o^4*(t*x+q*y+v)^8+2*(o^4+o^3)*(q*x-t*y+u)^2*
(t*x+q*y+v)^6-2*((2*o^4-o^3)*s+(o^4-2*o^3)*p)*(t*x+q*y+v)^6
+(o^4+4*o^3+o^2)*(q*x-t*y+u)^4*(t*x+q*y+v)^4-2*((3*o^4-o^3+o^2)*s
-(o^4-o^3+3*o^2)*p)*(q*x-t*y+u)^2*(t*x+q*y+v)^4+((6*o^4-6*o^3+o^2)*
s^2+2*(3*o^4-5*o^3+3*o^2)*p*s+(o^4-6*o^3+6*o^2)*p^2)*(t*x+q*y+v)^4
+2*(o^3+o^2)*(q*x-t*y+u)^6*(t*x+q*y+v)^2-2*((o^4-o^3+3*o^2)*s
-(3*o^3-o^2+o)*p)*(q*x-t*y+u)^4*(t*x+q*y+v)^2+
2*((3*o^4-5*o^3+3*o^2)*s^2-(2*o^4-3*o^3-3*o^2+2*o)*p*s
+(3*o^3-5*o^2+3*o)*p^2)*(q*x-t*y+u)^2*(t*x+q*y+v)^2
-2*((2*o^4-3*o^3+o^2)*s^3+(3*o^4-4*o^3+2*o^2-o)*p*s^2
+(o^4-2*o^3+4*o^2-3*o)*p^2*s-(o^3-3*o^2+2*o)*p^3)*(t*x+q*y+v)^2
+o^2*(q*x-t*y+u)^8+2*((o^3-2*o^2)*s+(2*o^2-o)*p)*(q*x-t*y+u)^6
+((o^4-6*o^3+6*o^2)*s^2+2*(3*o^3-5*o^2+3*o)*p*s
+(6*o^2-6*o+1)*p^2)*(q*x-t*y+u)^4-2*((o^4-3*o^3+2*o^2)*s^3
-(o^4-2*o^3+4*o^2-3*o)*p*s^2-(3*o^3-4*o^2+2*o-1)*p^2*s
-(2*o^2-3*o+1)*p^3)*(q*x-t*y+u)^2+((o^4-2*o^3+o^2)*s^4
+2*(o^4-o^3-o^2+o)*p*s^3+(o^4+2*o^3-6*o^2+2*o+1)*p^2*s^2
+2*(o^3-o^2-o+1)*p^3*s+(o^2-2*o+1)*p^4);
ideal pideal=u0+u1*x+u2*y+u3*r+u4*s,offset1,
(a^4-2*a^3+a^2)*offset2-(c^4-2*c^3+c^2)*offset1,
```

```
(a^4-2*a^3+a^2)*offset3-(i^4-2*i^3+i^2)*offset1,
(a^2)*offset4-(o^4*q^4*t^4+2*o^4*q^2*t^6+o^4*t^8+2*o^3*q^6*t^2
+4*o^3*q^4*t^4+2*o^3*q^2*t^6+o^2*q^8+2*o^2*q^6*t^2+o^2*q^4*t^4)*
offset1;
module pmodule=mpresmat(pideal,0);
```

Appendix D

A Maple program for the ALIAS based Delaunay graph conflict locator for semi-algebraic sets

After having specified the equations and inequality, we need to specify the set of equations (EQ) and the set of unknowns (VAR) of the system of equations and inequalities. Then, we need to specify the search intervals for all the variables (IN). The Maple program must specify where the ALIAS Maple library is located (by setting the libname variable if it is not located in the standard Maple library location), where the ALIAS C++ library is located (by setting the ALIAS/Lib variable) as well as where the BIAS/Profil C++ library is located (by setting the ALIAS/Profil variable). Then, some ALIAS parameters (prefixed with ALIAS) can be specified (debugging: ALIAS/debug, bisection behaviour: ALIAS/single_bisection, optimisation: ALIAS/optimised, 3B parameters: ALIAS/3B, ALIAS/Max3B, ALIAS/Delta3B). Then, we issue the parser command that will convert the previous code into C++ code that uses the C++ ALIAS library (for example GradientSolve in the case of

the Maple program shown here).

D.1 The system without the implicit equations of the generalised offsets

```
OA := subs(alpha=-3,beta=2,x1^2/beta+y1^2/(beta/alpha)-1);
OB := subs(u=5*x2-3*y2+7,v=2*x2+3*y2+4,alpha=5,beta=4,
u^2/beta+v^2/(beta/alpha)-1);
OC := subs(u=3*x3-2*y3+1,v=3*x3+2*y3+4,alpha=2,beta=1,
u^2/beta+v^2/(beta/alpha)-1);
NA := -diff(OA,y1)*(x-x1)+diff(OA,x1)*(y-y1);
NB := -diff(OB,y2)*(x-x2)+diff(OB,x2)*(y-y2);
NC := -diff(OC,y3)*(x-x3)+diff(OC,x3)*(y-y3);
DA := (x-x1)^2+(y-y1)^2-r^2;
DB := (x-x2)^2+(y-y2)^2 -r^2;
DC := (x-x3)^2+(y-y3)^2 - r^2;
OD := subs(u=5*x4-2*y4+1,v=x4+4*y4+2,alpha=3,beta=2,
u^2/beta+v^2/(beta/alpha)-1);
ND := -diff(OD,y4)*(x-x4)+diff(OD,x4)*(y-y4);
PVD:=(x-x4)^2+(y-y4)^2-r^2<0;
TOA:=subs(x1=xt1,y1=yt1,OA);
TOB:=subs(x2=xt2,y2=yt2,OB);
TOC:=subs(x3=xt3,y3=yt3,OC);
PVA:=(x-xt1)^2+(y-yt1)^2-r^2>=0;
TNA := -diff(TOA,yt1)*(x-xt1)+diff(TOA,xt1)*(y-yt1);
TNB := -diff(TOB,yt2)*(x-xt2)+diff(TOB,xt2)*(y-yt2);
TNC := -diff(TOC,yt3)*(x-xt3)+diff(TOC,xt3)*(y-yt3);
PVB:=(x-xt2)^2+(y-yt2)^2-r^2>=0;
```

```
PVC:=(x-xt3)^2+(y-yt3)^2-r^2>=0;
EQ:=[OA,OB,OC,NA,NB,NC,DA,DB,DC,TOA,TOB,TOC,TNA,TNB,TNC,
PVA,PVB,PVC,OD,ND,PVD];
VAR:=[x1,y1,x2,y2,x3,y3,x,y,xt1,yt1,xt2,yt2,xt3,yt3,x4,y4,r];
libname:="/cs/beta/People/Anton/ag/ALIAS/ALIAS/maple",libname;
with(ALIAS);
‘ALIAS/lib‘:="/cs/beta/People/Anton/ag/ALIAS/ALIAS/Lib-solaris";
‘ALIAS/profil‘:="/cs/beta/People/Anton/ag/ALIAS/ALIAS/Profil";
IN:=[[-1000,1000]];
for i from 1 to 15 do IN:=[op(IN),[-1000,1000]]: od;
IN:=[op(IN),[0,1000]];
‘ALIAS/debug‘ := 1;
‘ALIAS/single_bisection‘:=2;
‘ALIAS/optimized‘:=0;
‘ALIAS/3B‘:=0;
‘ALIAS/SubEq3B‘:=[1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1];
‘ALIAS/Max3B‘:=2000;
‘ALIAS/Delta3B‘:=0.1;
‘ALIAS/maxkraw‘:=30;
GradientSolve(EQ,VAR,IN);
```

D.2 The system with the implicit equations of the generalised offsets

```
offsete:=alpha^4*v^8+2*alpha^4*u^2*v^6+
2*alpha^3*u^2*v^6-4*alpha^4*r*v^6
+2*alpha^3*r*v^6-2*alpha^4*beta*v^6+4*alpha^3*beta*v^6
```

```
+alpha^4*u^4*v^4+4*alpha^3*u^4*v^4+alpha^2*u^4*v^4
-6*alpha^4*r*u^2*v^4+2*alpha^3*r*u^2*v^4
-2*alpha^2*r*u^2*v^4+2*alpha^4*beta*u^2*v^4
-2*alpha^3*beta*u^2*v^4+6*alpha^2*beta*u^2*v^4
+6*alpha^4*r^2*v^4-6*alpha^3*r^2*v^4+alpha^2*r^2*v^4
+6*alpha^4*beta*r*v^4-10*alpha^3*beta*r*v^4
+6*alpha^2*beta*r*v^4+alpha^4*beta^2*v^4
-6*alpha^3*beta^2*v^4+6*alpha^2*beta^2*v^4
+2*alpha^3*u^6*v^2+2*alpha^2*u^6*v^2
-2*alpha^4*r*u^4*v^2+2*alpha^3*r*u^4*v^2
-6*alpha^2*r*u^4*v^2+6*alpha^3*beta*u^4*v^2
-2*alpha^2*beta*u^4*v^2+2*alpha*beta*u^4*v^2
+6*alpha^4*r^2*u^2*v^2-10*alpha^3*r^2*u^2*v^2
+6*alpha^2*r^2*u^2*v^2-4*alpha^4*beta*r*u^2*v^2
+6*alpha^3*beta*r*u^2*v^2+6*alpha^2*beta*r*u^2*v^2
-4*alpha*beta*r*u^2*v^2+6*alpha^3*beta^2*u^2*v^2
-10*alpha^2*beta^2*u^2*v^2+6*alpha*beta^2*u^2*v^2
-4*alpha^4*r^3*v^2+6*alpha^3*r^3*v^2-2*alpha^2*r^3*v^2
-6*alpha^4*beta*r^2*v^2+8*alpha^3*beta*r^2*v^2
-4*alpha^2*beta*r^2*v^2+2*alpha*beta*r^2*v^2
-2*alpha^4*beta^2*r*v^2+4*alpha^3*beta^2*r*v^2
-8*alpha^2*beta^2*r*v^2+6*alpha*beta^2*r*v^2
+2*alpha^3*beta^3*v^2-6*alpha^2*beta^3*v^2
+4*alpha*beta^3*v^2+alpha^2*u^8+2*alpha^3*r*u^6
-4*alpha^2*r*u^6+4*alpha^2*beta*u^6-2*alpha*beta*u^6
+alpha^4*r^2*u^4-6*alpha^3*r^2*u^4+6*alpha^2*r^2*u^4
+6*alpha^3*beta*r*u^4-10*alpha^2*beta*r*u^4
+6*alpha*beta*r*u^4+6*alpha^2*beta^2*u^4
```

```
-6*alpha*beta^2*u^4+beta^2*u^4-2*alpha^4*r^3*u^2
+6*alpha^3*r^3*u^2-4*alpha^2*r^3*u^2
+2*alpha^4*beta*r^2*u^2-4*alpha^3*beta*r^2*u^2
+8*alpha^2*beta*r^2*u^2-6*alpha*beta*r^2*u^2
+6*alpha^3*beta^2*r*u^2-8*alpha^2*beta^2*r*u^2
+4*alpha*beta^2*r*u^2-2*beta^2*r*u^2+4*alpha^2*beta^3*u^2
-6*alpha*beta^3*u^2+2*beta^3*u^2+alpha^4*r^4-2*alpha^3*r^4
+alpha^2*r^4+2*alpha^4*beta*r^3-2*alpha^3*beta*r^3
-2*alpha^2*beta*r^3+2*alpha*beta*r^3+alpha^4*beta^2*r^2
+2*alpha^3*beta^2*r^2-6*alpha^2*beta^2*r^2+2*alpha*beta^2*r^2
+beta^2*r^2+2*alpha^3*beta^3*r-2*alpha^2*beta^3*r
-2*alpha*beta^3*r+2*beta^3*r+alpha^2*beta^4
-2*alpha*beta^4+beta^4;
offset1:=simplify(subs(alpha=3,beta=-2,u=x,v=y,offsete));

 offset2:=simplify(subs(u=5*x-3*y+7,v=2*x+3*y+4,alpha=5,
beta=-4,offsete));
 offset3:=simplify(subs(u=3*x-2*y+1,v=3*x+2*y+4,alpha=2,
beta=-1,offsete));
 offset4:=simplify(subs(u=5*x-2*y+1,v=x+4*y+2,alpha=3,
beta=-2,r=R4,offsete));
 offset1v:=simplify(subs(r=R1,offset1));
 offset2v:=subs(r=R2,offset2);
 offset3v:=subs(r=R3,offset3);
 t1:=R1-r>=0;t2:=R2-r>=0;t3:=R3-r>=0;t4:=R4-r<0;
EQ:=[offset1,offset2,offset3,offset4,offset1v,offset2v,
offset3v,t1,t2,t3,t4];
 VAR:=[x,y,r,R1,R2,R3,R4];
```

```
libname:="/cs/beta/People/Anton/ag/ALIAS/ALIAS/maple",libname;
with(ALIAS):
‘ALIAS/lib‘:="/cs/beta/People/Anton/ag/ALIAS/ALIAS/Lib-solaris":
‘ALIAS/profil‘:="/cs/beta/People/Anton/ag/ALIAS/ALIAS/Profil";
‘ALIAS/lib‘:="/cs/beta/People/Anton/ag/ALIAS/ALIAS/Lib-solaris";
IN:=[[-1000,1000],[-1000,1000],[0,1000],[0,1000],[0,1000],[0,1000],
[0,1000]];
‘ALIAS/debug‘:=1;
‘ALIAS/single_bisection‘:=2;
‘ALIAS/optimized‘:=1;
‘ALIAS/3B‘:=1;
‘ALIAS/Max3B‘:=2000;
‘ALIAS/Delta3B‘:=0.1;
‘ALIAS/maxkraw‘:=30;
‘ALIAS/SubEq3B‘:=[0,0,0,0,0,0,0,1,1,1,1];
GradientSolve(EQ,VAR,IN);
```

Index