Efficient Distance Computation for Quadratic Curves and Surfaces

Christian Lennerz   Elmar Schömer
Max-Planck-Institute for Computer Science
66123 Saarbrücken, Germany
{lennerz, schoemer}@mpi-sb.mpg.de

Abstract

Virtual prototyping and assembly planning require physically based simulation techniques. In this setting the relevant objects are mostly mechanical parts, designed in CAD-programs. When exported to the prototyping and planning systems, curved parts are approximated by large polygonal models, thus confronting the simulation algorithms with high complexity. Algorithms for collision detection in particular are a bottleneck of efficiency and suffer from accuracy and robustness problems. To overcome these problems, our algorithm directly operates on the original CAD-data. This approach reduces the input complexity and avoids accuracy problems due to approximation errors. We present an efficient algorithm for computing the distance between patches of quadratic surfaces trimmed by quadratic curves. The distance calculation problem is reduced to the problem of solving univariate polynomials of a degree of at most 24. Moreover, we will identify an important subclass for which the degree of the polynomials is bounded by 8.

1. Introduction

During the past few years an increasing interest in virtual reality (VR) techniques could be observed. Virtual reality plays an increasingly important part in social and business life, ranging from entertainment and education to VR-based training methods and numerous industrial applications. However, the 'classical' challenges that come from concepts like virtual prototyping and assembly planning still represent strong drivers for the development of VR-techniques.

To evaluate industrial prototypes and assembly processes, virtual environments have to simulate physical behavior. Particularly collisions between moving objects must be detected and resolved. Robustness and efficiency of such a simulation heavily depend on the quality of the collision detection algorithm. Dynamic collision detection routines are required in this context. They identify the earliest time of collision in a given time interval and return the touching pair of objects. In contrast to this technique, static detection algorithms only decide whether two objects intersect at a given point in time. Since they check discrete time frames they, however, cannot guarantee to find the earliest contact.

In [9] a framework is developed in which static distance computation algorithms can be used to design dynamic detection routines. The basic idea is that information about the distance and dynamic states of all objects provides lower bounds on the earliest time of collision.

The problem of distance computation has been well studied in the past for polyhedral objects, e.g. [5, 8, 3], but only little effort has spent on objects with curved surfaces [11]. There are two reasons for this situation. First, a polygonal representation is not considered a true restriction since real objects can be approximated arbitrarily precisely by a polyhedron. The second reason is the fact that the basic algorithms and predicates can be implemented robustly and very efficiently on polygons.

However, one has to consider that there is a trade-off between the accuracy of the approximation and the efficiency of the distance computation. On the one hand, precise approximations increase the complexity of the boundary representation and can affect the running time of the algorithm disproportionately. On the other hand, a small number of polygons compromises the quality or even the correctness of the simulation. Our considerations therefore raise the question, whether or not it is worth to handle curved objects directly. It means that one has to spend more effort on the basic routines computing the distance between two patches, but can profit from higher accuracy and a smaller number of faces.

In this paper we present an efficient algorithm for computing the distance between so-called 'quadratic complexes'. These are generalized polyhedra where the boundary consists of quadratic surface patches trimmed by quadratic curves. This class of surfaces plays an important part in designing mechanical parts and represents a set of modeling primitives provided by common CAD-systems. We reduce the distance calculation problem to the prob-

Central Surfaces: $\det(\boldsymbol{A}) \neq 0$			
Ellipsoids / Hyperboloids	$\boldsymbol{a} = \boldsymbol{0}$	$a_0 \neq 0$	
Cone	$\boldsymbol{a} = \boldsymbol{0}$	$a_0 = 0$	
Non-Central Surfaces: $\det(\boldsymbol{A}) = 0$			
Paraboloids	$A_3 = 0$	$a_3 \neq 0$	$a_0 = 0$
Ellipt. /Hyperbol. Cylinder	$A_3 = 0$	$\boldsymbol{a} = \boldsymbol{0}$	$a_0 \neq 0$
Parabol. Cylinder	$A_1 = A_3 = 0$	$a_1 \neq 0$	$a_0 = 0$

Table 1. Normal Forms of Quadratic Surfaces.

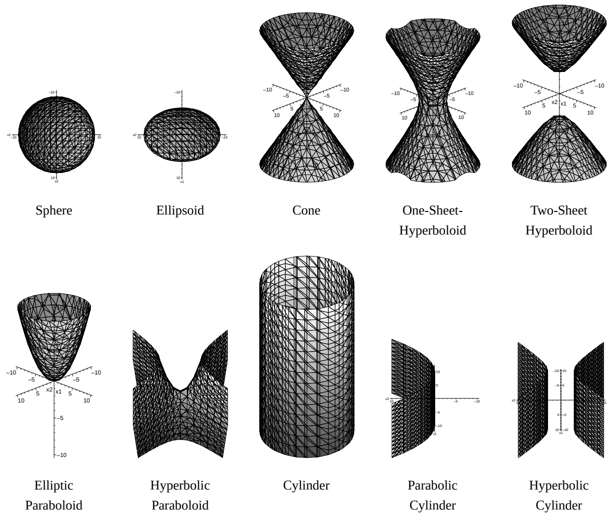

Figure 1. Typical examples of Quadratic Surfaces.

lem of solving univariate polynomial equations. For general quadrics we show that the degrees of the polynomials are bounded by 24. We have, however, identified an important subclass, the so-called natural quadrics, for which distance computation queries only require solving univariate polynomials of a degree of at most 8. We will also see that this upper bound is strict.

2. Quadratic Complexes and Extensions

Rigid bodies are usually modeled as polyhedra and are represented by topological and geometrical descriptions of their boundaries. Thereby faces are embedded on planes and edges are given by line segments. Quadratic complexes can be considered as the most natural generalization of this polyhedral model with respect to a curved boundary representation. The faces of a quadratic complex are geometrically described by quadratic surfaces (quadrics) with edges represented by quadratic curves (conics).

A quadratic surface is implicitly given by an algebraic equation of degree 2:

$$\{\boldsymbol{x} \in \mathbb{R}^3 \,|\, \boldsymbol{x}^T \boldsymbol{A} \boldsymbol{x} + 2\boldsymbol{a}^T \boldsymbol{x} + a_0 = 0\}, \qquad (1)$$

for a vector $\boldsymbol{a} \in \mathbb{R}^3$ and a symmetric matrix $\boldsymbol{A} \in \mathbb{R}^{3\times 3}$.
Table 1 gives an overview over the so-called 'normal forms' (except from some degenerate cases), providing a classification that we will use in this paper. Typical examples of quadratic surfaces are shown in figure 1.

A quadratic curve is explicitly given as the following point set:

$$\{\boldsymbol{p} \in \mathbb{R}^3 \,|\, \boldsymbol{p} \;=\; \boldsymbol{c} + r(t)\boldsymbol{u} + s(t)\boldsymbol{v}\}, \qquad (2)$$

where $(r, s) \;\in\; \{(\sin, \cos), (\sinh, \cosh), (\mathrm{id}, 0), (\mathrm{id}, \mathrm{id}^2)\}$ and $\boldsymbol{u}, \boldsymbol{v} \in \mathbb{R}^3$ with $\boldsymbol{u}^T\boldsymbol{v} = 0$.
Hence, conics are ellipses, hyperbolas, lines or parabolas.

We will also consider a subset of quadrics, the so-called 'natural quadrics'. This class consists of the plane, the sphere, the cylinder and the circular cone. Analogously the line and the circle form the set of 'natural conics'. Quadratic complexes with faces that are embedded on natural quadrics and trimmed by natural conics are consequently called 'natural quadratic complexes'.

Due to its importance in CAD it seems reasonable to extend the class of quadrics by the torus, which can be implicitly described by a polynomial of degree 4.

However, the set of quadratic complexes is not closed under Boolean operations, since the intersection curves cannot always be represented by quadratic curves.

3. The Distance Computation Problem

To get a precise mathematical formulation of the problem, we consider a quadratic complex as a non-empty, compact and connected set of points. The Euclidean distance δ between two points $\boldsymbol{p}$ and $\boldsymbol{q}$ can be extended to arbitrary point sets P and Q in a natural way:

$$\delta(P, Q) := \inf\{\delta(\boldsymbol{p}, \boldsymbol{q}) | \boldsymbol{p} \in P, \boldsymbol{q} \in Q\}. \qquad (3)$$

From a practical point of view it makes sense to consider the quadratic distance function δ^2, since in many cases square roots can be eliminated by this monotonous transformation. Apart from the (quadratic) distance value we request a pair of closest points as witness of the distance minimum between both quadratic complexes.

Definition 1. *Given two quadratic complexes $\boldsymbol{C}_1$, $\boldsymbol{C}_2$ at a given point and orientation, the distance computation prob-*

lem is to determine the global minimum of the distance function δ between the respective point sets, i.e.

(i) the value $\delta^* := \delta(\boldsymbol{C}_1, \boldsymbol{C}_2)$,

(ii) a pair of points $(\boldsymbol{p}, \boldsymbol{q})$, s.t. $\delta^* = \delta(\boldsymbol{p}, \boldsymbol{q})$.

To compute the distance minimum, we can restrict ourselves to the boundaries that are described by the sets of faces. The next section consequently deals with closest points between quadratic surfaces.

4. Closest Points Between Faces

The basic problem is to determine the distance between two faces, that are – from a geometrical point of view – trimmed patches of quadratic surfaces Q_1 and Q_2:

$$\begin{aligned} Q_1 &:= \{\boldsymbol{x} \,|\, \boldsymbol{x}^T \boldsymbol{A}\boldsymbol{x} + 2\boldsymbol{a}^T\boldsymbol{x} + a_0 = 0\}, \\ Q_2 &:= \{\boldsymbol{y} \,|\, \boldsymbol{y}^T \boldsymbol{B}\boldsymbol{y} + 2\boldsymbol{b}^T\boldsymbol{y} + b_0 = 0\}. \end{aligned}$$

Let us first assume that the considered face pair is disjoint. If it is not, the distance value δ is zero and we are done. As long as we consider quadratic surfaces this condition can be easily checked by solving polynomials of a degree of at most 4 [7, 10]. Extending this class of surfaces by the torus, the former result still holds with the only exception that the intersection problem between a torus and a non-natural quadratic surface requires the solution of a higher degree polynomial [6, 10].

The disjointness assumption allows us to characterize a pair of closest points between two faces:

Theorem 1. *Let f_1 and f_2 be disjoint faces of quadratic complexes that are embedded on the quadratic surfaces Q_1 and Q_2. If $(\boldsymbol{p}_1, \boldsymbol{p}_2)$ is a pair of closest points between f_1 and f_2, then either*

(i) $(\boldsymbol{p}_1, \boldsymbol{p}_2)$ is an extremum of the distance function between Q_1 and Q_2, i.e. there are $\lambda, \mu \in \mathbb{R}$, $\lambda, \mu \neq 0$, s.t.

$$\boldsymbol{n}(\boldsymbol{p}_1) = \lambda(\boldsymbol{p}_2 - \boldsymbol{p}_1) \qquad \boldsymbol{n}(\boldsymbol{p}_2) = \mu(\boldsymbol{p}_1 - \boldsymbol{p}_2),$$

where $\boldsymbol{n}(\boldsymbol{p}_i)$ denotes the normal of Q_i in $\boldsymbol{p}_i$, or

(ii) $\boldsymbol{p}_1$ or $\boldsymbol{p}_2$ lies on the boundary of the face f_1 or f_2, respectively.

Proof. If neither $\boldsymbol{p}_1 \in f_1$ nor $\boldsymbol{p}_2 \in f_2$ are located at the boundary of the respective face, then both points form a local extremum of the distance function between Q_1 and Q_2. Since both faces do not intersect, $\boldsymbol{p}_1$ and $\boldsymbol{p}_2$ are points for which the interpolating line is perpendicular to both surfaces in the respective points. This observation can be formally proven by setting up the LAGRANGE-function $\mathcal{L}$ for the problem $\min(\boldsymbol{x} - \boldsymbol{y})^2$, $\boldsymbol{x} \in Q_1$ $\boldsymbol{y} \in Q_2$:

$$\begin{aligned} \mathcal{L}(\boldsymbol{x}, \boldsymbol{y}; \alpha, \beta) &= (\boldsymbol{x} - \boldsymbol{y})^2 + \alpha(\boldsymbol{x}^T\boldsymbol{A}\boldsymbol{x} + 2\boldsymbol{a}^T\boldsymbol{x} + a_0) \\ &\quad + \beta(\boldsymbol{y}^T\boldsymbol{B}\boldsymbol{y} + 2\boldsymbol{b}^T\boldsymbol{y} + b_0) \end{aligned}$$

and the following LAGRANGE-conditions:

$$\begin{aligned} \partial\mathcal{L}(.)/\partial\boldsymbol{x} = 0 &\Leftrightarrow \alpha(\boldsymbol{A}\boldsymbol{x} + \boldsymbol{a}) = \boldsymbol{y} - \boldsymbol{x}, && (4) \\ \partial\mathcal{L}(.)/\partial\boldsymbol{y} = 0 &\Leftrightarrow \beta(\boldsymbol{B}\boldsymbol{y} + \boldsymbol{b}) = \boldsymbol{x} - \boldsymbol{y}, && (5) \\ \partial\mathcal{L}(.)/\partial\alpha = 0 &\Leftrightarrow \boldsymbol{x}^T A\boldsymbol{x} + 2\boldsymbol{a}^T\boldsymbol{x} + a_0 = 0, && (6) \\ \partial\mathcal{L}(.)/\partial\beta = 0 &\Leftrightarrow \boldsymbol{y}^T B\boldsymbol{y} + 2\boldsymbol{b}^T\boldsymbol{y} + b_0 = 0, && (7) \end{aligned}$$

where $\boldsymbol{n}(\boldsymbol{x}) \parallel \boldsymbol{A}\boldsymbol{x} + \boldsymbol{a}$ and $\boldsymbol{n}(\boldsymbol{y}) \parallel \boldsymbol{B}\boldsymbol{y} + \boldsymbol{b}$.

Figure 2 provides a 2-dimensional illustration of both cases and the necessary precondition. Closest points are marked by enclosing squares whereas the points of case (i), forming a distance extremum between Q_1 and Q_2, are connected by their interpolating line segment. □

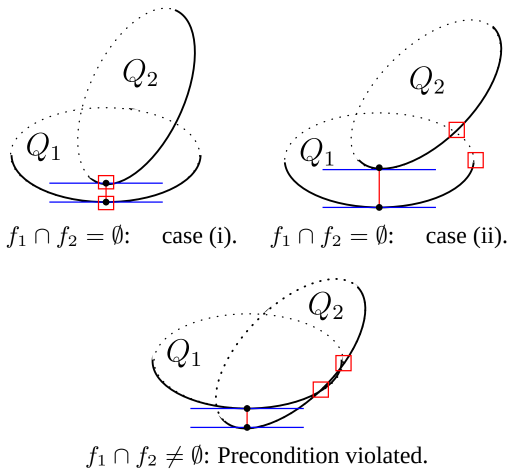

Figure 2. Illustration of Theorem 1.

4.1. A Generic Algorithm

Theorem 1 suggests a simple recursive procedure (cf. algorithm 3) for computing the distance between two faces. Arguments of the algorithm are the entities of the boundary representation, i.e. a face, an edge or a vertex. When called on two faces, the algorithm first checks the disjointness condition by the routine INTERSECT. If the faces collide the distance value is set to zero and a pair of points witnessing the intersection is returned. Otherwise, all point pairs representing extrema of the distance function between both surfaces are computed. To do so, the subroutine EXTREMA enumerates all pairs $(\boldsymbol{p}_1, \boldsymbol{p}_2)$ that fulfill the conditions of case (i) for the untrimmed entities. In the variable δ_G we

Input: Entities E_1 and E_2 of type face, edge or vertex.
Output: $\delta(E_1, E_2)$ and a pair of closest points $(\boldsymbol{p}_1, \boldsymbol{p}_2)$.
ENTITYDISTANCE(E_1, E_2)
$\left[isDisjoint, (\boldsymbol{p}_1, \boldsymbol{p}_2)\right] \leftarrow$ INTERSECT(E_1, E_2)
if $isDisjoint = false$
    return $\left[0, (\boldsymbol{p}_1, \boldsymbol{p}_2)\right]$
$\delta_G \leftarrow \infty$
while $\left[\delta, (\boldsymbol{q}_1, \boldsymbol{q}_2)\right] \leftarrow$ EXTREMA(E_1, E_2)
    if $(\boldsymbol{q}_1 \in E_1)$**and** $(\boldsymbol{q}_2 \in E_2)$
        if $\delta < \delta_G$
            $\delta_G \leftarrow \delta, \quad (\boldsymbol{p}_1, \boldsymbol{p}_2) \leftarrow (\boldsymbol{q}_1, \boldsymbol{q}_2)$
foreach subentity E **of** E_1
    $\left[\delta, (\boldsymbol{q}_1, \boldsymbol{q}_2)\right] \leftarrow$ ENTITYDISTANCE(E, E_2)
    if $\delta < \delta_G$
        $\delta_G \leftarrow \delta, \quad (\boldsymbol{p}_1, \boldsymbol{p}_2) \leftarrow (\boldsymbol{q}_1, \boldsymbol{q}_2)$
foreach subentity E **of** E_2
    $\left[\delta, (\boldsymbol{q}_1, \boldsymbol{q}_2)\right] \leftarrow$ ENTITYDISTANCE(E_1, E)
    if $\delta < \delta_G$
        $\delta_G \leftarrow \delta, \quad (\boldsymbol{p}_1, \boldsymbol{p}_2) \leftarrow (\boldsymbol{q}_1, \boldsymbol{q}_2)$
return $\left[\delta_G, (\boldsymbol{p}_1, \boldsymbol{p}_2)\right]$

Figure 3. A generic algorithm for computing the distance between two entities of the boundary description.

compute the minimal distance of all these 'extremal' points that lie inside the trimmed patches. However, the theorem also states that the closest points can also be found at the boundary of at least one of the patches (case(ii)). Therefore two recursive calls to the distance procedure must be executed, with the entity E_1 in the first call and the entity E_2 in the second call restricted to their boundaries ∂E_1 and ∂E_2, respectively. In our algorithm the elements of ∂E_i are called 'subentities' of E_i, $i = 1, 2$. If the original arguments of the procedure were two faces, then the recursive calls would reciprocally determine the distance between the edges of one face and the other patch. The recursion ends if case (i) holds for a pair of points computed by procedure EXTREMA. This happens at the latest if the procedure is called on two vertices.

Algorithm 3 represents an elegant high-level formulation of the whole procedure. A practical implementation, however, must prevent repeated distance computation between identical pairs of entities. Moreover, one can avoid the recursive calls on the boundary elements when both the trimmed as well as the untrimmed entities do not intersect and the extremal points with minimal distance between the untrimmed entities lie inside the trimmed patches.

4.2. Degree Complexity of the Polynomial Systems

Procedure EXTREMA solves the central task of algorithm 3. It identifies the local extrema of the distance function between untrimmed entities of the boundary representation. In the next sections we will prove the following result which gives an upper bound on the degree complexity of the systems that have to be solved.

Theorem 2. *The distance between two faces of quadratic complexes can be computed by solving systems of univariate and bivariate polynomials in which the degree of every variable is 6 at most. These systems can be solved by finding the roots of univariate polynomials of a degree of at most 24.*

4.3. The Surface-Surface Case

In this section we will set up a system of bivariate polynomial equations that gives the local extrema of the distance function in the case of two quadratic surfaces.

Solving The Lagrange System

In the proof of theorem 1 we have seen that the perpendicularity conditions of case (i) are a result of setting up the LAGRANGE formalism for the distance minimization problem between quadrics. By setting $\lambda := 1/\alpha$ and $\mu := 1/\beta$ we can derive from conditions (4) and (5):

$$\begin{aligned} \boldsymbol{C}_{\lambda,\mu}\boldsymbol{x} &= -(\boldsymbol{B}\boldsymbol{a} + \lambda\boldsymbol{b} + \mu\boldsymbol{a}), \quad (8)\\ \boldsymbol{C}_{\lambda,\mu}^T\boldsymbol{y} &= -(\boldsymbol{A}\boldsymbol{b} + \lambda\boldsymbol{b} + \mu\boldsymbol{a}), \quad (9) \end{aligned}$$

with $\boldsymbol{C}_{\lambda,\mu} := \boldsymbol{B}\boldsymbol{A} + \lambda\boldsymbol{B} + \mu\boldsymbol{A}$.
If we solve these equations for $\boldsymbol{x}$ and $\boldsymbol{y}$, we get:

$$\boldsymbol{x} = -\boldsymbol{C}_{\lambda,\mu}^{-1}\boldsymbol{c}_B \qquad \boldsymbol{y} = -{\boldsymbol{C}_{\lambda,\mu}^{-1}}^T\boldsymbol{c}_A,$$

with $\boldsymbol{c}_B := \boldsymbol{B}\boldsymbol{a} + \lambda\boldsymbol{b} + \mu\boldsymbol{a}$ and $\boldsymbol{c}_A := \boldsymbol{A}\boldsymbol{b} + \lambda\boldsymbol{b} + \mu\boldsymbol{a}$. Substituting the expressions for $\boldsymbol{x}$ and $\boldsymbol{y}$ in (6) and (7) gives the following system of polynomial equations:

$$\begin{aligned} f(\lambda,\mu) &= \boldsymbol{c}_B^T\overline{\boldsymbol{C}}_{\lambda,\mu}^T\boldsymbol{A}\overline{\boldsymbol{C}}_{\lambda,\mu}\boldsymbol{c}_B - \quad (10)\\ &\quad 2|\boldsymbol{C}_{\lambda,\mu}|\boldsymbol{a}^T\overline{\boldsymbol{C}}_{\lambda,\mu}\boldsymbol{c}_B + a_0|\boldsymbol{C}_{\lambda,\mu}|^2 = 0,\\ g(\lambda,\mu) &= \boldsymbol{c}_A^T\overline{\boldsymbol{C}}_{\lambda,\mu}\boldsymbol{B}\overline{\boldsymbol{C}}_{\lambda,\mu}^T\boldsymbol{c}_A - \quad (11)\\ &\quad 2|\boldsymbol{C}_{\lambda,\mu}|\boldsymbol{b}^T\overline{\boldsymbol{C}}_{\lambda,\mu}^T\boldsymbol{c}_A + b_0|\boldsymbol{C}_{\lambda,\mu}|^2 = 0, \end{aligned}$$

where $\overline{\boldsymbol{C}}_{\lambda,\mu}$ denotes the adjoint and $|\boldsymbol{C}_{\lambda,\mu}|$ the determinant of $\boldsymbol{C}_{\lambda,\mu}$.

The Inverse of $C_{\lambda,\mu}$

In order to solve the given system, we must express the inverse of $C_{\lambda,\mu}$ as a bivariate matrix polynomial in λ and μ. The following proposition shows how the adjoint as well as the determinant can be written in terms of matrix coefficients in A and B.

Proposition 1. *The adjoint and determinant of* $C_{\lambda,\mu} = BA + \lambda B + \mu A$ *is given by*

$$\begin{aligned}
\overline{C_{\lambda,\mu}} &= \overline{B}\lambda^2 + \overline{A}\mu^2 + T_A\overline{B}\lambda + \overline{A}T_B\mu + \\
&\quad (T_BT_A - T_{AB})\lambda\mu + \overline{A}\,\overline{B}, \\
|C_{\lambda,\mu}| &= |B|\lambda^3 + |A|\mu^3 + |A||B| + \\
&\quad |B|tr(A)\lambda^2 + |A|tr(B)\mu^2 + \\
&\quad |B|tr(\overline{A})\lambda + |A|tr(\overline{B})\mu + \\
&\quad tr(\overline{B}A)\lambda^2\mu + tr(\overline{A}B)\lambda\mu^2 + \\
&\quad (tr(\overline{A})tr(\overline{B}) - tr(\overline{A}\,\overline{B}))\lambda\mu,
\end{aligned}$$

where $T_M := tr(M)E - M$ *for a matrix* $M \in \mathbb{R}^{3\times3}$.

Proof. We start with a simple problem, asking for the inverse of $G_\lambda := D + \lambda C$. Since the characteristic polynomial of a matrix $M \in \mathbb{R}^{3\times3}$ is given by

$$|M - \lambda E| \;=\; -\lambda^3 + \mathrm{tr}(M)\lambda^2 - \mathrm{tr}(\overline{M})\lambda + |M|.$$

Setting $M := -DC^{-1}$ we get

$$\begin{aligned}
|G_\lambda| &= -|C||M - \lambda E| \\
&= |C|\lambda^3 + \mathrm{tr}(D\overline{C})\lambda^2 + \mathrm{tr}(C\overline{D})\lambda + |D|. \qquad (12)
\end{aligned}$$

Now one can easily verify that the adjoint can be written as the following matrix polynomial:

$$\overline{G}_\lambda = \overline{C}\lambda^2 + (T_DT_C - T_{CD})\lambda + \overline{D}. \qquad (13)$$

The determinant and the adjoint of $C_{\lambda,\mu}$ are the result of setting $D := BA + \lambda B$, $C := A$ and recursively applying rule (12) and (13). □

The theorem shows that the adjoint of $C_{\lambda,\mu}$ is a bivariate matrix polynomial of degree 2 in λ and μ. Since the degree of the determinant polynomial is 3, we can conclude that both polynomials have degree 6 in λ and μ.

Central Surfaces

For central surfaces the vector a vanishes in normal form. However, the degree of f and g does not decrease in this special case since it is still determined by $|C_{\lambda,\mu}|^2$. To analyze the complexity of solving this bivariate system, one first observes that both polynomials are sparse, having a total degree of 6. Hence, it follows from the BEZOUT bound that over $\mathbb{C}^2$ the number of complex roots is at most 36. These roots can be found with the help of elimination methods. Eliminating one variable and solving the univariate resultant polynomial $\mathrm{Res}(f,g)$ give the bivariate solutions projected onto the non-eliminated variable. Since the leading coefficients in λ as well as in μ are constants, the degree of $\mathrm{Res}(f,g)$ is exactly the number of bivariate roots and therefore bounded by 36.

However, it is not necessary to solve this polynomial directly. The following lemma will show that, in our setting, the resultant is always the product of two polynomials of a lower degree. Solving the latter polynomials improves numerical stability as well as computational efficiency.

Lemma 1. *Let f, g be the polynomials given in (10), (11) and the system h defined as follows:*

$$h(\lambda,\mu) := (h_1,h_2,h_3)^T \;=\; \overline{C}_{\lambda,\mu}c_B \;=\; \mathbf{0}.$$

Then the roots of $h = 0$ are also solutions to $f = g = 0$.

Proof. W.l.o.g. we can assume that the quadric given in normal form does not contain the origin, i.e. $x \neq \mathbf{0}$. Now let us consider values of λ and μ for which h vanishes. By multiplying equation (8) with $\overline{C}_{\lambda,\mu}$ from the lefthand side, we get $|C_{\lambda,\mu}|x = -\overline{C}_{\lambda,\mu}c_B = 0$ and therefore $|C_{\lambda,\mu}| = 0$. Applying the same transformation to (9) gives $|C_{\lambda,\mu}|y = -\overline{C}^T_{\lambda,\mu}c_A = 0$. Since the determinant vanishes, the same must hold for $\overline{C}^T_{\lambda,\mu}c_A$. Now it is easy to see that every term in f and g becomes zero and we can conclude that the solutions of h solve the equations f and g. □

The system $h = \mathbf{0}$ consists of three polynomial equations $h_i = 0$, $i = 1,2,3$, that have degree 2 in λ as well as μ. For every h_i, h_j with $1 \leq i < j \leq 3$, it holds that the resultant $r_{ij} = \mathrm{Res}(h_i,h_j)$ is a polynomial of degree 3 in the non-eliminated variable. The three polynomials r_{ij} are identical up to a constant factor and by Lemma 1 we know that every root of r_{ij} solves the resultant polynomial of the system $f = g = 0$. Moreover, one can show that the multiplicity of these roots is 4. Hence, we have found a polynomial of degree 12, i.e. r^4_{ij}, that divides $\mathrm{Res}(f,g)$. These considerations prove that the closest points between two central surfaces can be determined by solving polynomials of a degree of at most 24.

If Q_1 is a cone and Q_2 a general central surface then f simplifies to a polynomial of degree 4 in λ as well as μ. Hence, the resultant polynomial is of a degree of at most 24 and can be written as the product of two polynomials of degree 12. In the case of two cones we analogously observe that the roots of the resultant polynomial can be determined by solving polynomials of a maximum degree 4.

Non-Central Surfaces

In the case of non-central surfaces we have $|A| = |B| = 0$ and therefore $|C_{\lambda,\mu}|$ simplifies to a quadratic polynomial in both, λ and μ. This observation should give an intuition why the polynomial systems for non-central surfaces are less complex than in the case of central surfaces.

If Q_1 and Q_2 are both paraboloids, then f and g become bivariate polynomials of degree 5 in λ and μ. Since the total degree is also 5 in both polynomials, the resultant is a degree 25 polynomial in the worst case. However, our considerations above have shown that $\text{Res}(f, g)$ can be factorized into two lower degree polynomials. Since the equations $h_i = 0$, $i = 1, 2, 3$, do not change in degree compared to the general case, one of the two factors is a polynomial of a degree of at most 12 and the other of a degree of at most 13.

For elliptic and hyperbolic cylinders f and g collapse to degree 2 polynomials in λ and μ. The resultant is therefore a quartic polynomial that cannot be factorized as in previous cases. The reason is that $\boldsymbol{h} = \mathbf{0}$ has only trivial solutions $\lambda = 0$ or $\mu = 0$ that can be eliminated from f and g before computing the resultant.

At this point we will omit the cases where Q_1 is a central surface and Q_2 not. As one would expect, the degrees of the polynomials are lower than in the case of central surfaces, but higher compared to the situation where only non-central surfaces are involved. The results of this section are recorded in the following table:

	Central Surfaces		**Non-Central Surfaces**		
	$a_0 \neq 0$	$a_0 = 0$	$\boldsymbol{a} \neq \mathbf{0}$	$\boldsymbol{a} = \mathbf{0}$	rg$\boldsymbol{A} = 1$
$a_0 \neq 0$	24	12	18	12	8
$a_0 = 0$		4	8	4	2
$\boldsymbol{a} \neq \mathbf{0}$			13	8	5
$\boldsymbol{a} = \mathbf{0}$				4	2

4.4. The Point-Surface Case

As in the section before we will start by setting up the LAGRANGE-formalism for the respective distance minimization problem. W.l.o.g. we assume that the surface is given in normal form. Let $\boldsymbol{p}$ denote the (transformed) query point. Then the LAGRANGE-function $\mathcal{L}$ is given by

$$\mathcal{L}(\boldsymbol{x}; \alpha) = (\boldsymbol{x} - \boldsymbol{p})^2 + \alpha(\boldsymbol{x}^T \boldsymbol{A}\boldsymbol{x} + 2\boldsymbol{a}^T \boldsymbol{x} + a_0).$$

The partial derivatives with respect to $\boldsymbol{x}$ and α yield the LAGRANGE-conditions:

$$\frac{\partial \mathcal{L}(.)}{\partial \boldsymbol{x}} = 0 \iff \alpha(\boldsymbol{A}\boldsymbol{x} + \boldsymbol{a}) = \boldsymbol{p} - \boldsymbol{x}, \tag{14}$$

$$\frac{\partial \mathcal{L}(.)}{\partial \alpha} = 0 \iff \boldsymbol{x}^T A \boldsymbol{x} + 2\boldsymbol{a}^T \boldsymbol{x} + a_0 = 0. \tag{15}$$

Again the first equation reflects the geometric intuition that the interpolating line through $\boldsymbol{p}$ and $\boldsymbol{x}$ is parallel to the surface normal in $\boldsymbol{x}$. It can be transformed into

$$\boldsymbol{x} = (\boldsymbol{E} + \alpha \boldsymbol{A})^{-1}(\boldsymbol{p} - \alpha \boldsymbol{a}). \tag{16}$$

Substituting $\boldsymbol{x}$ in (15) we get the univariate system:

$$\begin{aligned} f(\alpha) \;=\; & (\boldsymbol{p} - \alpha\boldsymbol{a})^T \overline{\boldsymbol{D}}_\alpha \boldsymbol{A} \overline{\boldsymbol{D}}_\alpha (\boldsymbol{p} - \alpha \boldsymbol{a}) + \\ & 2\boldsymbol{a}^T \overline{\boldsymbol{D}}_\alpha (\boldsymbol{p} - \alpha\boldsymbol{a})|\boldsymbol{D}_\alpha| + a_0 |\boldsymbol{D}_\alpha|^2 \;=\; 0, \end{aligned} \tag{17}$$

with $\boldsymbol{D}_\alpha := \boldsymbol{E} + \alpha \boldsymbol{A}$.

From equations (12) and (13), it follows that the adjoint of $\boldsymbol{D}_\alpha$ is a quadratic polynomial in α, whereas its determinant is of degree 3:

$$\begin{aligned} \overline{\boldsymbol{D}}_\alpha &= \overline{\boldsymbol{A}}\alpha^2 + \boldsymbol{T}_A \alpha + \boldsymbol{E} \\ &= \text{diag}[d_2 d_3, d_1 d_3, d_1 d_2], \\ |\boldsymbol{D}_\alpha| &= |\boldsymbol{A}|\alpha^3 + \text{tr}(\overline{\boldsymbol{A}})\alpha^2 + \text{tr}(\boldsymbol{A})\alpha + 1 \\ &= d_1 d_2 d_3, \end{aligned}$$

where $d_i := 1 + \alpha A_i$ and A_i denotes the ith diagonal element of $\boldsymbol{A}$.

Expanding both quantities in (17) gives a degree 6 polynomial of the form

$$\begin{aligned} f(\alpha) = {} & A_1 p_{\alpha 1}^2 d_2^2 d_3^2 + A_2 p_{\alpha 2}^2 d_1^2 d_3^2 + A_3 p_{\alpha 3}^2 d_1^2 d_2^2 + \\ & 2(a_1 p_{\alpha 1} d_1 d_2^2 d_3^2 + a_2 p_{\alpha 2} d_1^2 d_2 d_3^2 + a_3 p_{\alpha 3} d_1^2 d_2^2 d_3) + \\ & a_0 d_1^2 d_2^2 d_3^2 \end{aligned} \tag{18}$$

with $\boldsymbol{p}_\alpha := \boldsymbol{p} - \alpha \boldsymbol{a}$.

Central Surfaces

If we consider central faces, the vector $\boldsymbol{a}$ is equal to $\mathbf{0}$ and therefore f simplifies to

$$f(\alpha) = A_1 p_1^2 d_2^2 d_3^2 + A_2 p_2^2 d_1^2 d_3^2 + A_3 p_3^2 d_1^2 d_2^2 + a_0 d_1^2 d_2^2 d_3^2.$$

The degree of this polynomial is still determined by the term $d_1^2 d_2^2 d_3^2$ and therefore does not decrease compared with the general case. However, if we additionally assume that a_0 vanishes, i.e. if we are dealing with a cone, then f becomes a polynomial of degree 4 in α:

$$f(\alpha) = A_1 p_1^2 d_2^2 d_3^2 + A_2 p_2^2 d_1^2 d_3^2 + A_3 p_3^2 d_1^2 d_2^2. \tag{19}$$

Non-Central Surfaces

In the case of non-central surfaces we first analyze the situation where $\boldsymbol{a} = \mathbf{0}$, i.e. where the given face is embedded on an elliptic or hyperbolic cylinder. Since A_3 is equal to zero, d_3 becomes 1 and in comparison to (18) the degree of the polynomial f decreases to 4:

$$f(\alpha) = A_1 p_1^2 d_2^2 + A_2 p_2^2 d_1^2 + a_0 d_1^2 d_2^2.$$

If the cylinder is parabolic, i.e. $A_1 = A_3 = a_0 = 0, a_1 \neq 0$, then d_1 as well as d_3 are constants and we obtain the following polynomial of degree 3 in α:

$$f(\alpha) = a_1 p_{\alpha 1} d_2^2. \tag{20}$$

Finally we have to consider the case of paraboloids for which $a_3 \neq 0$ and A_3 as well as a_0 vanish. Again d_3 is equal to 1, reducing the degree of f to 5.

$$f(\alpha) \;=\; A_1 p_1^2 d_2^2 + A_2 p_2^2 d_1^2 + 2a_3 p_{\alpha 3} d_1^2 d_2^2. \tag{21}$$

The following table summarizes our results in the point-surface case:

Point - Central Surface		
Ellipsoid	Hyperboloid	Cone
6	6	4
Point - Non-Central Surface		
Paraboloids	Elliptic / Hyperbolic Cylinders	Parabolic Cylinder
5	4	3

4.5. The Curve-Surface Case

In the former approach, the point $\boldsymbol{p}$ was considered a constant, not a variable. To derive the system of polynomial equations in the curve-surface case, we simply substitute $\boldsymbol{p}$ by the explicit representation of quadratic curves.

$$P: \quad \boldsymbol{p}(t) \;=\; c + r(t)\boldsymbol{u} + s(t)\boldsymbol{v}.$$

The new variable t introduces a third LAGRANGE-condition

$$\frac{\partial \mathcal{L}(.)}{\partial t} \;=\; 0 \quad \Longleftrightarrow \quad (\boldsymbol{x}-\boldsymbol{p})^T \frac{\partial \boldsymbol{p}}{\partial t}.$$

By substituting $\boldsymbol{x}$ according to (16), we get the condition:

$$g(\alpha, t) \;:=\; \left(\overline{\boldsymbol{D}}_\alpha(\boldsymbol{p}-\alpha\boldsymbol{a}) - |\boldsymbol{D}_\alpha|\boldsymbol{p}\right)\frac{\partial \boldsymbol{p}}{\partial t} \;=\; 0.$$

Expanding the polynomials $\overline{\boldsymbol{D}}_\alpha$ and $|\boldsymbol{D}_\alpha|$ yields the second equation of the system:

$$g(\alpha, t) \;=\; \tilde{p}_1 p_1' d_2 d_3 + \tilde{p}_2 p_2' d_1 d_3 + \tilde{p}_3 p_3' d_1 d_2 \;=\; 0,$$

with $\tilde{\boldsymbol{p}} := \boldsymbol{A}\boldsymbol{p} + \boldsymbol{a}$ and $\boldsymbol{p}' := \frac{\partial \boldsymbol{p}}{\partial t}$.

Central Surfaces

Let us first consider central surfaces for which $\boldsymbol{a} = \boldsymbol{0}$. By interpreting (18) as a polynomial in α and t we get the following system of bivariate polynomial equations:

$$f(\alpha,t) = A_1 p_1^2 d_2^2 d_3^2 + A_2 p_2^2 d_1^2 d_3^2 + A_3 p_3^2 d_1^2 d_2^2 + a_0 d_1^2 d_2^2 d_3^2 = 0,$$
$$g(\alpha,t) = A_1 p_1 p_1' d_2 d_3 + A_2 p_2 p_2' d_1 d_3 + A_3 p_3 p_3' d_1 d_2 = 0.$$

Whereas the degree in α is 6 for f and 2 for g, the degree of t obviously depends on the curve involved. First we use the following substitutions to rationally parameterize ellipses and hyperbolas:

$$r(t) = \frac{1-t^2}{1+t^2}, \quad s(t) = \frac{2t}{1+t^2}, \qquad \text{(Ellipse)} \tag{22}$$
$$r(t) = \frac{1+t^2}{1-t^2}, \quad s(t) = \frac{2t}{1-t^2}. \qquad \text{(Hyperbola)} \tag{23}$$

To eliminate the denominators in p_i, $i = 1, 2, 3$, we multiply f and g by $(1 \pm t^2)$ and obtain a system of polynomial equations with degree 4 in t. Since the total degree of f and g is 10 and 6 respectively, the BEZOUT bound for the degree of the resultant is 60. In contrast to the surface-surface case, this value drastically overestimates the actual degree. Therefore we computed the so-called 'mixed-volume' ($MV(f,g)$) [1], which – in the case of sparse polynomials – leads to a better bound on the degree of the resultant. For the given system $f = g = 0$ the mixed-volume bound is tight indicating a degree of 32.

However, it is easy to observe that if the polynomial equation $d_i = 0$ has a solution α_i, then the vector (α_i, t_i) is a solution of the bivariate system for every t_i solving the equation $p_i = 0$, $i = 1, 2, 3$. Moreover, one can show that every α_i is a root of multiplicity 4 in Res(f, g, t), whereas every t_i has multiplicity 2 in Res(f, g, α). Hence Res(f, g) can be written as the following product:

$$\text{Res}(f,g,t) \;=\; h_\alpha \prod_{i=1}^{3} d_i^4 \;=\; h_\alpha \prod_{i=1}^{3} (\alpha - \alpha_i)^4,$$
$$\text{Res}(f,g,\alpha) \;=\; h_t \prod_{i=1}^{3} p_i^2 \;=\; h_t \prod_{i=1}^{3} (t-t_{i1})^2 (t-t_{i2})^2,$$

where h_α and h_t are univariate polynomials of a degree of at most 20.

If P is a parabola, then g becomes a cubic polynomial in t. Since $r(t) = t$ and $s(t) = t^2$, p_i is quadratic and p_i' is linear in t, leading to a degree 4 polynomial f and a degree 3 polynomial g. The mixed-volume of f and g decreases to 26 and therefore the degrees of the polynomials h_α, h_t to 14. For the curve being a line, i.e. $r(t) = t$, $s(t) = 0$, the polynomial f is quadratic in t and g turns out to be a linear function. The resultant polynomial also simplifies such that the mixed-volume becomes 10. In this simple setting, however, the roots α_i, $i = 1, 2, 3$ have multiplicity 2 and therefore h_α and h_t are polynomials of a degree of at most 4.

In the special case of a cone surface, f also becomes a degree 4 polynomial in α, whereas the degree in t does not change compared to the setting discussed above:

$$f(\alpha,t) = A_1 p_1^2 d_2^2 d_3^2 + A_2 p_2^2 d_1^2 d_3^2 + A_3 p_3^2 d_1^2 d_2^2 = 0 \tag{24}$$
$$g(\alpha,t) = A_1 p_1 p_1' d_2 d_3 + A_2 p_2 p_2' d_1 d_3 + A_3 p_3 p_3' d_1 d_2 = 0.$$

Similar to the previous case, one can show that computing the roots of the resultant requires solving univariate polynomials of a degree of at most 12 (P: ellipse / hyperbola), 8 (P:

parabola) and 2 (P: line) respectively. The following table gives an overview of the results described in this section:

Curve - Central Surface			
	Ellipsoid	Hyperboloids	Cone
Ellipse	20	20	12
Hyperbola	20	20	12
Parabola	14	14	8
Line	4	4	2

Non-Central Surfaces

For non-central sufaces with $\boldsymbol{a} = \boldsymbol{0}$ and $a_0 \neq 0$, f remains a bivariate polynomial of degree 6 in α, whereas g simplifies to a linear function in α:

$$f(\alpha, t) \;=\; A_1 p_1^2 d_2^2 + A_2 p_2^2 d_1^2 + a_0 d_1^2 d_2^2 \;=\; 0,$$
$$g(\alpha, t) \;=\; A_1 p_1 p_1' d_2 + A_2 p_2 p_2' d_1 \;=\; 0.$$

Since $d_3 = 1$, we obtain the following factorization of the resultant polynomial:

$$\text{Res}(f, g, t) \;=\; h_\alpha \prod_{i=1}^{2} d_i^4 \;=\; h_\alpha \prod_{i=1}^{2} (\alpha - \alpha_i)^4,$$
$$\text{Res}(f, g, \alpha) \;=\; h_t \prod_{i=1}^{2} p_i^2 \;=\; h_t \prod_{i=1}^{2} (t - t_{i1})^2 (t - t_{i2})^2.$$

Moreover, we can see from the mixed-volume function that the degree of the resultant polynomial is at most 20 (P: ellipse / hyperbola), 16 (P: parabola) and 8 (P: line) respectively. Hence, the maximum degree of the univariate polynomials to solve is 12, 8 or 2, depending on the type of the curve.

In the case of a parabolic cylinder $\boldsymbol{a}$ no longer vanishes, but A_1, A_3 as well as a_0 are all zero. Therefore f becomes a cubic and g a linear polynomial in α:

$$f(\alpha, t) \;=\; a_1 p_{\alpha 1} d_2^2 \;=\; 0,$$
$$g(\alpha, t) \;=\; a_1 p_1' d_2 + A_2 p_2 p_2' \;=\; 0.$$

To compute the roots of the resultant polynomial one has to solve polynomials of a degree of at most 8 (P: ellipse / hyperbola), 5 (P: parabola) and 1 (P: line) respectively.

If the surface is an elliptic or hyperbolic paraboloid, we have already seen (cf. (21)) that f is polynomial of degree 5 in α. It is easy to see that g also simplifies to a quadratic polynomial in α:

$$f(\alpha, t) \;=\; A_1 p_1^2 d_2^2 d_+ A_2 p_2^2 d_1^2 + 2 a_3 p_{\alpha 3} d_1^2 d_2^2 \;=\; 0,$$
$$g(\alpha, t) \;=\; A_1 p_1 p_1' d_2 + A_2 p_2 p_2' d_1 + a_3 p_3' d_1 d_2 \;=\; 0.$$

Again our factorization shows that the degrees of the univariate polynomials to solve are much lower than the degrees of the respective resultant polynomial.

Our considerations in this section are summarized in the following table:

Curve - Non-Central Surface			
	Paraboloids	Ellipt. / Hyperb. Cylinders	Parab. Cylinder
Ellipse	16	12	8
Hyperbola	16	12	8
Parabola	11	8	5
Line	3	2	1

4.6. The Point-Curve Case

To compute the distance between a point $\boldsymbol{p}$ and a quadratic curve Q, we assume w.l.o.g. that Q is embedded on the $x_1 x_2$-plane and centered around the origin, i.e.

$$Q: \quad \boldsymbol{q}(t) = r(t)\boldsymbol{u} + s(t)\boldsymbol{v}, \quad \boldsymbol{u}^T \boldsymbol{v} = 0.$$

By considering the projection $\overline{\boldsymbol{p}}$ of $\boldsymbol{p}$ onto the $x_1 x_2$-plane, the task is reduced to a 2-dimensional problem:

$$\min_t (\overline{\boldsymbol{p}} - r(t)\boldsymbol{u} - s(t)\boldsymbol{v})^2.$$

Setting the derivative of the distance function equal to zero gives

$$f(t) \;=\; r r' \boldsymbol{u}^2 + s s' \boldsymbol{v}^2 - r' \overline{\boldsymbol{p}}^T \boldsymbol{u} - s' \overline{\boldsymbol{p}}^T \boldsymbol{v} = 0, \qquad (25)$$

with $r' \equiv \frac{dr}{dt}$ and $s' \equiv \frac{ds}{dt}$. In the case of an ellipse, condition (25) is equivalent to the following polynomial equation of degree 4 in t:

$$\begin{aligned} f(t) \;=\; & \overline{\boldsymbol{p}}^T \boldsymbol{v} t^4 + 2\big[\overline{\boldsymbol{p}}^T \boldsymbol{u} - (\boldsymbol{v}^2 - \boldsymbol{u}^2)\big] t^3 \\ & - \overline{\boldsymbol{p}}^T \boldsymbol{v} \quad + 2\big[\overline{\boldsymbol{p}}^T \boldsymbol{u} + (\boldsymbol{v}^2 - \boldsymbol{u}^2)\big] t^2 \;=\; 0. \end{aligned}$$

A similar result can be derived for the point-hyperbola problem.

If Q is a parabola, then the degree decreases and f becomes a cubic polynomial in t:

$$f(t) \;=\; 2\boldsymbol{v}^2 t^3 + (\boldsymbol{u}^2 - \overline{\boldsymbol{p}}^T \boldsymbol{v}) t - \overline{\boldsymbol{p}}^T \boldsymbol{u} = 0. \qquad (26)$$

In the case of a line, condition (25) simplifies to the following linear equation:

$$f(t) \;=\; \boldsymbol{u}^2 t - \overline{\boldsymbol{p}}^T \boldsymbol{v} = 0. \qquad (27)$$

The following table shows the maximum degrees of the univariate polynomials to solve in the point-curve case:

Point - Curve			
Ellipse	Hyperbola	Parabola	Line
4	4	3	1

4.7. The Curve-Curve Case

The remaining case leads to the problem of computing the distance between two quadratic curves. To simplify matters, we still assume that one curve is located as described in 4.6, i.e.

$$P: \quad \boldsymbol{p}(t) = r_1(t_1)\boldsymbol{u}_1 + s_1(t_1)\boldsymbol{v}_1, \quad \boldsymbol{u}_1^T\boldsymbol{v}_1 = 0,$$

whereas the other curve is in arbitrary position and orientation:

$$Q: \quad \boldsymbol{q}(t) = \boldsymbol{c}_2 + r_2(t_2)\boldsymbol{u}_2 + s_2(t_2)\boldsymbol{v}_2, \quad \boldsymbol{u}_2^T\boldsymbol{v}_2 = 0.$$

The partial derivatives of the distance function $\delta^2(t_1,t_2) = (\boldsymbol{q}(t_2) - \boldsymbol{p}(t_1))^2$ yield the following system of bivariate equations:

$$[\boldsymbol{q}(t_2) - \boldsymbol{p}(t_1)]^T \left[-\frac{\partial r_1}{\partial t_1}\boldsymbol{u}_1 - \frac{\partial s_1}{\partial t_1}\boldsymbol{v_1} \right] = 0, \qquad (28)$$

$$[\boldsymbol{q}(t_2) - \boldsymbol{p}(t_1)]^T \left[\ \frac{\partial r_2}{\partial t_2}\boldsymbol{u}_2 + \frac{\partial s_2}{\partial t_2}\boldsymbol{v_2} \right] = 0. \qquad (29)$$

If P and Q are both ellipses, then conditions (28) and (29) can be written in the following manner:

$$f(t_1,t_2) = (1+t_1^2)f_1(t_1,t_2) + (1+t_2^2)f_2(t_1), \qquad (30)$$

$$g(t_1,t_2) = (1+t_1^2)g_1(t_2) + (1+t_2^2)g_2(t_1,t_2), \qquad (31)$$

where f_i and g_i, $i = 1,2$, are polynomials of degrees at most 2 in t_1 and t_2. This representation immediately shows that every $(\xi_1,\xi_2) \in \{-i,i\}^2$ solves the bivariate system. Moreover, it follows that $(1+t_1^2)^2$ is a factor of the resultant $\text{Res}(f,g,t_2)$. Since the mixed-volume of the sparse polynomials f and g is 20, we can conclude that computing the distance between two ellipses requires solving univariate polynomials of a degree of at most 16.

The same result holds for the hyperbola-hyperbola- as well as for the ellipse-hyperbola-problem. In both cases one analogously observes that the degree 4 polynomial, $(1-t_1^2)^2$ and $(1+t_1^2)(1-t_1^2)^2$ respectively, is a divisor of $\text{Res}(f,g,t_2)$.

When considering two parabolas, one obtains polynomials f and g, each having total degree 3. Therefore $\text{Res}(f,g)$ is a polynomial of degree 9 in the non-eliminated variable.

The case of two lines or a line and a parabola is also straightforward, leading to a linear and cubic equation respectively.

Setting up the bivariate system (28), (29) for P being an ellipse or hyperbola and Q a line yields to polynomials f and g that have a mixed-volume of 6. By considering f and g to be polynomials in t_2, it turns out that the leading coefficient vanishes for $t_1 = \pm i$ (P: ellipse) or $t_1 = \pm 1$ (P: hyperbola) in both polynomials. From the resultant theory we have that $\text{Res}(f,g,t_2)$ also vanishes for these values and the quadratic polynomial $(1 \pm t_1^2)$ must be a factor.

Analogously one can show that in the ellipse-parabola- as well as in the hyperbola-parabola-case the *two* highest coefficients of f and g vanish for $t_1 = \pm i$ or $t_1 = \pm 1$, respectively. Therefore the degree 16 polynomial $\text{Res}(f,g,t_2)$ can be represented as the product of $(1 \pm t_1^2)^2$ and a univariate polynomial of degree 12 in t_1. The following table gives an overview of the degree bounds in this section:

Curve-Curve	Ellipse / Hyberbola	Parabola	Line
Ellipse / Hyperbola	16	12	4
Parabola		9	3
Line			1

4.8. Natural Quadrics, Conics and the Torus

Our statements so far hold for arbitrary quadratic curves and surfaces. This class, however, contains very simple geometric objects for which the general results can be substantially improved. With the natural quadrics and conics we have identified a set of special cases, which additionally is of practical importance.

4.8.1 Natural Conics

Distance queries between points and 'linear' objects (lines, planes) are standard problems in computational geometry and computer graphics [2]. They involve simple linear systems that we will not discuss here.

Hence, the circle remains the only interesting natural conic. Computing its distance to linear objects leads to polynomial equations of a degree of at most 4. In the point-circle case we can see from section 4.6 that computing the closest points requires solving a quadratic polynomial.

The circle-plane case shows the same degree complexity. We can rotate the coordinate system so that the plane is given by $x_3 = 0$. Then the distance between a point of the circle $\boldsymbol{p}(\phi)$ and the plane P is given as

$$\delta(\boldsymbol{p}(\phi), P) = c_3 + r(\cos(\phi)u_3 + \sin(\phi)v_3).$$

Minimizing δ over ϕ and applying the t-substitution (cf. 4.5) on the derivative yields to a quadratic equation in t.

The circle-circle case is the most difficult one since it requires the solutions of a polynomial of degree 8 [2]. Moreover, NEFF has shown that this is a lower bound on the degree complexity if the arithmetical operations are restricted to field operations and square roots [4].

4.8.2 Natural Quadrics and the Torus

Concerning the natural quadrics we first observe that the cases involving spheres and cylinders are already solved since they can be reduced to the case of lower dimensional objects. The distance to a sphere is just the distance to its center minus the radius. In the case of a circular cylinder we can analogously restrict ourselves to its axis and finally subtract the radius. The same argument holds for the torus – although it is not a quadratic surface. Here, the distance can be computed by considering its main circle.

Finally, we have to treat the case of a circular cone. From (19) we know that the extrema of the distance function between a point and a general cone is given by:

$$f(\alpha) \;=\; A_1 p_1^2 d_2^2 d_3^2 + A_2 p_2^2 d_1^2 d_3^2 + A_3 p_3^2 d_1^2 d_2^2 = 0. \quad (32)$$

In the special case of a circular cone, we have $A_1 = A_2$ and $d_1 = d_2$. Condition (32) therefore simplifies to a polynomial equation of degree 2 in α:

$$f(\alpha) \;=\; A_1 p_1^2 d_3^2 + A_1 p_2^2 d_3^2 + A_3 p_3^2 d_1^2 = 0.$$

For the curve-cone problem we consider system (24). Exploiting the fact that the cone is circular, we get:

$$\begin{aligned} f(\alpha, t) &= A_1 p_1^2 d_3^2 + A_1 p_2^2 d_3^2 + A_3 p_3^2 d_1^2 = 0, \\ g(\alpha, t) &= A_1 p_1 p_1' d_3 + A_1 p_2 p_2' d_3 + A_3 p_3 p_3' d_1 = 0. \end{aligned}$$

For $\boldsymbol{p}$ being a line, the resultant of f and g is a polynomial of degree 4 for which we know a divisor of degree 2 (cf. our considerations in 4.5). In the case of a circle, the mixed-volume of f and g is 12. Since we have already realized that d_3^4 is a factor of $\mathrm{Res}(f, g, t)$, we can restrict ourselves to solving polynomials of a degree of at most 8.

The remaining case is the one in which two cones are involved. For general cones we have seen in 4.3 that polynomials of a degree of at most 4 must be solved to compute the extrema of the distance function. In the special case of two circular cones this bound cannot be improved.

Our considerations above prove the following theorem:

Theorem 3. *The distance between two faces of natural quadratic complexes can be computed by solving univariate polynomials of a degree of at most 8.*

5. Conclusions

In this paper we have presented an algorithm for computing the minimal distance between two patches of quadratic surfaces bounded by quadratic curves. To find the closest points on both patches, systems of univariate and bivariate polynomial equations have to be solved. Though there are numerical techniques for finding the solutions of *multivariate* systems *directly*, our experience has shown that these methods either do not guarantee to find all roots of the system (NEWTON method) or they are not efficient enough (interval methods) to fulfill our real-time requirements. We have therefore considered algebraic techniques that reduce the multivariate root finding problem to the univariate case. The degrees of the resulting polynomials are usually high and hence elimination methods are often not appropriate in the context of floating-point computations. However, we could show that for all cases in which the degrees of the resulting polynomials become critical, a factorization into two lower degree polynomials can be found. Thereby one of the polynomial factors is easy to solve because its roots are of high multiplicity, whereas the other is of moderate degree and can be solved using numerically stable techniques.

Since in most CAD-systems the intersection curve between two quadratic surfaces is represented by NURBS, we will extend the set of trimming curves by this more complex type. Then our object class can be considered to be (approximately) closed under Boolean operations.

In a final step, we will speed up the overall algorithm by setting up a bounding volume hierarchy that reduces the number of distance computations between pairs of faces.

References

[1] J. Canny and I. Emiris. An efficient algorithm for the sparse mixed resultant. In *Proc. Intern. Symp.Applied Algebra, Algebraic Algor. and Error-Corr. Codes*, volume 263 of *Lecture Notes in Computer Science.*, pages 89–104. Springer-Verlag, 1993.

[2] D. H. Eberly. *3D game engine design : a practical approach to real-time computer graphics*. Morgan Kaufmann, 2001.

[3] S. Ehmann and M. Lin. Accurate and fast proximity queries between polyhedra using surface decomposition. In *Computer Graphics Forum (Proc. Eurographics)*, 2001.

[4] R. Farouki, C. Neff, and M. O'Connor. Automatic parsing of degenerate quadric-surface intersection. *ACM Transactions on Graphics*, 8(3):174–203, 1989.

[5] E. Gilbert, D. Johnson, and S. Keerthi. A fast procedure for computing the distance between complex objects in 3d space. *IEEE Journ. of Robot. Autom.*, 4(2):193–203, 1988.

[6] K.-J. Kim. *Torus and Simple Surface Intersection Based on a Configuration Space Approach*. Ph.D. thesis, Dep. of Computer Science and Engineering, POSTECH, Feb. 1998.

[7] J. Levin. A parametric algorithm for drawing pictures of solid objects composed of quadric surfaces. *Commun. ACM*, 19(10):555–563, 1976.

[8] M. C. Lin and J. F. Canny. Efficient algorithms for incremental distance computation. In *Proc. IEEE Internat. Conf. Robot. Autom.*, volume 2, pages 1008–1014, 1991.

[9] B. Mirtich. *Impulse-based dynamic simulation of rigid body systems*. PhD thesis, University of California, Berkeley, 1996.

[10] J. Reichel, E. Schömer, T. Warken, and C. Lennerz. Efficient collision detection for curved solid objects. In *Proc. 7^{th} ACM Symp. on Solid Modeling and Applications, SM' 02*, 2002.

[11] C. Turnbull and S. Cameron. Computing distances between nurbs-defined convex objects. In *IEEE Int. Conf. of Robot. Autom.*, pages 3686–3690, 1998.