Exact Arrangements on Tori and Dupin Cyclides

Eric Berberich*
Max-Planck-Institut für Informatik

Michael Kerber†
Max-Planck-Institut für Informatik

Abstract

An algorithm and implementation is presented to compute the exact arrangement induced by arbitrary algebraic surfaces on a parametrized ring Dupin cyclide. The family of Dupin cyclides contains as a special case the torus. The intersection of an algebraic surface of degree n with a reference cyclide is represented as a real algebraic curve of bi-degree $(2n, 2n)$ in the two-dimensional parameter space of the cyclide. We use Eigenwillig and Kerber: "Exact and Efficient 2D-Arrangements of Arbitrary Algebraic Curves", SODA 2008, to compute a planar arrangement of such curves and extend their approach to obtain more asymptotic information about curves approaching the boundary of the cyclide's parameter space. With that, we can base our implementation on the general software framework by Berberich et. al.: "Sweeping and Maintaining Two-Dimensional Arrangements on Surfaces: A First Step", ESA 2007. Our contribution provides the demanded techniques to model the special geometry of surfaces intersecting a cyclide and the special topology of the reference surface of genus one. The contained implementation is complete and does not assume generic position. Our experiments show that the combinatorial overhead of the framework does not harm the efficiency of the method. Our experiments show that the overall performance is strongly coupled to the efficiency of the implementation for arrangements of algebraic plane curves.

CR Categories: F.2.2 [Theory of Computation]: Nonnumerical Algorithms and Problems—Geometrical problems and computations; J.6 [Computer Applications]: Computer-aided Engineering—Computer-aided design

Keywords: Dupin ring cyclide, torus, arrangements, surfaces, generic programming, CGAL, exact geometric computation, robustness

1 Introduction

Consider a surface S in $\mathbb{R}^3$ and a set C of curves on S. The arrangement $\mathcal{A}(C)$ is the subdivision of S into cells of dimensions zero, one, and two with respect to C. The cells are called vertices, edges, and faces, respectively.

Berberich et al. [2007b] introduced a general software framework for sweeping a set of curves on a parametric surface S. We present an implementation for the case that S is a ring Dupin cyclide and the arrangement on it is induced by intersections of S with algebraic surfaces of arbitrary degree. Our approach follows the *exact geometric computation paradigm* [Yap 2004]: it always computes the exact arrangement, undistorted by rounding errors, of the given input, and also handles all degeneracies like singular points or intersections with high multiplicity.

Dupin Cyclides have been introduced by Dupin [1822] as surfaces whose lines of curvature are all circular. One can think of a (ring) Dupin cyclide as a torus with variable tube radius. Dupin cyclides are the generalization of the "natural" geometric surfaces like planes, cylinders, cones, spheres and tori, what makes them useful for applications in solid modeling; compare [Chandru et al. 1989; Pratt 1990; Boehm 1990; Johnstone 1993; Pratt 1995].

Our algorithm is this: we follow the framework of [Berberich et al. 2007b], and perform a sweep-line algorithm [Bentley and Ottmann 1979] on the intersection curves of the Dupin cyclide with the surfaces in the parameter space of the cyclide. The primitives of the sweep are specified by an instance of the GEOMETRYTRAITS template parameter which is given by the recent work of Eigenwillig and Kerber and Wolpert [Eigenwillig et al. 2007; Eigenwillig and Kerber 2008]. With that model, one can sweep over algebraic plane curves of arbitrary degree. During the sweep constructs the arrangement, it interacts with a model of the TOPOLOGYTRAITS concept; this model controls the creation and manipulation of arrangement features at the boundary of the parameter space, i.e., identifications in our case. We implemented such a model for the case of a Dupin cyclide. The arrangement on the Dupin cyclide is represented by a doubly-connected edge-list (DCEL), where points are attached to vertices and curves are stored with edges.

Our approach is, in principle, not restricted to Dupin cyclides. It is applicable to any surface that provides a rational parameterization, as long as a suitable type for the TOPOLOGYTRAITS parameter can be provided. However, in practice, the degrees of the algebraic curves in the parameter space constitute a limit of practical usability of the approach.

Our implementation in C++ deeply benefits from generic programming capabilities, i.e., the actual behavior of a class is determined at compile-time by properly instantiating it with a model fulfilling a certain concept. In our case, we are using CGAL's[1] class template `Arrangement_on_surface_2` that expects instantiations for its template parameters GEOMETRYTRAITS and the TOPOLOGYTRAITS. For both we provide proper types.

Related work: Arrangements in the plane have been well studied during the past decades [Halperin 2004], and also quite a number of exact and efficient implementations appeared [Fogel et al. 2006]. Two-dimensional arrangements on surfaces, especially with exact implementation, became more popular recently. They form a fundamental substructure of three-dimensional arrangements and thus can also serve as a basis to construct them. Simpler examples are arrangements of geodesic arcs on a sphere [Berberich et al. 2007b], and arrangements of circular arcs on a sphere by [Cazals and Loriot 2007]. More complicated surfaces considered so far are arrangements induced by quadrics intersecting a reference quadric. Three approaches exist. The first actually computes more, namely the adjacency relationship between intersection of a set of quadrics [Dupont et al. 2007]. The other two project the intersection curves onto the xy-plane. The original work [Berberich et al. 2005a] maintains two arrangements, one for the lower part of the reference quadrics and one for its upper part; a connection between these two is missing. Instead, [Berberich et al. 2007b] introduces a small extension of the projection to simulate the parameter space of the reference quadric. This way, it benefits from the same framework that we are also applying for ring cyclides. In contrast to that

*e-mail: eric@mpi-inf.mpg.de
†e-mail: mkerber@mpi-inf.mpg.de

[1] See the project homepage: http://www.cgal.org

work, our contribution does not simulate the parameter space. The sweep is explicitly performed in parameter space. They have in common to add knowledge about what happens on the boundary. But they differ in how to do it.

Outline: We start by introducing main properties of Dupin cyclides in Section 2, in particular how they are parameterized by rational functions. Then, we show details of our implementation: in Section 3, we present the high-level structure, and refine the treatment of the GEOMETRYTRAITS in Section 4. Section 5 deals with the details of the TOPOLOGYTRAITS parameter, and shows how cases not yet covered by the software framework could also be handled. Finally, we report on our experiments in Section 6.

2 Dupin cyclides

Dupin introduced *cyclides* as surfaces whose lines of curvature are all circles [Dupin 1822]. Later, the term "cyclide" has been used for quartic surfaces with the circle at infinity as double curve [Forsyth 1912]; Dupin's cyclides have been called *Dupin cyclides* instead. In this work, we only consider Dupin cyclides and use the term *cyclide* according to Dupin's definition for shorter notation. Most of the material in this section appears more detailed in [Bühler 1995, § 1].

The maybe most intuitive way of constructing a (Dupin) cyclide goes back to Maxwell, we cite it from Boehm [1990]:

> Let a sufficiently long string be fastened at one end to one focus f of an ellipse, let the string be kept always tight while sliding smoothly over the ellipse, then the other end z sweeps out the whole surface of a cyclide Z.

Note that choosing a circle in this construction yields a torus. We will assume that Dupin cyclide is in *standard position and orientation*, i.e., the chosen base ellipse is defined by

$$(x/a)^2 + (y/b)^2 = 1, \quad a \geq b > 0.$$

The cyclide is defined uniquely by a, b, and a parameter μ that is the length of the string minus a. However, the cyclide can have self-intersections. We define $c = \sqrt{a^2 - b^2}$, which is the distance between the focus and the center of the ellipse. If $c < \mu \leq a$, we get a *ring cyclide* which is a surface without self-intersections. In other cases we either get a so-called *horned cyclide* (for $0 < \mu \leq c$), or a *spindle cyclide* (for $\mu > a$), compare [Bez 2007]. We can only handle ring cyclides in our algorithm, so we always assume that $c < \mu \leq a$ is satisfied. Figure 2.1 shows two examples.

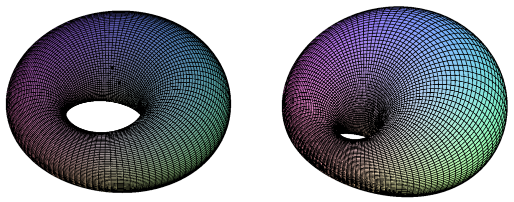

Figure 2.1: *(Left) Cyclide with $a = 1$, $b = 0.99$, $\mu = 0.5$, (Right) Cyclide with a=13, $b = 12$, $\mu = 9$*

A parameterization of the cyclide goes back to Forsyth [1912]. He also gave the following two alternatives for an implicit equation of the cyclide:

$$\begin{aligned}(x^2 + y^2 + z^2 - \mu^2 + b^2)^2 &= 4(ax - c\mu)^2 + 4b^2y^2 \\ (x^2 + y^2 + z^2 - \mu^2 - b^2)^2 &= 4(cx - a\mu)^2 - 4b^2z^2\end{aligned}$$

With these equations, it is easy to prove [Johnstone 1993] that the intersection of the cyclide with the plane $y = 0$ consists of the two circles:

$$(x + a)^2 + z^2 = (\mu + c)^2 \tag{2.1}$$
$$(x - a)^2 + z^2 = (\mu - c)^2, \tag{2.2}$$

and the intersection with $z = 0$ are the two circles

$$(x + c)^2 + y^2 = (a + \mu)^2 \tag{2.3}$$
$$(x - c)^2 + y^2 = (a - \mu)^2. \tag{2.4}$$

In our case of a ring cyclide, we always have that the interiors of (2.1) and (2.2) are disjoint, and that the circle (2.4) is contained in the interior of (2.3).

The parameterization of the cyclide is given by

$$\begin{pmatrix} \phi \\ \psi \end{pmatrix} \mapsto \begin{pmatrix} \frac{\mu(c - a\cos\phi\cos\psi) + b^2\cos\phi}{a - c\cos\phi\cos\psi} \\ \frac{b(a - \mu\cos\psi)\sin\phi}{a - c\cos\phi\cos\psi} \\ \frac{b(c\cos\phi - \mu)\sin\psi}{a - c\cos\phi\cos\psi} \end{pmatrix}$$

with $\phi, \psi \in [-\pi, \pi]$. We investigate which portion of the cyclide is parameterized at the boundaries of the parameter space:

Lemma 2.1. *If $\phi = \pi$ or ($\phi = -\pi$) is fixed, the parameterization above yields the circle $(x + a)^2 + z^2 = (\mu + c)^2$. If $\psi = \pi$ (or $\psi = -\pi$) is fixed, it yields the circle $(x + c)^2 + y^2 = (a + \mu)^2$. We call these circles* tube circle *and* outer circle*, respectively.*

Proof. Fix $\phi = \pi$. This yields the parameterization

$$\psi \mapsto \begin{pmatrix} \frac{\mu(c + a\cos\psi) - b^2}{a + c\cos\psi} \\ 0 \\ \frac{-b(c + \mu)\sin\psi}{a + c cos\psi} \end{pmatrix}$$

Since the denominator does not vanish, this parameterizes a closed path in the plane $y = 0$, so it must be one of the circles (2.1) or (2.2) By setting $\psi = \pi$, we get the point $(-\mu - c - a, 0, 0)$, so it must be circle (2.1). The same argument can be used for $\psi = \pi$. □

The tube circle and the outer circle meet in the point $p := (-\mu - c - a, 0, 0)$. We call this point the *pole* of the cyclide. Our application needs a rational parameterization of the cyclide without trigonometric functions. We use the standard trick to get rid of these functions (compare [Gallier 2001]): Using the identities

$$\cos\theta = \frac{1 - \tan^2\frac{\theta}{2}}{1 + \tan^2\frac{\theta}{2}} \quad \sin\theta = \frac{2\tan\frac{\theta}{2}}{1 + \tan^2\frac{\theta}{2}},$$

we set $u := \tan\frac{\phi}{2}, v := \tan\frac{\psi}{2}$. This yields

$$P : \mathbb{R}^2 \to \mathbb{R}^3, \begin{pmatrix} u \\ v \end{pmatrix} \mapsto$$

$$\begin{pmatrix} \frac{\mu(c(1+u^2)(1+v^2) - a(1-v^2)(1-u^2)) + b^2(1-u^2)(1+v^2)}{a(1+u^2)(1+v^2) - c(1-u^2)(1-v^2)} \\ \frac{2u(a(1+v^2) - \mu(1-v^2))b}{a(1+u^2)(1+v^2) - c(1-u^2)(1-v^2)} \\ \frac{2v(c(1-u^2) - \mu(1+u^2))b}{a(1+u^2)(1+v^2) - c(1-u^2)(1-v^2)} \end{pmatrix}$$

The image of P is the cyclide without the tube circle and the outer circle. By setting $\phi = \pi$ (or $\psi = \pi$) and applying the same trick, we also obtain rational parameterizations of the tube circle (of the outer circle). Of course, we also get them by taking the limit of P when $u \to \infty$ ($v \to \infty$).

Intuitively, this parameterization cuts the cyclide along the outer circle and the tube circle, and "rolls out" the cyclide to the plane.

Therefore, we call the outer circle and the tube circle the *cut circles* of the cyclide.

We also use the homogeneous parameterization of the cyclide, where the denominator is written as a separate variable. Define $u_+ := 1 + u^2$, $u_- := 1 - u^2$, $v_+ := 1 + v^2$ and $v_- := 1 - v^2$:

$$\hat{P} : \mathbb{R}^2 \to \mathbb{R}^4, \begin{pmatrix} u \\ v \end{pmatrix} \mapsto \begin{pmatrix} \mu(cu_+v_+ - au_-v_-) + b^2u_-v_+ \\ 2u(av_+ - \mu v_-)b \\ 2v(cu_- - \mu u_+)b \\ au_+v_+ - cu_-v_- \end{pmatrix}$$

Here are the homogeneous parameterization for the tube circle

$$\hat{PT} : \quad \mathbb{R} \to \mathbb{R}^4, \quad v \mapsto \begin{pmatrix} \mu(cv_+ + av_-) - b^2v_+ \\ 0 \\ -2v(c+\mu)b \\ av_+ + cv_- \end{pmatrix}$$

and the outer circle

$$\hat{PO} : \quad \mathbb{R} \to \mathbb{R}^4, \quad u \mapsto \begin{pmatrix} \mu(cu_+ + au_-) + b^2u_- \\ 2u(a+\mu)b \\ 0 \\ au_+ + cu_- \end{pmatrix}.$$

We will also use the following homogeneous representation of the pole. Note that $\hat{p}$ indeed represents p, since $b^2 = a^2 - c^2$:

$$\hat{p} := \begin{pmatrix} -\mu(a-c) - b^2 \\ 0 \\ 0 \\ a-c \end{pmatrix}$$

3 Our implementation

We use the software framework presented by Berberich et. al. [2007b] as part of CGAL's new `Arrangement_on_surface_2` package. It provides an arrangement class that can be used to construct, maintain, overlay, and query two-dimensional arrangements on a parametric surface. It conceptually performs a sweep in the parameter space, i.e., a line $u = u_0$ is swept to the right through the parameter space. The actual sweep-"line" is the image of $u = u_0$ on the surface S under some parametrization $P_S(u, v) = (x(u, v), y(u, v), z(u, v))$. The correct intuition for a cyclide is to sweep with a circle of variable radius along the tube of the cyclide. Observe that curves in our chosen parameter space can also approach infinity, i.e., the ones that intersect cut circles of the cyclide.

Special diligence is needed for such curves at boundaries of the parameter space. The parameter space of the cyclide contains so called *identifications* of both pairs of opposite boundaries. More precisely, $\forall v \in V, P_S(u_{\min}, v) = P_S(u_{\max}, v)$ and $\forall u \in U, P_S(u, v_{\min}) = P_S(u, v_{\max})$, so for each point on the outer- and the tube-circle there exist two pre-images (four for the pole) in parameter space, which leads to two problems for the sweep.

(1) The event queue of the sweep line algorithms needs a unique order.

(2) For the multiple pre-images of a point only one DCEL-vertex should be constructed.

CGAL's new `Arrangement_on_surface_2` package tackles these problems by modularity. To instantiate the package's main class, models of two concepts must be provided as template parameter.

GEOMETRYTRAITS: A proper instantiation for this parameter fulfills CGAL'S ARRANGEMENTTRAITS_2 concept [Wein et al. 2007]. The concept defines the types `Curve_2`, `X_monotone_curve_2`, and `Point_2` to model has to provide, and also some operations on them: Curves are split into x-monotone subcurves, points can be compared lexicographically, and the intersections of x-monotone curves are computed. The collection is operations enables a generic sweep line algorithm to compute the arrangement induced by a given set of curves. The open question is: which kind of curves do we have to handle on the cyclide, i.e., which model must be used?

We aim to represent the curves on the cyclide as algebraic curves in the two-dimensional parameter space, and compute the arrangement of these plane curves. Section 4 introduces our type for the GEOMETRYTRAITS parameter and focusses on pecularities when using it "on" the cyclide.

TOPOLOGYTRAITS: Originally, the arrangement package itself was responsible to construct and maintain all DCEL-features. This has changed with its new version, but only for features belonging to the boundary of a parameter space. For such objects, the new arrangement class interacts with the given model of the TOPOLOGYTRAITS concept. This instance is responsible to determine the underlying DCEL-representation, to create the empty representation (a bounded face for the cyclide) and to implement identifications and contractions (another kind of a special boundary). It is also the responsibility of the TOPOLOGYTRAITS instantiation to support the decisions whether to construct new faces or to create holes. We discover these and more details of our TOPOLOGYTRAITS model in Section 5.

4 Arrangements of Algebraic Plane Curves

We aim to represent the curves on the cyclide as algebraic curves in parameter space, and compute the arrangement of these plane curves. Let P denote the parameterization of the cyclide. Consider a surface of degree n, implicitly defined by $F \in \mathbb{Z}[x, y, z]$, and let $\hat{F}$ denote its homogenization.

Lemma 4.1. *The vanishing set of $f := \hat{F}(\hat{P}(u, v)) \in \mathbb{Z}[u, v]$ parameterizes the intersection points of F with the cyclide away from the cut circles.*

Proof. By definition, the vanishing set of $F(P(u, v))$ in $\mathbb{R}^2$ defines the intersection curve of F and P away from the cut circles. On the other hand, $F(P(u, v)) = 0$ if and only if $f = \hat{F}(\hat{P}(u, v)) = 0$. □

In this way, we obtain intersection curves $f_1, \ldots, f_n$ in the parameter space for the intersections of the cyclide with the surfaces $F_1, \ldots, F_n$. Our approach is to work directly in parameter space with the curves f_i, although they are of quite high degree (bidegree up to $(2 \cdot \deg F_i, 2 \cdot \deg F_i)$).[2] Therefore, we need a model of CGAL'S ARRANGEMENTTRAITS_2 concept for algebraic curves in $\mathbb{R}^2$, regardless of their degree.

Such a model has recently been provided by Eigenwillig and Kerber [Eigenwillig and Kerber 2008] based on the observation [Berberich et al. 2005b] that all required operations emerge from the topological and geometric analyses of single curves [Eigenwillig et al. 2007] and pairs of them. Combined with CGAL's `Arrangement_2` class, an implementation of the the sweep-line paradigm [Bentley and Ottmann 1979], it constitutes a robust implementation to compute arrangements of algebraic

[2] A different parameterization of the cyclide might lead to curves f_i of smaller (bi)-degree, We are neither aware of such a better parameterization, nor of a result that proves optimality of the chosen $\hat{P}$. We remark that our implementation of the traits classes is based on $\hat{P}$ and probably would not work with other parameterizations without modifications.

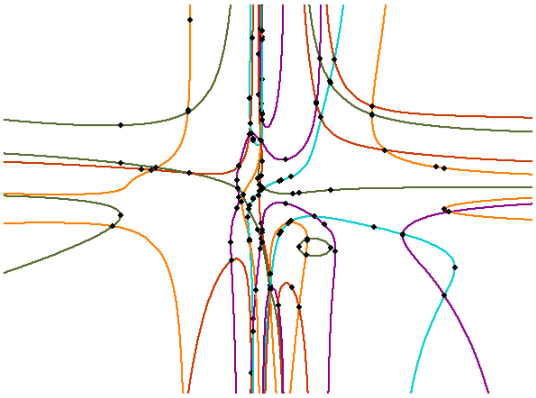

Figure 4.1: *Cut-out of an arrangement in the parameter space of a cyclide, induced by* 5 *intersecting surfaces of degree* 3

curves. No condition is imposed on the input, i.e., curves can have arbitrary degree, and contain degeneracies, like covertical intersections, vertical asymptotes and isolated points.

The main source of efficiency of that model consists in avoiding expensive symbolic computations as much as possible. Instead, it applies approximate methods for the analysis, without sacrificing the exactness of the overall result. Its main tool for this approximate computations is the *Bitstream Descartes method* [Eigenwillig 2008; Eigenwillig et al. 2005], an adaptive-precision root solver for univariate polynomials, and an extension for the case of non-square-free polynomials (m-k-Bitstream-Descartes method [Eigenwillig et al. 2007]). For symbolic computations in the algorithm, the *(signed) subresultant sequence* [Basu et al. 2006, §4] is used; it is the computation of this sequence which mainly limits the usage of the arrangement algorithm for higher degrees.

We point out that the algorithm presents the arrangement with respect to the original coordinate system, without imposing any generic condition on the input curves. For the analysis of curve and curve pairs, a linear change of coordinates might be applied if curves have vertical asymptotes or covertical critical points, but a subsequent *backshear* step translates the geometric information back to the original coordinate system.

Both the curve analysis and the curve pair analysis have been implemented in the ALCIX library which is part of EXACUS.[3] We also provide a model for CGAL's upcoming ALGEBRAICKERNELWITHANALYSIS_2 concept that can be used to instantiate CGAL's `Curved_kernel_via_analysis_2`, that will also be part in a future release. Given the analysis of curves and pairs of them, this generic implementation provides the geometric primitives required for CGAL's `Arrangement_on_surface_2` package, e.g., to run the sweep-line algorithm. The `Curved_kernel_via_analysis_2` implements the ideas shortly presented in [Berberich et al. 2005b] in CGAL.

To solve problem (1) of Section 3, the framework of [Berberich et al. 2007b] relies on a clever combination of simple comparison functors demanded from the proper model of CGAL's ARRANGEMENTTRAITS_2 concept. The main functor implements the lexicographic comparison of points that are not lying on the boundary of two-dimensional parameter space. In addition, we must provide functors to compare curve-ends approaching boundaries of the parameter space, i.e., we are asked for the horizontal or vertical alignment of two curve-ends infinitesimally away from the boundary. Their combination defines the lexicographic order of sweep-line events (i.e., "curve-ends approaching a cut circle" and "points not lying on a cut circle") on the cyclide. As we symbolically removed the cut circles from the sweep, the order of curve-ends approaching a cut circle is encoded by the order of the corresponding curve-ends in parameter space approaching infinity. Thus, we only re-interpret (and redirect) functors comparing curve-ends approaching infinity in parameter space as (to) comparison functors for curve-ends approaching a cut circle.

[3] See the project homepage: `http://www.mpi-inf.mpg.de/EXACUS`

5 Handling the identifications of the cyclide

Remember that the parameter space of the cyclide contains two identifications, one for each cut circle of the cyclide. Our solution to problem (2) of Section 3 consists of a model of the TOPOLOGYTRAITS concepts that maintains two sorted sequences of DCEL-vertices to implement these identifications. The sequences are sorted using u- and v-coordinates, and thus we call them u- and v-sequence, respectively. The coordinates are given by the horizontal and vertical asymptotes of the introduced intersection curves in parameter space. Section 5.1 focuses on how to obtain these values, especially for horizontal asymptotes. Whenever the arrangement detects a curve-end approaching a cut circle, it asks the topology traits whether a DCEL-vertex is already stored. If not, a new one is created and stored in the proper sequence; if yes, that one is used and the identification interactively takes place. A deletion is handled similarly.

The topology traits also monitors whether the insertion or deletion of a curve implies a face split or a hole creation. First, recall the planar case and consider a face in a (bounded) planar arrangement that contains a one-dimensional hole. Such a hole consists of an open sequence of curves. The surrounding face is split into two when adding a new curve closes the sequence to form a loop. The new face will be inside the originating face. Similarly, two faces are merged and a one-dimensional hole is created when a curve is removed from such a loop.

In contrast, on a cyclide, closing a loop might have different implications. We can distinguish three cases:

1. A new face is created as a hole inside the originating one (as described for the plane).

2. No new face is created, but the hole cycle is now turned to describe two outer boundary cycles of the face.

3. A face is split into two, but each cannot be understood, up to definition, as a hole inside the other.

To identify the different cases, we need the term of a perimetric path. A path of curves is *perimetric* if it crosses the identifications an odd number of times. While non-polar crossings are easy to detect, polar crossings require some special attention. Then, (1) happens if the closed path is non-perimetric, while (2) and (3) require the loop to be perimetric. (2) only occurs in a special situation, namely if the face in focus is bounded and does not have an outer boundary so far. In all other cases, closing a perimetric loop results in (3), i.e, a face bordered by two outer boundary cycles is split into two faces. Our topology traits class detects the different cases and obviously computes the crossings of a path with the identifications as a basic tool. It also assigns the resulting outer boundary cycles in case (3) correctly to the faces. For an illustration of the mentioned cases, we refer to Figure 5.1.

Beside these basic tools, each model of the TOPOLOGYTRAITS concept defines small helper classes that are used to overlay two such arrangements, to insert curves incrementally using a zone-computation, or to perform efficient point location. Our model is no exception. For the full description of the concept, we refer to [Berberich et al. 2007b] and [Berberich et al. 2007a].

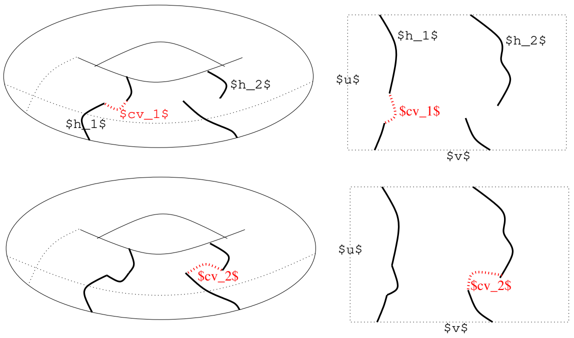

Figure 5.1: *Adding curves on cyclide (left) and in its parameter space (right): The initial arrangement consist of a single bounded face that contains two one-dimensional hole-cycles h_1, h_2 (inner boundaries). (Top) Case 2: Adding cv_1 yields that h_1 is transformed into two outer boundary-cycles of the single face. (Bottom) Case 3: Adding cv_2 splits the face into two, while two outer boundary-cycles emerge from h_2.*

5.1 Endpoints of arcs on the boundaries

Recall the u- and v-sequence that implement the identifications of the parameter space's top- and bottom-boundary, and left- and right-boundary, respectively. Functors to report the order of points on the identifications are part of the ARRANGEMENT_TRAITS_2 concept. To do so, remember that these points are points at infinity in the parameter space and assume that they are ends of unbounded arcs.

As far as explained in [Eigenwillig et al. 2007] and [Eigenwillig and Kerber 2008], the specified end of an unbounded arc is represented in two ways (if extending to infinity): Either, the arc is asymptotic to a vertical line $u = \alpha$ (i.e., it approaches the top- or bottom-boundary); in this case, the arc knows its symbolic endpoint $(\alpha, +\infty)$ or $(\alpha, -\infty)$. In this case the u-sequence can be ordered by comparing α values. Or, the arc is unbounded in u-direction (i.e., it approaches the left- or right-boundary); then its end is just represented by a fictitious vertex $-\infty_i$ or $+\infty_i$. To sort the v-sequence, this information is not sufficient, since it must know the v-value of the boundary point to which the arc converges.

We next describe how to compute whether an arc that is unbounded in u has an asymptote $v = \beta$. If so, its symbolic endpoint is either $(-\infty, \beta)$ or $(+\infty, \beta)$. This corresponds to an arc on the cyclide that intersects the tube circle at $P(\infty, \beta)$. Otherwise, the arc is unbounded also in v-direction and converges to one of the four "points" $(\pm\infty, \pm\infty)$. This corresponds to an arc on the cyclide running into the cyclide's pole. After all, each unbounded arc has a symbolic endpoint of type $(\alpha, \pm\infty)$, $(\pm\infty, \beta)$, or $(\pm\infty, \pm\infty)$, which suffices to compare the "v"-values of them.

We can concentrate on one plane curve f; the method is interactively applied to a curve in focus of the arrangement during its construction- or update-step. Horizontal asymptotes can only occur at a finite number of easily computable points, namely as roots of the leading coefficient of f with respect to u:

Lemma 5.1. *Let $f = \sum_{i=0}^{n} a_i(v)u^i \in \mathbb{Z}[u, v]$. If $v = \beta$ is a horizontal asymptote, then $a_n(\beta) = 0$.*

Proof. Assume that $a_n(\beta) \neq 0$. We show that each arc converging with its v-coordinate to β must be finite. This shows the absence of an asymptotic arc at β.

As $a_n(\beta) \neq 0$, there is a closed, finite interval I around β such that $a_n(\tilde{\beta}) \neq 0$ for all $\tilde{\beta} \in I$. By the Cauchy bound [Basu et al. 2006, Lemma 10.2], the absolute value of each real root of $f(x, \tilde{\beta})$ is smaller than $u(\tilde{\beta}) := \sum_{i=0}^{n} \left|\frac{a_i(\tilde{\beta})}{a_n(\tilde{\beta})}\right|$. Since $a_n(\tilde{\beta}) \neq 0$, this function is continuous, and thus bounded in I. Let $u_{\max}$ be the maximum. It follows that each arc which converges to β in its v-coordinates to a u-coordinate in the interval $[-u_{\max}, u_{\max}]$. ☐

Let $\beta_1 < \ldots < \beta_m$ denote the real roots of $a_n(v)$. We define $\beta_0 = -\infty$ and $\beta_{m+1} = +\infty$. For each arc of f unbounded in u-direction, we have to assign one of the points $(\pm\infty, \beta_i)$, $i = 0, \ldots, m+1$ as endpoint.

We choose rational intermediate values $q_0, \ldots, q_m$ such that $\beta_i < q_i < \beta_{i+1}$ for all $i \in \{0, \ldots, m\}$. We call the $m + 2$ intervals $(-\infty, q_0)$, (q_0, q_1),. . .,(q_{m-1}, q_m),(q_m, ∞) the *buckets*. Each buckets contains exactly one of the β_i's.

We explain our method for the left end side of the curve, it works analogously on the right: Choose a (rational) value b on the left of any *critical x-coordinate* (x-coordinates of singularities, x-extreme points or vertical asymptotes) of the curve f. Since the curve analysis of f knows about all critical x-coordinates, b is easy to compute. Next, compute

$$u_0 := \min\{b, \min_{j=0,\ldots,m} \min\{\mu \mid f(\mu, q_j) = 0\}\}$$

by isolating the real roots of $f(x, q_i)$. Isolate the real roots of $f(u_0, y)$, and determine the bucket each root falls into.

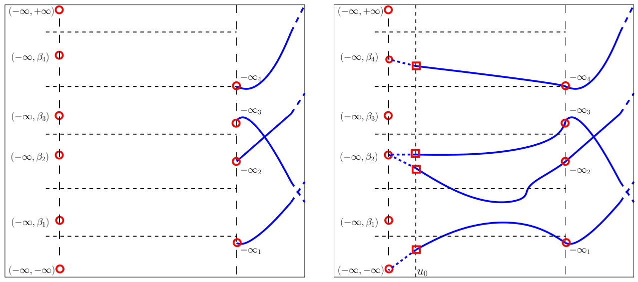

Figure 5.2: *(Left) Fictitious endpoints for the left end of the curve, and the buckets of the curve. (Right) Roots of the curve for a u-coordinate that is on the left of any bucket change. Information about horizontal asymptotes can be read off directly.*

Theorem 5.2. *Let the i-th root of $f(u_0, y)$ be in the bucket of q_j. Then, the i-th arc of f with $u \to -\infty$ converges to $(-\infty, \beta_j)$*

Proof. Since $u_0 < b$, the i-th root over $f(u_0, y)$ lies on the i-th arc of f that goes to $u = -\infty$. Moreover, u_0 is smaller than any root of $f(x, q_k)$, $k = 0, \ldots, m$. It follows that f does not intersect any line $x = q_k$ on the left of u_0. Consequently, the i-th arc of f cannot change the bucket anymore on the left of u_0. So, $(-\infty, \beta_j)$ is the only possible endpoint of the arc. ☐

5.2 Other features on the boundaries

Currently, features on the boundaries are only detected if they are incident to an arc in the arrangement, as just described. However, two types of features cannot be detected this way: First, if a surface just touches the cyclide in a point on one of the cut circles; in this case, in parameter space there is an isolated point at infinity which has no incident arc. Second, a surface might intersect the cyclide in a whole cut circle. Then, in parameter space a whole line at infinity is contained.

The current version of the framework does not take into account such features, so they are missed in the output. However, we show that they are obtainable with no additional computational effort, and thus can be integrated in a later version without worsen the performance.

Let C be a cyclide with parameterizations P whose cut circles are parameterized by PT and PO, and whose pole is p, as defined in Section 2. Consider a surface of degree n, implicitly defined by $F \in \mathbb{Z}[x, y, z]$. Again, let $f(u, v) := \hat{F}(\hat{P}(u, v))$. We show that f also encodes the intersections of F with both cut circles in its formal leading coefficients. Observe that $\deg f \leq 4n$, $\deg_u f \leq 2n$ and $\deg_v \leq 2n$.

Lemma 5.3. *Let* $\mathrm{coef}(f, x_i, j) \in \mathbb{R}[x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n]$ *denote the coefficient of* f *in* x_i^j*. Then*

$$\begin{aligned} \hat{F}(\hat{PT}(v)) &= \mathrm{coef}(f, u, 2n) \\ \hat{F}(\hat{PO}(u)) &= \mathrm{coef}(f, v, 2n) \\ \hat{F}(\hat{p}) &= \mathrm{coef}(\mathrm{coef}(f, u, 2n), v, 2n). \end{aligned}$$

Proof. Since $\mathrm{coef}(\cdot, x_i, j)$ is a linear function, it suffice to show the equality for the case that $\hat{F} = x^i y^j z^k w^l$ is a monomial with $i+j+k+l = n$. For the first part of the lemma, we can assume that $j = 0$, since for $j > 0$, $\hat{F}(\hat{PT}(u, v)) = 0$, and also, $\deg_u(f) < 2n$. Let $\hat{P}_1, \ldots, \hat{P}_4$ denote the polynomials of $\hat{P}$'s parameterization. Then, we have

$$\mathrm{coef}(f, u, 2n) = (\mathrm{coef}(P_1, u, 2))^i (\mathrm{coef}(P_3, u, 2))^k (\mathrm{coef}(P_4, u, 2))^l,$$

and comparing this with $\hat{F}(\hat{PT}(u, v))$ yields equality. The other two statements follow with similar arguments. $\square$

This lemma shows that isolated intersection points on the cut circles appear as real roots of $\mathrm{coef}(f, u, 2n)$ or $\mathrm{coef}(f, v, 2n)$. Moreover, we have the following:

Corollary 5.4.

- $\deg_u f < 2n$ *if and only if* F *and* C *intersect in the whole tube circle.*
- $\deg_v f < 2n$ *if and only if* F *and* C *intersect in the whole outer circle.*
- $\deg f < 4n$ *if and only if* F *and* C *intersect in the pole.*

We finally describe how the framework can be extended to handle such features: Recall u- and v sequence to store DCEL-vertices for intersections with cut circles. So far, these sequences were filled during the sweep whenever an infinite arc was detected. Instead, we propose to fill them before the sweep starts, by isolating the real roots of $\mathrm{coef}(f, u, 2n)$ and $\mathrm{coef}(f, v, 2n)$ for each parameter curve f, whose degree is $2n$ with respect to u, or v, respectively. Also, if any polynomial has a degree smaller than $4n$, we insert a vertex for the pole. This assures that all isolated vertices at the boundary are inserted. Moreover, if any polynomial has degree smaller than $2n$ in u, we connect consecutive vertices in the u-sequence by an edge (this form a cycle, starting and ending at the pole) to represent the tube circle. If any polynomial has degree smaller than $2n$ in v, we do the same for the v-sequence.

Note that this treatment does not cause notable extra cost in the computation: Computing the degree is trivial, and the root isolation step is performed anyway when computing asymptotic arcs of a curve f, so the result can be cached.

It is possible to extend this idea with respect to the interactive constructions and updates triggered from within the `Arrangement_on_surface_2` package. We are currently waiting for the finish of this recent extension of the framework, that naturally improves the GEOMETRYTRAITS and TOPOLOGYTRAITS concepts. Then, the splitting of combinatorial and geometrical operations is preserved, including even cases where geometric objects lie on the boundary of parameter space, i.e., isolated points and curves on an identification in the case of an cyclide.

Instance	#S	#V,#E,#F	t	t (2D)
ipl-1	10	119,190,71	**0.14**	0.14
ipl-1	20	384,682, 298	**0.58**	0.58
ipl-1	50	1837,3363,1526	**2.14**	2.00
ipl-2	10	358,575,217	**1.07**	1.25
ipl-2	20	1211,2147,937	**3.14**	3.04
ipl-3	10	542,847,305	**4.84**	4.62
ipl-3-6points	10	680,1092,412	**32.43**	31.17
ipl-3-2sing	10	694,1062,368	**5.82**	5.57
ipl-4	10	785,1204,419	**50.42**	49.97
ipl-4-6points	10	989,1529,540	**461.74**	450.54
ipl-4-2sing	10	933,1471,538	**53.01**	52.78

Table 1: *Running times (in seconds) to construct arrangements on* S_1 *induced by algebraic surfaces*

Instance	#S	#V,#E,#F	t	t (2D)
ipl-1	10	169,280,111	**0.53**	0.46
ipl-1	20	456,808,352	**0.86**	0.54
ipl-1	50	3228,6084,2856	**3.78**	3.33
ipl-2	10	450,710,260	**1.22**	1.21
ipl-2	20	1323,2247,924	**3.44**	3.57
ipl-3	10	474,682,208	**5.24**	5.36
ipl-4	10	988,1406,418	**50.93**	52.43

Table 2: *Running times (in seconds) to construct arrangements on* S_2 *induced by algebraic surfaces*

6 Results

We first observed that implementing only quite small models and relying in parallel on matured software reduces development and debugging time compared to coding a full implementation from scratch.

We also run experiments to check that this approach does not lack efficiency. All test are executed on an AMD Dual-Core Opteron(tm) 8218 multi-processor Debian Etch platform, each core equipped with 1 MB internal cache and clocked at 1 GHz. The total memory consists of 32 GB. As compiler we used `g++` in version 4.1.2 with flags `-O2 -DNDEBUG`. Two results were computed for each instance, one that computes the arrangement using the *cyclidean topology* (`onSurface`), the other is computing the two-dimensional arrangement of the induced intersection curves *in parameter space*, i.e., with the topology of an unbounded plane (`Arrangements`).

Our implementation allows to transform a cyclide in standard position and orientation, i.e., to translate it by a vector and to rotate it with respect to a rotational matrix with rational entries. In our tests, we used two different reference cyclides. First, the "standard torus" S_1 with $a = 2$, $b = 2$, $\mu = 1$, centered at the origin with no applied rotation. Second, a non-torical cyclide S_2 with $a = 13$, $b = 12$ and $\mu = 11$, centered at $(1, 1, 1)$ and a rotation defined by the matrix

$$\frac{1}{3}\begin{pmatrix} 2 & -2 & 1 \\ 2 & 1 & -2 \\ 1 & 2 & 2 \end{pmatrix}.$$

Our first class of test examples are surfaces of fixed degree which

Instances	#S	#V,#E,#F	t
quadrics	10	428,646,219	**1.59**
degree-3	5	240,314,74	**1.56**
Overlay	-	942,1508,566	**1.91**
degree-3	10	794,1218,424	**6.25**
degree-4	10	325,418,93	**13.36**
Overlay	-	1623,2644,1021	**13.83**
degree-4	10	816,1188,372	**50.86**
degree-4	5	325,418,93	**13.52**
Overlay	-	1581,2488,907	**47.30**

Table 3: *Running times (in seconds) to construct arrangements induced by algebraic surfaces of different degree on S_2, and to overlay them afterwards.*

interpolate randomly chosen points on a three-dimensional grid, having no or some degeneracies wrt S_1: the surfaces in "6points" instances share at least 6 common points on S_1, one of them is the pole of S_1. The surfaces in the "2sing" instances induce (at least) two singular intersections on S_1.

Running times are listed in Tables 1 and 2. For such random examples, our algorithm shows a good general behavior, even for higher degree surfaces. Degeneracies with respect to the reference surface result in higher running times as the instance "6points" shows. But this effect already appears in parameter space, as the last column indicates. In general, it is observable and remarkable that in all tested instances, the spent time on the cyclides is (almost) identical to the computation of the curves in their parameter space. This let us conclude two outcomes: First, the performance of our implementation is not harmed by the cyclidean topology traits, i.e., that one is as efficient as the topology traits of the unbounded plane. Second, the additionally required computation of horizontal asymptotes seems (as expected) to be a cheap task. Most time is spent for geometric operations on algebraic curves. Thus, we infer that the chosen approach strongly hinges on the efficiency of the underlying 2D-implementation for arrangements of algebraic curves and conclude the parametric ansatz to be successfull in its idea.

The implementation of a small helper class in the TOPOLOGY-TRAITS model enables CGAL's overlay mechanism, i.e., to overlay two arrangements on the same cyclide by using the capabilities of generic programming. Thus, we also generated instances of random surfaces with degree up to 4 intersecting S_2, picked two of them, computed their arrangement and finally overlaid them. A selection of such combinations along with the size of the arrangements and running times is presented in Table 3. We want to remark, that due to persistent caching, the times for the overlay are usually less than the sum of the times required to create the two originating arrangements. The reason is simply that during the overlay only some additional pairs of algebraic curves have to be newly created.

Finally, we want to remark, that we also can immediately use other techniques implemented for CGAL's `Arrangement_on_surface_2`, such as notifications, extending the DCEL by user data, or to locate a given point in the parameter space of the cyclide in an induced arrangement.

7 Conclusion

Our work demonstrates the usefulness of generic programming: the combination of the planar arrangement algorithm for arbitrary curves with the software framework for arrangement on surfaces yields an arrangement algorithm for tori and Dupin cyclides almost immediately. New code was only written for the computation of the parameterized intersection curves, for the asymptotic behavior of infinite curve arcs (Section 5.1), and for the topology traits of the cyclide. Relying on already tested and optimized code reduces the implementation effort, and makes the algorithm more robust and more efficient.

Though we propose solutions to missing parts of the framework: as already mentioned in Section 5.2, isolated features at the boundary of the parameter space are not yet handled. We are already working on the adaptation of our traits classes with respect to the extended framework that will support geometric objects on identifications.

We also believe that the performance could be further improved: the computed arrangements often contain numerous vertically asymptotic arcs (compare Figure 4). The strategy proposed in [Eigenwillig et al. 2007] to shear non-regular curves and shearing back afterwards therefore results in a change of coordinates for many curves. A comparably efficient alternative approach that avoids to shear might be more suitable for this subclass of curves.

Acknowledgements

We want to thank Ron Wein, who answered our questions on CGAL's arrangements in depth. We also thank Ophir Setter for comments on a draft of the paper.

References

ARGE, L., HOFFMANN, M., AND WELZL, E., Eds. 2007. Algorithms - ESA 2007, 15th Annual European Symp., Eilat, Israel, October 8-10, 2007, Proceedings, vol. 4698 of *LNCS*, Springer.

BASU, S., POLLACK, R., AND ROY, M.-F. 2006. *Algorithms in Real Algebraic Geometry*, 2nd ed., vol. 10 of *Algorithms and Computation in Mathematics*. Springer.

BENTLEY, J. L., AND OTTMANN, T. A. 1979. Algorithms for reporting and counting geometric intersections. *IEEE Transactions on Computers C-28*, 643–647.

BERBERICH, E., HEMMER, M., KETTNER, L., SCHÖMER, E., AND WOLPERT, N. 2005. An exact, complete and efficient implementation for computing planar maps of quadric intersection curves. In *Proc. of the 21st Annual Symp. on Computational Geometry (SCG 2005)*, 99–106.

BERBERICH, E., EIGENWILLIG, A., HEMMER, M., HERT, S., KETTNER, L., MEHLHORN, K., REICHEL, J., SCHMITT, S., SCHÖMER, E., AND WOLPERT, N. 2005. Exacus: Efficient and exact algorithms for curves and surfaces. In *Proc. of the 13th Annual European Symp. on Algorithms (ESA 2005)*, Springer, vol. 3669 of *LNCS*, 155–166.

BERBERICH, E., FOGEL, E., HALPERIN, D., MEHLHORN, K., AND WEIN, R., 2007. A general framework for processing a set of curves defined on a continuous 2D parametric surface. `http://www.cs.tau.ac.il/cgal/Projects/arr_on_surf.php`.

BERBERICH, E., FOGEL, E., HALPERIN, D., MEHLHORN, K., AND WEIN, R. 2007. Sweeping and maintaining two-dimensional arrangements on surfaces: A first step. In Arge et al. [Arge et al. 2007], 645–656.

BEZ, H. E. 2007. Rational maximal parametrisations of dupin cyclides. In *Mathematics of Surfaces XII*, Springer, R. Martin, M. Sabin, and J. Winkler, Eds., vol. 4647 of *LNCS*, 78–92.

BOEHM, W. 1990. On cyclides in geometric modeling. *Computer Aided Geometric Design 7*, 243–255.

BÜHLER, K. 1995. *Rationale algebraische Kurven auf Dupinschen Zykliden*. Master's thesis, Universität Karlsruhe. in german.

CAZALS, F., AND LORIOT, S. 2007. Computing the exact arrangement of circles on a sphere, with applications in structural biology. Technical Report 6049, INRIA Sophia-Antipolis.

CHANDRU, V., DUTTA, D., AND HOFFMANN, C. M. 1989. On the geometry of dupin cyclides. *The Visual Computer 5*, 277–290.

DUPIN, C. 1822. *Applications de Géométrie et de Méchanique*. Bachelier, Paris.

DUPONT, L., HEMMER, M., PETITJEAN, S., AND SCHÖMER, E. 2007. Complete, exact and efficient implementation for computing the adjacency graph of an arrangement of quadrics. In Arge et al. [Arge et al. 2007], 633–644.

EIGENWILLIG, A., AND KERBER, M. 2008. Exact and efficient 2d-arrangements of arbitrary algebraic curves. In *Proc. of the Nineteenth Annual ACM-SIAM Symp. on Discrete Algorithms (SODA08)*, 122–131.

EIGENWILLIG, A., KETTNER, L., KRANDICK, W., MEHLHORN, K., SCHMITT, S., AND WOLPERT, N. 2005. A Descartes algorithm for polynomials with bit-stream coefficients. In *8th International Workshop on Computer Algebra in Scientific Computing (CASC 2005)*, vol. 3718 of *LNCS*, 138–149.

EIGENWILLIG, A., KERBER, M., AND WOLPERT, N. 2007. Fast and exact geometric analysis of real algebraic plane curves. In *Proocedings of the 2007 International Symp. on Symbolic and Algebraic Computation (ISSAC 2007)*, C. W. Brown, Ed., 151–158.

EIGENWILLIG, A. 2008. *Real Root Isolation for Exact and Approximate Polynomials Using Descartes' Rule of Signs*. PhD thesis, Universität des Saarlandes, Germany.

FOGEL, E., HALPERIN, D., KETTNER, L., TEILLAUD, M., WEIN, R., AND WOLPERT, N. 2006. Arrangements. In *Effective Computational Geometry for Curves and Surfaces*, J.-D. Boissonnat and M. Teillaud, Eds. Spinger, ch. **1**, 1–66.

FORSYTH, A. 1912. *Lectures on the Differential Geometry of Curves and Surfaces*. Cambridge University Press.

GALLIER, J., 2001. Internet supplement to 'geometric methods and applications for computer science and engineering', chapter 23: Rational surfaces. http://www.cis.upenn.edu/˜jean/gbooks/geom2.html.

HALPERIN, D. 2004. Arrangements. In *Handbook of Discrete and Computational Geometry*, J. E. Goodman and J. O'Rourke, Eds., 2nd ed. Chapman & Hall/CRC, ch. 24, 529–562.

JOHNSTONE, J. K. 1993. A new intersection algorithm for cyclides and swept surfaces using cycle decomposition. *Computer Aided Geometric Design 10*, 1–24.

PRATT, M. J. 1990. Cyclides in computer aided geometric design. *Computer Aided Geometric Design 7*, 221–242.

PRATT, M. J. 1995. Cyclides in computer aided geometric design ii. *Computer Aided Geometric Design 12*, 131–152.

WEIN, R., FOGEL, E., ZUKERMAN, B., AND HALPERIN, D. 2007. 2D arrangements. In *CGAL-3.3 User and Reference Manual*.

YAP, C. K. 2004. Robust geometric computation. In *Handbook of Discrete and Computational Geometry*, J. E. Goodman and J. O'Rourke, Eds., 2nd ed. CRC Press, ch. 41, 927–952.