Approximating Algebraic Space Curves by Circular Arcs*

Szilvia Béla[1] and Bert Jüttler[2]

[1] Doctoral Program in Computational Mathematics
[2] Institute of Applied Geometry,
Johannes Kepler University, Altenberger Str. 69, 4040 Linz, Austria

Abstract. We introduce a new method to approximate algebraic space curves. The algorithm combines a subdivision technique with local approximation of piecewise regular algebraic curve segments. The local technique computes pairs of polynomials with modified Taylor expansions and generates approximating circular arcs. We analyze the connection between the generated approximating arcs and the osculating circles of the algebraic curve.

Keywords: algebraic space curve, circular arc, subdivision

1 Introduction

Representing algebraic space curves is a fundamental problem of geometric computing. These curves are defined as the intersection curves of algebraic surfaces. Computation of such a surface-surface intersection is a basic operation in geometric modeling. It is important for evaluating set operations, for computing boundary curves and closely related to self-intersection problems. Several algorithms have been introduced to compute algebraic space curves. A survey of the topic is given by Patrikalakis and Maekawa [1].

Intersecting low degree implicitly defined surfaces has attracted a lot of interest in the literature. Quadratic surfaces are the simplest curved surfaces, therefore they are frequently used in computational geometry. The intersection computation of such surfaces has been discussed thoroughly in [2–6].

Several different methods have been developed for computing the intersection of algebraic surfaces (see [8–10]). Many of them are symbolic-numeric algorithms. The most widely used numeric methods are the lattice evaluation, tracing and subdivision-based methods. The lattice evaluation techniques solve a set of low dimensional sub-problems. Then the solution of these sub-problems is interpolated to approximate the general solution. Marching or tracing methods generate point sequences along the connected components of the curve. They necessarily use some topological information to find starting, turning and singular points [11–14]. Subdivision algorithms are based on the "divide and conquer" paradigm.

* This work was supported by the Austrian Science Fund (FWF) through the Doctoral Program in Computational Mathematics, subproject 3

They decompose the problem into several sub-problems, and sort these problems according to the curve topology [15, 16]. The decomposition terminates if suitable approximating primitives can be generated for each sub-problem [17]. In order to construct these approximating primitives, several local approximation techniques can be applied, such as interpolation, bounding region generation, least-squares approximation or Newton-type methods [18].

The existing local approximation techniques are based on the use of different types of approximating primitives. These primitives are often low degree curves since they have low computational costs. Several approximation algorithms generate line segments. However, algorithms that use quadratic curves may achieve a better convergence rate. A general quadratic curve represented in algebraic form can have self-intersection points or cups. In order to avoid to use non-regular curves as approximating primitives we generate circular arcs to approximate regular segments of the algebraic curve. As an alternative one can consider circular splines or bi-arcs as regular approximation primitives [23–25]. Circular arcs have many advantages compared to spline curves, such as exact arc length, offset and simple closest point computation. Therefore our local technique generates fat arcs (i.e., circular arcs with an error tolerance) as bounding primitives. The method we describe here can be used as a preprocessing step for approximating general curves by arc splines with a given tolerance, cf. [7].

In this paper we describe a hybrid algorithm which combines a subdivision technique with a new local approximation method. First we describe a technique for generating approximating circular arcs for regular algebraic curve segments. These arcs are closely related to the differential geometry of the algebraic curve. We discuss this relation and use it to confirm the convergence of the approximation technique. Then we present an algorithm using the arc generation technique combined with error estimation. In the end of the paper we describe how to combine the local arc generation approach with the global subdivision process and demonstrate the behavior of the algorithm by several examples.

2 Generating Approximating Circular Arcs

A segment of an algebraic space curve is given as the zero sets of two polynomials f and g in an axis-aligned box $\Omega_0 \subset \mathbb{R}^3$. It allocates the point set

$$\mathcal{C}(f, g, \Omega_0) = \{\mathbf{x} : f(\mathbf{x}) = 0 \wedge g(\mathbf{x}) = 0 \wedge \mathbf{x} \in \Omega_0\}.$$

In order to use a local approximation method, we consider different segments of the curve in different sub-domains of the original box $\Omega \subset \Omega_0$. All these sub-domains are again assumed to be axis-aligned boxes. We would like to approximate segments of the curve by circular arcs. Clearly this is possible only for sub-domains, that do not contain singularities of the algebraic curve. In order to control this property on a certain domain we define regular points and regular segments of algebraic space curves as follows.

Definition 1. *A point* $\mathbf{p}$ *of an algebraic space curve* $\mathcal{K} = \mathcal{C}(f, g, \Omega) \subset \mathbb{R}^3$ *is called* **regular**, *if the vectors* $\nabla f(\mathbf{p})$ *and* $\nabla g(\mathbf{p})$ *are linearly independent (and called* **singular** *otherwise). An* **algebraic space curve is regular** *in the sub-domain* Ω, *if any point of the segment in* Ω *is regular.*

We suppose that the sub-domain $\Omega \subseteq \Omega_0$, where we compute, contains only regular segments of the algebraic curve. Then it does not contain curve segments with self-intersections [16]. However, the domain can contain several segments (even closed loops) of the curve.

We present a local approximation technique. First it reformulates the algebraic equations, which define the curve. More precisely, we try to find a certain combination of the given polynomials f and g, that possesses a special Hessian matrix in the center point $\mathbf{c} = (c^x, c^y, c^z)$ of the sub-domain Ω. Such a new polynomial h can be defined as the combination

$$h = kf + lg, \tag{1}$$

where k and l are linear polynomials and $(x, y, z) \in \Omega$

$$\begin{aligned} k(x, y, z) &= a + k_1(x - c^x) + k_2(y - c^y) + k_3(z - c^z), \\ l(x, y, z) &= b + l_1(x - c^x) + l_2(y - c^y) + l_3(z - c^z). \end{aligned}$$

The zero level set of the polynomial h

$$\mathcal{Z}(h, \Omega) = \{\mathbf{x} : h(\mathbf{x}) = 0 \ \wedge \ \mathbf{x} \in \Omega\}$$

is a surface, which contains the algebraic curve defined by f and g

$$\mathcal{K} = \mathcal{C}(f, g, \Omega) \subseteq \mathcal{Z}(h, \Omega).$$

We choose the coefficients of k and l such that the Hessian of h is a scalar multiple of the identity matrix in the center of the domain $\mathbf{c}$

$$\mathrm{Hess}(h)(\mathbf{c}) = \begin{pmatrix} \lambda & 0 & 0 \\ 0 & \lambda & 0 \\ 0 & 0 & \lambda \end{pmatrix}, \quad \lambda \in \mathbb{R}. \tag{2}$$

If such an h can be computed, then the quadratic Taylor expansion of h about $\mathbf{c}$ has a spherical zero level set. In order to find h, we solve a linear system with eight variables (the coefficients of k and l) and five equations, that can be deducted from (2)

$$\left.\begin{aligned} h_{xx}(\mathbf{c}) - h_{yy}(\mathbf{c}) &= 0 \\ h_{yy}(\mathbf{c}) - h_{zz}(\mathbf{c}) &= 0 \\ h_{xy}(\mathbf{c}) &= 0 \\ h_{yz}(\mathbf{c}) &= 0 \\ h_{xz}(\mathbf{c}) &= 0 \end{aligned}\right\} \tag{3}$$

where

$$h_{uv} = \frac{\partial h}{\partial u \partial v}, \quad u, v \in \{x, y, z\}.$$

If the system has full rank, then the solution set in the space of coefficients of k and l is three-dimensional. Therefore we choose two coefficients as parameters in advance. More precisely, we suppose that the value of the constant term of the polynomials k and l are arbitrary but fixed $(a, b) \in \mathbb{R}^2$ and different from zero ($a \neq 0$ and $b \neq 0$).

Lemma 1. *Given two polynomials f and g over the domain $\Omega \subset \mathbb{R}^3$. We suppose that*

$$\|\nabla f(\mathbf{c}) \times \nabla g(\mathbf{c})\| \neq 0 \tag{4}$$

holds in the center $\mathbf{c}$ of the domain. Then for any pair of $(a, b) \in \mathbb{R}^2$, where $a \neq 0$ and $b \neq 0$, there exist an exactly one-dimensional family of non-trivial polynomials, k and l, such that $h = kf + lg$ satisfies (3).

Proof. The Hessian matrix of h can be expressed with the help of f, g, k and l as

$$\begin{aligned}\mathrm{Hess}(h)(\mathbf{c}) = \nabla k(\mathbf{c})\nabla f(\mathbf{c})^{\mathrm{T}} + \nabla f(\mathbf{c})\nabla k(\mathbf{c})^{\mathrm{T}} + a\,\mathrm{Hess}(f)(\mathbf{c}) \\ +\nabla l(\mathbf{c})\nabla g(\mathbf{c})^{\mathrm{T}} + \nabla g(\mathbf{c})\nabla l(\mathbf{c})^{\mathrm{T}} + b\,\mathrm{Hess}(g)(\mathbf{c}).\end{aligned} \tag{5}$$

For any values of the parameters $a \neq 0$ and $b \neq 0$ the system (3) can be reformulated as

$$\mathbf{A}\mathbf{k} = \begin{pmatrix} f_x(\mathbf{c}) & -f_y(\mathbf{c}) & 0 & g_x(\mathbf{c}) & -g_y(\mathbf{c}) & 0 \\ 0 & f_y(\mathbf{c}) & -f_z(\mathbf{c}) & 0 & g_y(\mathbf{c}) & -g_z(\mathbf{c}) \\ f_y(\mathbf{c}) & f_x(\mathbf{c}) & 0 & g_y(\mathbf{c}) & g_x(\mathbf{c}) & 0 \\ 0 & f_z(\mathbf{c}) & f_y(\mathbf{c}) & 0 & g_z(\mathbf{c}) & g_y(\mathbf{c}) \\ f_z(\mathbf{c}) & 0 & f_x(\mathbf{c}) & g_z(\mathbf{c}) & 0 & g_x(\mathbf{c}) \end{pmatrix} \begin{pmatrix} k_1 \\ k_2 \\ k_3 \\ l_1 \\ l_2 \\ l_3 \end{pmatrix} = \mathbf{b}, \tag{6}$$

where the vector of constants is

$$\mathbf{b} = -a \begin{pmatrix} \frac{1}{2}(f_{xx}(\mathbf{c}) - f_{yy}(\mathbf{c})) \\ \frac{1}{2}(f_{yy}(\mathbf{c}) - f_{zz}(\mathbf{c})) \\ f_{xy}(\mathbf{c}) \\ f_{yz}(\mathbf{c}) \\ f_{xz}(\mathbf{c}) \end{pmatrix} - b \begin{pmatrix} \frac{1}{2}(g_{xx}(\mathbf{c}) - g_{yy}(\mathbf{c})) \\ \frac{1}{2}(g_{yy}(\mathbf{c}) - g_{zz}(\mathbf{c})) \\ g_{xy}(\mathbf{c}) \\ g_{yz}(\mathbf{c}) \\ g_{xz}(\mathbf{c}) \end{pmatrix}.$$

In order to be certain that the system (6) has a one-dimensional solution set, we have to show that the matrix $\mathbf{A}$ has rank 5. Therefore we analyze the 5×5 sub-matrices of $\mathbf{A}$. We denote with $\mathbf{A}_i$ the matrix, which we derive from $\mathbf{A}$ by deleting ith column. The determinants of the matrices $\mathbf{A}_4$, $\mathbf{A}_5$ and $\mathbf{A}$ are

$$\begin{aligned}\det(\mathbf{A}_4) &= -f_x(\mathbf{c})\|\nabla f(\mathbf{c}) \times \nabla g(\mathbf{c})\|^2, \\ \det(\mathbf{A}_5) &= f_y(\mathbf{c})\|\nabla f(\mathbf{c}) \times \nabla g(\mathbf{c})\|^2, \\ \det(\mathbf{A}_6) &= -f_z(\mathbf{c})\|\nabla f(\mathbf{c}) \times \nabla g(\mathbf{c})\|^2.\end{aligned}$$

We know that $\|\nabla f(\mathbf{c}) \times \nabla g(\mathbf{c})\| \neq 0$. This also implies, that one of the coordinates of $\nabla f(\mathbf{c})$, $f_x(\mathbf{c}), f_y(\mathbf{c})$ or $f_z(\mathbf{c})$ is non-zero. Consequently, that one of the

determinants of $\mathbf{A}_4, \mathbf{A}_5$ or $\mathbf{A}_6$ is not zero. So $\mathbf{A}$ always has a full rank 5. The solution of the system $\mathbf{Ak} = \mathbf{b}$ exists, and it is a one-dimensional subspace in $\mathbb{R}^6$. □

According to Lemma 1, for any pair of (a, b) where $a \neq 0$ and $b \neq 0$, there exists a one-parameter family of polynomials k and l, such that $kf + lg$ satisfies (3). From this family of polynomials we always choose the one, which minimizes the Euclidean norm

$$\|\mathbf{k}\|_2 \rightarrow \min \text{ subject to } \mathbf{Ak} = \mathbf{b}, \tag{7}$$

where $\mathbf{k} = (k_1, k_2, k_3, l_1, l_2, l_3)$ is the common coefficient vector of k and l. This guarantees that the solution behaves as numerically well as possible during the computations.

Given the polynomials f and g, a value of (a, b) and a center point $\mathbf{c}$ of the domain Ω, we define the function

$$\mathcal{F}(f, g, (a, b), \mathbf{c}) = h = kf + lg \tag{8}$$

according to the construction in Lemma 1 and the assumption (7).

Remark 1. Suppose that the right-hand side of the system (6), i.e. the vector $\mathbf{b}$, vanishes for a certain pair of (a, b). In this case the solution set of (6) is a line in $\mathbb{R}^6$, which passes through the origin. Then the linear combination $af + bg$ fulfills the condition (2). According to (7) we always choose the solution of the system (6), which has the smallest length. In this special case, both k and l are constants.

The polynomial $h = \mathcal{F}(f, g, (a, b), \mathbf{c})$ fulfills the special condition for the Hessian (2). Thus the quadratic Taylor expansion of h about $\mathbf{c}$ has a spherical zero level set

$$T^2_{\mathbf{c}} h(\mathbf{x}) = h(\mathbf{c}) + \nabla h(\mathbf{c})^T(\mathbf{x} - \mathbf{c}) + \frac{1}{2} h_{xx}(\mathbf{c})(\mathbf{x} - \mathbf{c})^T(\mathbf{x} - \mathbf{c}), \quad \forall \mathbf{x} \in \Omega. \tag{9}$$

If we compute two polynomials for two different pairs of parameters $(a, b) \neq (a', b')$

$$\hat{f} = \mathcal{F}(f, g, (a, b), \mathbf{c}) \quad \text{and} \quad \hat{g} = \mathcal{F}(f, g, (a', b'), \mathbf{c}), \quad \text{such that} \quad a, b, a', b' \neq 0,$$

then their quadratic Taylor expansions about $\mathbf{c}$ can be denoted by

$$p = T^2_{\mathbf{c}} \hat{f} \quad \text{and} \quad q = T^2_{\mathbf{c}} \hat{g}.$$

These two polynomials define the algebraic set

$$\mathcal{S} = \mathcal{C}(p, q, \Omega) = \{\mathbf{x} : p(\mathbf{x}) = 0 \wedge q(\mathbf{x}) = 0 \wedge \mathbf{x} \in \Omega\}. \tag{10}$$

If this algebraic set is not empty and $p \neq q$, then it forms a circular arc. This arc can be used as an approximating circular arc of the curve $\mathcal{K} = \mathcal{C}(f, g, \Omega)$. Later on the error of the approximation is estimated by a distance bound of the algebraic curves $\hat{\mathcal{K}} = \mathcal{C}(\hat{f}, \hat{g}, \Omega)$ and $\mathcal{S}$.

3 Convergence of Approximating Arcs

We analyze in this section the behavior of the generated approximating arcs and its convergence properties.

3.1 Connection with the Osculating Circle

Now we suppose that the center of the computational domain Ω is a point of the algebraic curve $\mathcal{K} = \mathcal{C}(f, g, \Omega)$ defined by the polynomials f and g and that it is not an inflection point of the curve. If the center point is denoted by $\mathbf{c}$, then

$$f(\mathbf{c}) = g(\mathbf{c}) = 0. \tag{11}$$

This special case plays an important role during the computations, since later we would like to approximate the curve in such sub-domains of the original domain, which tightly enclose the algebraic curve.

For an arbitrary pair of parameters we compute a new polynomial as the combination of f and g as we defined in (8)

$$h = \mathcal{F}(f, g, (a, b), \mathbf{c}).$$

Consider the quadratic polynomial

$$s = T^2_{\mathbf{c}} h.$$

According to the assumption (11) the center of the domain is a point of the zero set of h and s

$$h(\mathbf{c}) = s(\mathbf{c}) = af(\mathbf{c}) + bg(\mathbf{c}) = 0. \tag{12}$$

Then the quadratic approximating polynomial s takes the following form

$$s(\mathbf{x}) = \nabla h(\mathbf{c})^{\mathrm{T}}(\mathbf{x} - \mathbf{c}) + \lambda(\mathbf{x} - \mathbf{c})^{\mathrm{T}}(\mathbf{x} - \mathbf{c}), \tag{13}$$

where

$$\nabla h(\mathbf{c}) = a\nabla f(\mathbf{c}) + b\nabla g(\mathbf{c}), \tag{14}$$

and

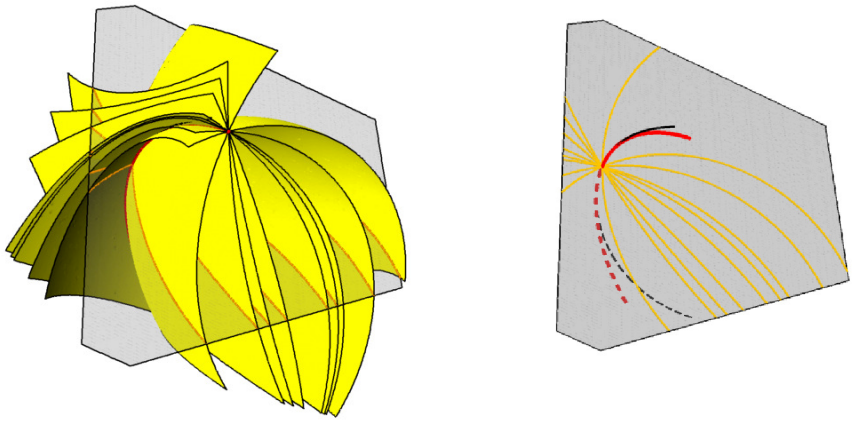
$$\mathrm{Hess}(h)(\mathbf{c}) = \lambda \mathbf{I}^3,$$

as in (2). Suppose that $\lambda \neq 0$, then $s = 0$ can be written in the form

$$\left\langle \mathbf{x} - \left(\mathbf{c} + \frac{1}{\lambda}\nabla h(\mathbf{c}) \right), \mathbf{x} - \left(\mathbf{c} + \frac{1}{\lambda}\nabla h(\mathbf{c}) \right) \right\rangle = \frac{|\nabla h(\mathbf{c})|^2}{\lambda^2}, \tag{15}$$

which is an equation of a sphere. Therefore the radius of this sphere can be computed as

$$r = \frac{|\nabla h(\mathbf{c})|}{\lambda}. \tag{16}$$



(a) (b)

Fig. 1. Sphere family computed with Taylor expansion modification about a point on the algebraic curve (a) and its intersection with the normal plane of the curve (b). The thin, black curve is the algebraic curve. The red circle is the osculating circle.

Remark 2. The zero set of s defined in (13) depends only on the ratio of the chosen parameters a and b. Therefore the sphere family, computed for different values of (a, b), is a one-parametric surface family. It can be parametrized by the ratio of a and b. This follows from the computational method of k and l and the special form of the sphere equations (13). Fig. 1 (a) visualizes several members of such a sphere family for different values of a/b.

Lemma 2. *We assume that* (11) *is satisfied in the point* $\mathbf{c}$*. Then for each sphere equation, computed for a certain* $(a, b) \in \mathbb{R}^2$*,* $a, b \neq 0$*, the center of the sphere* $s = 0$ *lies in the normal plane of the algebraic curve in the point* $\mathbf{c}$*. Moreover the inverse of the radius of the sphere is exactly the normal curvature* κ_n *of the tangent direction* $\nabla f(\mathbf{c}) \times \nabla g(\mathbf{c})$ *of the surface* $\mathcal{F}(f, g, (a, b), \mathbf{c})$ *in the point* $\mathbf{c}$*.*

Proof. Suppose that in a certain neighborhood of the point $\mathbf{c}$ the algebraic curve can be parametrized with arc length parametrization. It is not a restriction, since we are computing only with the regular segments of the algebraic curve. The parametrization is denoted by

$$\mathbf{p}(t), \quad \text{where} \quad \mathbf{p}(t_0) = \mathbf{c}.$$

This curve lies on the surface $h = 0$ according to the definition, therefore it satisfies

$$\frac{d^i h(\mathbf{p}(t))}{dt^i} = 0,$$

for any i. If we compute the first derivative in the point $\mathbf{c}$, we obtain that

$$\left.\frac{dh(\mathbf{p}(t))}{dt}\right|_{t=t_0} = \langle \nabla h(\mathbf{c}), \mathbf{p}'(t_0)\rangle = 0.$$

Thus the tangent vector of the algebraic curve is parallel to the cross product of the gradients $\nabla f(\mathbf{c})$ and $\nabla g(\mathbf{c})$. In (14) we observed, that

$$\nabla h(\mathbf{c}) = a\nabla f(\mathbf{c}) + b\nabla g(\mathbf{c}).$$

Since s is the quadratic Taylor expansion of h about $\mathbf{c}$, we obtain that

$$\langle \nabla s(\mathbf{c}), \mathbf{p}'(t_0)\rangle = 0.$$

This implies, that for any values of the parameters (a, b) the gradient of the associated sphere is in the normal plane of the algebraic curve in the point $\mathbf{c}$.

The second derivative is

$$\left.\frac{d^2h(\mathbf{p}(t))}{dt^2}\right|_{t=t_0} = \langle \nabla h(\mathbf{c}), \mathbf{p}''(t_0)\rangle + \mathbf{p}'(t_0)^{\mathrm{T}}\mathrm{Hess}(h)(\mathbf{c})\mathbf{p}'(t_0) =$$

$$= \langle \nabla h(\mathbf{c}), \mathbf{p}''(t_0)\rangle + \lambda\langle \mathbf{p}'(t_0), \mathbf{p}'(t_0)\rangle = 0.$$

Since we used the arc length parametrization, therefore

$$\langle \nabla h(\mathbf{c}), \mathbf{p}''(t_0)\rangle + \lambda = 0.$$

The polynomial s is the quadratic Taylor expansion of h about $\mathbf{c}$, therefore also

$$\langle \nabla s(\mathbf{c}), \mathbf{p}''(t_0)\rangle = -\lambda.$$

If we expand the scalar product, then we get

$$|\nabla s(\mathbf{c})||\mathbf{p}''(t_0)|\cos\varphi = -\lambda,$$

where φ denotes the angle of the surface normal $\nabla h(\mathbf{c}) = \nabla s(\mathbf{c})$ and the normal direction of the algebraic curve in $\mathbf{c}$. According to the Theorem of Meusnier and (16) we finally arrive at

$$\kappa\cos\varphi = \kappa_n = \frac{\lambda}{|\nabla s(\mathbf{c})|} = \frac{1}{r},$$

which proves the lemma. □

As an example, Fig. 1 (b) shows the intersection of the sphere family and the normal plane of the algebraic curve. Each sphere of the family intersects this plane in a great circle. These circles intersect each other in two points on the normal of the algebraic space curve. The second intersection point is not shown in the Figure.

Corollary 1. *The functions f and g define an algebraic curve $\mathcal{K} = \mathcal{C}(f, g, \Omega)$ in $\Omega \subset \mathbb{R}^3$. We assume that the point $\mathbf{c} \in \Omega$ lies on the algebraic curve $\mathbf{c} \in \mathcal{K}$. We compute the function family $h(a, b) = \mathcal{F}(f, g, (a, b), \mathbf{c})$ with special Hessian for f and g in the point $\mathbf{c}$. The quadratic Taylor expansion for any (a, b) pair $a, b \neq 0$ has a spherical zero level set. The intersection of this sphere family is a circle, which is the osculating circle of $\mathcal{K}$ in the point $\mathbf{c}$.*

Proof. In each point of a curve on a surface, the osculating circle is the normal section of the curvature sphere of the surface [19]. In Lemma 2 we observed, that this curvature sphere for any $h(a, b) = 0$ surface is the zero set of the quadratic Taylor expansion. These spheres have the same intersection curve with the osculating plane of $\mathcal{K}$ in the point $\mathbf{c}$, which is exactly the osculating circle.□

3.2 Convergence of Approximating Circles

The method, which is described in Sect. 2, generates an approximation of the intersection curve of two algebraic surfaces f and g in the sub-domain $\Omega \subseteq \Omega_0$ by a circular arc. This approximating arc is defined as the intersection curve of two spheres. These spheres are given as the zero level set of two polynomials p and q. The polynomials are the quadratic Taylor expansion of certain polynomials with a special Hessian about the center point $\mathbf{c}$ of the domain Ω. The special polynomials are computed as the combination of the functions f and g in the form $kf + lg = \mathcal{F}(f, g, (a, b), \mathbf{c})$ for certain pair $a, b \neq 0$. Note that $\mathbf{c}$ is not required to lie on the curve segment.

In order to prove the convergence of the generated circles, we have to show that the computed polynomials depend continuously on the points of Ω_0 for a fixed choice of (a, b). It means, that the polynomial $\mathcal{F}(f, g, (a, b), \mathbf{c})$ depends continuously on the choice of the point $\mathbf{c}$.

Lemma 3. *Given two polynomials $f, g \in C^2$ over the domain Ω_0. We suppose that for any point $\mathbf{c} \in \Omega_0$*

$$\|\nabla f(\mathbf{c}) \times \nabla g(\mathbf{c})\| \neq 0. \tag{17}$$

For an arbitrary but fixed pair of a and $b \in \mathbb{R} \setminus \{0\}$ we compute the polynomial

$$h = \mathcal{F}(f, g, (a, b), \mathbf{c})$$

with a special Hessian (see in Lemma 1) under the condition (7)*. Then h depends continuously on the points of the domain Ω_0.*

Proof. We have to show that the computed linear factors k and l depend continuously on the point $\mathbf{c}$. We computed the coefficient vector $\mathbf{k} = (k_1, k_2, k_3, l_1, l_2, l_3)$, such that it satisfies the linear system $\mathbf{A}\mathbf{k} = \mathbf{b}$ in (6) and minimizes the l_2-norm of the vector $\mathbf{k}$ (see (7)). If (17) is true, then $\mathbf{A}$ has full rank in any point $\mathbf{c} \in \Omega_0$. For a full rank matrix the vector, which satisfies (6) and (7), can be computed as

$$\mathbf{k} = \underbrace{\mathbf{A}^{\mathrm{T}}(\mathbf{A}\mathbf{A}^{\mathrm{T}})^{-1}}_{\mathbf{A}^{\dagger}}\mathbf{b}.$$

The matrix $\mathbf{A}^\dagger$ is the so called Moore-Penrose generalized inverse of $\mathbf{A}$ (see [20]). If $f, g \in C^2$, then the matrix $\mathbf{A}$ and the vector $\mathbf{b}$ depend continuously on the point $\mathbf{c}$. Therefore the vector $\mathbf{k}$ also depends continuously on the point $\mathbf{c}$. The values of $a \neq 0$ and $b \neq 0$ are fixed real numbers. So all coefficients a, b, k_i, $i = 1 \dots 3$ and l_i, $i = 1 \dots 3$ depend continuously on $\mathbf{c}$. Therefore also $kf + lg$ depends continuously on the point $\mathbf{c}$. □

The next corollary follows from Corollary 1 and Lemma 3. If we modify the Taylor expansion as described in Sect. 2, then we can establish a result concerning the behavior of a sequence of the generated approximating circles.

Corollary 2. *Suppose we have a nested sequence of sub-domains* $(\Omega_i)_{i=1,2,3\dots} \subset \Omega_0$

$$\Omega_{i+1} \subset \Omega_i,$$

which have decreasing diameters δ_i*, such that*

$$\lim_{i \to \infty} \delta_i = 0,$$

and $\mathbf{c}_i$ *denotes the center point of* Ω_i*. Consider a pair of functions* f *and* g*, which defines an algebraic curve in* $\Omega_0 \subset \mathbb{R}^3$

$$\mathcal{K}_0 = \mathcal{C}(f, g, \Omega_0) = \{\mathbf{x} : f(\mathbf{x}) = 0 \wedge g(\mathbf{x}) = 0 \wedge \mathbf{x} \in \Omega_0\}.$$

Suppose that there exists a point $\mathbf{p}$*, which satisfies* $f(\mathbf{p}) = g(\mathbf{p}) = 0$ *and* $\mathbf{p} \in \Omega_i$ *for all* i*. We compute* $\hat{f}_i = \mathcal{F}(f, g, (a, b), \mathbf{c}_i)$ *and* $\hat{g}_i = \mathcal{F}(f, g, (a', b'), \mathbf{c}_i)$ *for fixed values of* $a, b, a', b' \neq 0$*. We consider the circles* $\mathcal{S}_i$ *defined by the zero set of the quadratic Taylor expansions* $p_i = T^2_{\mathbf{c}_i} \hat{f}_i$ *and* $q_i = T^2_{\mathbf{c}_i} \hat{g}_i$*. Then the sequence of these circles converges to a limit circle, which is the osculating circle of* $\mathcal{K}_0$ *in the point* $\mathbf{p}$*.*

The following corollary guarantees, that the local algebraic reformulation of the polynomials asymptotically does not influence the shape of the algebraic curve (no additional curve segment arises).

Corollary 3. *We assume that the parameters* a, b, a' *and* b' *are all non-zero and satisfy* $ab' \neq a'b$ *and that* $\mathbf{c}$ *varies in the compact domain* Ω_0*. There exists a constant* $d > 0$ *such that if the diameter* δ_Ω *of* $\Omega \subset \Omega_0$ *satisfies* $\delta_\Omega < d$ *and* $\mathbf{c} \in \Omega$ *then*

$$\mathcal{C}(f, g, \Omega) = \mathcal{C}(\hat{f}, \hat{g}, \Omega), \tag{18}$$

where $\hat{f} = \mathcal{F}(f, g, (a, b), \mathbf{c})$ *and* $\hat{g} = \mathcal{F}(f, g, (a', b'), \mathbf{c})$*.*

Proof. We write the algebraic reformulation as

$$\begin{pmatrix} \hat{f} \\ \hat{g} \end{pmatrix} = \begin{pmatrix} k_1 \; l_1 \\ k_2 \; l_2 \end{pmatrix} \begin{pmatrix} f \\ g \end{pmatrix} = \mathbf{L_c} \begin{pmatrix} f \\ g \end{pmatrix}.$$

Algorithm 1 `ArcLocal3d` $(f, g, \Omega, \varepsilon)$

Require: The curve is regular in Ω.
1: $\mathbf{c} \leftarrow$ center of Ω
2: $\hat{f} = \mathcal{F}(f, g, (a, b), \mathbf{c}), \quad \hat{g} = \mathcal{F}(f, g, (a', b'), \mathbf{c}), \quad a, b, a', b' \in \mathbb{R} \setminus \{0\}$
3: $\mathcal{S} \leftarrow$ zero set of $T^2_{\mathbf{c}} \hat{f}$ and $T^2_{\mathbf{c}} \hat{g}$ {approximating circle}
4: **if** $\mathcal{S} \neq \emptyset$ **then**
5: $G \leftarrow$ lower bound for $\|\nabla \hat{f}\|$ and $\|\nabla \hat{g}\|$
6: $K \leftarrow$ upper bound for $|\nabla \hat{f} \cdot \nabla \hat{g}|$
7: **if** $0 < G$ and $0 < G^2 - K$ **then**
8: $\varrho \leftarrow$ upper bound of $\mathrm{HD}_\Omega(\mathcal{S}, \mathcal{C}(\hat{f}, \hat{g}, \Omega))$ {see Sect. 4.2}
9: **if** $\varrho \leqslant \varepsilon$ **then**
10: **return** $\mathcal{S} \cap \Omega$ {approximating arc has been found}
11: **end if**
12: **end if**
13: **end if**
14: **return** $\emptyset$ {no approximating arc has been found}

If $\det(\mathbf{L}_{\mathbf{c}}(\mathbf{x})) \neq 0$ for all $\mathbf{x} \in \Omega$ then (18) is satisfied. (Note that this is merely a sufficient condition, but not a necessary one.) We know that

$$\mathbf{L}_{\mathbf{c}}(\mathbf{c}) = ab' - a'b \neq 0.$$

According to Lemma 3 the linear polynomials $k_{1,2}$ and $l_{1,2}$ depend continuously on $\mathbf{c}$ and $\mathbf{x}$. Thus also $\nabla k_{1,2}$ and $\nabla l_{1,2}$ change continuously. Hence there exists a global upper bound B for all $\mathbf{x}$ and $\mathbf{c} \in \Omega_0$ such that

$$\|\nabla \det(\mathbf{L}_{\mathbf{c}}(\mathbf{x}))\| < B.$$

Consequently, $\det(\mathbf{L}_{\mathbf{c}}(\mathbf{x})) \neq 0$ is satisfied for all domains Ω containing $\mathbf{c}$ whose diameter does not exceed $|ab' - ba'|/(2B)$. □

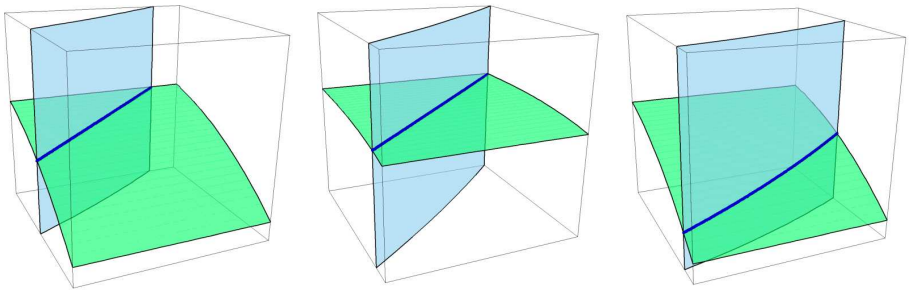
One might introduce an additionally test certifying that $\det(\mathbf{L}_{\mathbf{c}})$ does not vanish in Ω. We omitted this test since we never experienced unwanted branches in our numerical experiments.

4 Algorithm and Error Bounds

Based on the results of Sect. 2 we formulate Algorithm 1. It generates an approximation of regular algebraic curve segments by circular arcs and a bound of the approximation error.

4.1 Local Algorithm

The local method is applied in such sub-domains of the original computational domain, which contains regular curve segments. Later on we will describe a global algorithm, which combines this local algorithm with subdivision method.



(a) (b) (c)

Fig. 2. Examples for local approximating arc generation with the help of `ArcLocal3d` for the parameter pairs $(a, b) = (1, 2)$ and $(a', b') = (2, 1)$.

Table 1. Arc approximation of three different curve segments (see Fig. 2) for four different parameter choices.

$(a, b); (a', b')$	Curve (a)	Curve (b)	Curve (c)
$(1, 2); (2, 1)$	0.01526	0.01184	0.01771
$(1, 5); (5, 1)$	0.01528	0.01177	0.01774
$(1, 100); (100, 1)$	0.01561	0.01151	0.01827
$(1, 1000); (1000, 1)$	0.01566	0.01150	0.01829

Remark 3. The algorithm `ArcLocal3d` first reformulates the polynomials. Therefore we have to fix the value of two parameter pairs (a, b) and (a', b'). According to Remark 2 if $a, b, a', b' \neq 0$ and $a/b \neq a'/b'$, then any choice of these parameter pairs generate similar results, since the generated approximating arcs converge to the same limit circle, the osculating circle. It is not possible to improve the general behavior of the algorithm by the choice of these parameters.

Fig. 2 presents three curve segments approximated by arcs with the help of the local algorithm for the parameter pairs $(a, b) = (1, 2)$ and $(a', b') = (2, 1)$. We approximate the same curve segments by three other choice of the parameter pairs in the unit cube. Table 1 compares the error bounds (see bounding method in Sect. 4.2) of these approximations. In all remaining examples, which will be presented in Section 5.2, we chose the parameters as $(1, 2)$ and $(2, 1)$.

The algorithm generates the approximating arc in algebraic form. Clearly, if we would like to represent the output in a parametric form, it is also possible to describe the circular arcs as rational quadratic curves.

The error estimation method (described in Sect. 4.2) is based on the convex hull property of polynomials represented in Bernstein-Bézier form. This technique is used to bound the distance between the zero level set of each polynomial and the associated quadratic Taylor expansion. Then an upper bound is

generated for the one-sided Hausdorff distance of the approximating arc and the algebraic curve.

The algorithm is successful, if the approximating arc is found and the error bound is smaller then the prescribed tolerance ε. In this case the algorithm returns a circular arc, which approximates the curve segment in the appropriate sub-domain Ω. If the local algorithm fails then it returns the empty set.

4.2 Error Estimate

We describe a method here to estimate the distance of two algebraic space curves. In order to get a distance bound we combine a distance bound of parametric and algebraic curves and a distance estimation strategy between algebraic surfaces.

In order to bound the distance of algebraic and parametric curves we recall a result from [21]. We assume that the parametric curve segment $\mathbf{s}(t)$ is defined with the parameter domain $t \in [0,1]$ in $\Omega \subset \mathbb{R}^3$. The curve traces the point set

$$\mathcal{S} = \{\mathbf{s}(t) \,:\, t \in [0,1]\}.$$

The algebraic curve $\mathcal{K} = \mathcal{C}(f,g,\Omega)$ is defined as the simultaneous zero set of the polynomials f and g. The one-sided Hausdorff distance of $\mathcal{S}$ with respect to $\mathcal{K}^* = \mathcal{K} \cup \partial\Omega$ is defined as

$$\mathrm{HD}_\Omega(\mathcal{S},\mathcal{K}) = \sup_{t\in[0,1]} \inf_{\mathbf{x}\in\mathcal{K}^*} \|\mathbf{x} - \mathbf{s}(t)\|. \tag{19}$$

The boundary $\partial\Omega$ in (19) is needed for technical reasons, see [21].

Theorem 1 ([21]). *Suppose that the polynomials f and g define the algebraic curve $\mathcal{K} = \mathcal{C}(f,g,\Omega)$ in $\Omega \subset \mathbb{R}^3$. We assume that positive constants G and K exist for all $\mathbf{x} \in \Omega$, such that*

$$G \le \|\nabla f(\mathbf{x})\| \quad \textit{and} \quad G \le \|\nabla g(\mathbf{x})\|,$$

and

$$|\nabla f(\mathbf{x}) \cdot \nabla g(\mathbf{x})| \le K.$$

If

$$G^2 - K > 0,$$

and provided that there exist a positive constant M, such that

$$\forall t \in [0,1], \quad f(\mathbf{s}(t))^2 + g(\mathbf{s}(t))^2 \le M^2,$$

the one-sided Hausdorff distance is bounded by

$$\mathrm{HD}_\Omega(\mathcal{S},\mathcal{K}) \le \frac{M}{\sqrt{G^2 - K}}. \tag{20}$$

Theorem 1 can be used for bounding the distance of implicitly defined algebraic curves. The Bernstein-Bézier(BB) norm denoted as $\|.\|^{\Omega}_{\mathrm{BB}}$, is the maximum absolute value of the coefficients in the BB-form of the polynomial represented in the domain Ω. With the help of this norm we can bound the distance of algebraic surfaces over a certain computational domain $\Omega \subset \mathbb{R}^3$. The distance bound can be defined between an arbitrary polynomial f and an approximating polynomial p for all point in the domain

$$\varepsilon = \|f - p\|^{\Omega}_{\mathrm{BB}}. \tag{21}$$

Due to the convex hull property

$$|f(\mathbf{x}) - p(\mathbf{x})| \leq \varepsilon, \quad \forall \mathbf{x} \in \Omega.$$

Therefore for all $\mathbf{x} \in \Omega$ such that $p(\mathbf{x}) = 0$

$$\|f(\mathbf{x})\| \leq \varepsilon. \tag{22}$$

Suppose that an algebraic curve $\mathcal{K}$ is defined by the polynomials f and g

$$\mathcal{K} = \mathcal{C}(f, g, \Omega) = \{\mathbf{x} : f(\mathbf{x}) = 0 \wedge g(\mathbf{x}) = 0 \wedge \mathbf{x} \in \Omega\}.$$

An approximating space curve $\mathcal{S}$ is given by two approximating algebraic surfaces $p = 0$ and $q = 0$

$$\mathcal{S} = \mathcal{C}(p, q, \Omega) = \{\mathbf{x} : p(\mathbf{x}) = 0 \wedge q(\mathbf{x}) = 0 \wedge \mathbf{x} \in \Omega\}.$$

In order to estimate the distance of algebraic space curves we measure first the distance of the defining algebraic surfaces. Suppose that the polynomial p approximates f, and q is an approximating polynomial of g. We estimate the distance between the algebraic surfaces and the approximating surfaces pairwise

$$\varepsilon_1 = \|f - p\|^{\Omega}_{\mathrm{BB}}, \quad \varepsilon_2 = \|g - q\|^{\Omega}_{\mathrm{BB}}.$$

According to (22) for all $\mathbf{x} \in \mathcal{S}$

$$|f(\mathbf{x})| \leq \varepsilon_1 \quad \text{and} \quad |g(\mathbf{x})| \leq \varepsilon_2,$$

thus

$$\sqrt{f(\mathbf{x})^2 + g(\mathbf{x})^2} \leq \sqrt{\varepsilon_1^2 + \varepsilon_2^2}.$$

Therefore Theorem 1 can be applied to bound the distance of $\mathcal{K}$ and $\mathcal{S}$ with the help of the constants G, K and

$$M = \sqrt{\varepsilon_1^2 + \varepsilon_2^2}.$$

In order to compute the constants G, K, ε_1 and ε_2, we represent the polynomials in Bernstein-Bézier form and use the convex hull property of the representation form.

Remark 4. This error bound and the method for computing it can be extended to the two-sided Hausdorff distance. The role of the polynomials f, g and p, q can be exchanged in the bound (21). In this case, the values of the algebraic distances (ε_1 and ε_2) do not change. The bounds on the gradients G and K will change. However, both bounds converge to the same value when the domain Ω shrinks to a point. This follows from the construction of p and q, which satisfy

$$\nabla f(\mathbf{c}) = \nabla p(\mathbf{c}) \quad \text{and} \quad \nabla g(\mathbf{c}) = \nabla q(\mathbf{c})$$

in the center $\mathbf{c}$ of the computational domain.

5 Global Subdivision Method

Subdivision is a frequently used technique and it is often combined with local approximation methods. Such hybrid algorithms subdivide the computational domain in order to separate regions where the topology of the curve can be described easily. The local curve approximation techniques can be applied in the sub-domains, where the topology of the curve has been successfully analyzed. The regions with unknown curve behavior can be made smaller and smaller with subdivision.

5.1 Global Algorithm

The algorithm `GenerateArcs` (see Algorithm 2) generates approximating arcs for general algebraic space curves. It combines the arc generation for regular curve segments `ArcLocal3d` (see Algorithm 1) with recursive subdivision.

First it analyzes the Bernstein–Bézier coefficients of the polynomials with respect to the current sub-domain. If no sign changes are present for one or both of the polynomials, then the current sub-domain does not contain any component of the algebraic curve. Otherwise, if the curve is regular within the domain, it tries to apply the local arc generation algorithm. If this is not successful, then the algorithm either subdivides the current domain into eight sub-domains, or returns the entire domain, if its diameter is already below the user-defined threshold ε.

Note that the algorithm may return sub-domains that do not contain any segments of the implicitly defined curve. However, it is guaranteed to return a set of approximating primitives, such that each point of the implicitly defined curve has at most the distance ε to one of the primitives.

The generated approximating arcs are generally disconnected. It follows from the approximation technique described in the local algorithm `ArcLocal3d`. This technique always considers information only about the algebraic curve segment, which is located in the computational sub-domain. Due to the user-specified error bound ε, the end points of two neighboring approximating arcs along the curve cannot be farther away from each other than 2ε. However a post processing step can be applied in the end of the global algorithm to connect the approximating

Algorithm 2 `GenerateArcs`$(f, g, \Omega, \varepsilon)$

if the box is empty **then**
  return $\emptyset$
end if
if the curve is regular in Ω **then**
  $\mathcal{A} \leftarrow$ `ArcLocal3d`$(f, g, \Omega, \varepsilon)$ {arc generation}
  if $\mathcal{A} \neq \emptyset$ **then**
    return $\mathcal{A}$ {... has been successful}
  end if
end if
if diameter of $\Omega > \varepsilon$ **then**
  subdivide the box into 8 subboxes $\Omega_1, \ldots, \Omega_8$ {subdivision}
  return $\bigcup_{i=1}^{8}$ `GenerateArcs`$(f, \Omega_i, \varepsilon)$ {recursive call}
end if
return Ω {current box is small enough}

arcs, which represent the same segment of the curve. For instance based on the error bound, one can apply approximation techniques, which generate continuous curves within tolerance bands [7].

5.2 Examples

Example 1. In this example we approximate the intersection curve of quadric surfaces. We apply the algorithm `GenerateArcs` for three different intersections of four different pairs of quadric surfaces. The outputs are represented in Fig. 3. The number of used approximating primitives are given in Tab. 2 for each intersection curve. If the curve has a singular point (here in 1.(b), 2.(c), 3.(b) and 4.(c)), then the algorithm returns not only arcs but also bounding boxes as approximating primitives. All the examples are represented in the unit cube $[0, 1]^3$. The intersection curves are approximated within the tolerance $\varepsilon = 0.01$.

According to the local approximation technique the global algorithm generates only bounding boxes around the singular points of the curve. If singular points are present, then our method generates bounding boxes, but no arcs. More sophisticated techniques are required for dealing with singular points, see e.g. [26]. Our further aim is to develop such computational tests based on arc approximation.

Example 2. In this example we approximate the intersection curve of non-quadratic surface patches by two different methods. One is using the circular arc generation technique combined with subdivision. The other technique generates only approximating line segments as local approximating primitives and combines it with iterative subdivision. This second method also reformulates the algebraic system locally. It generates two different linear combination of the algebraic equations such that the new polynomials have perpendicular normal vector in the center of the computational sub-domain. The approximating line

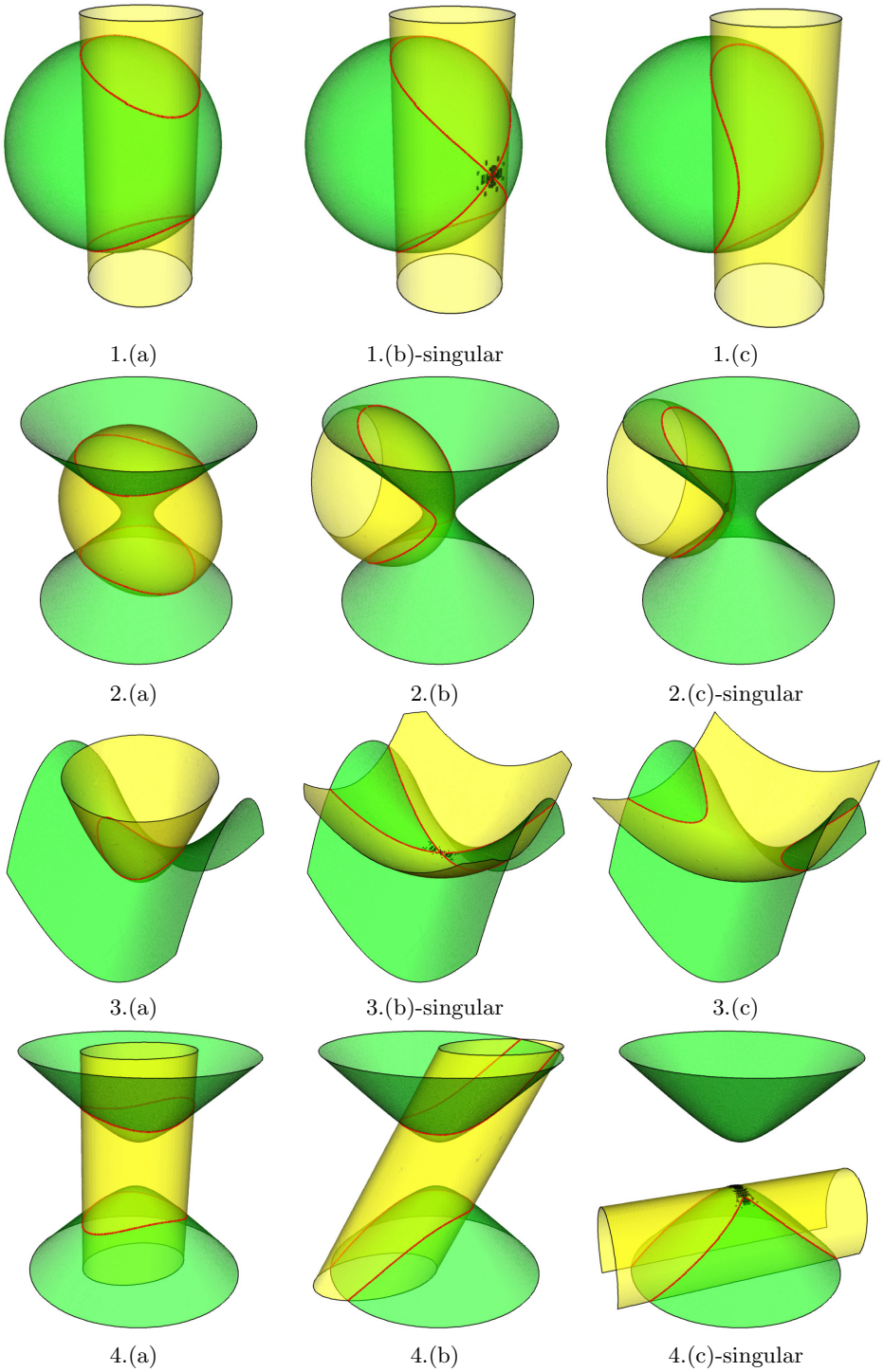


Fig. 3. Approximation of the intersection of quadric surfaces.

Table 2. Approximating intersection curve of quadric surfaces. The number of used approximating primitives are given for the examples shown in Fig. 3.

Quadric Surfaces	Position (see Fig. 3)	Number of Arcs	Number of Boxes
1. sphere + cylinder	(a)	80	0
	(b)-singular	104	248
	(c)	52	0
2. ellipsoid + hyperboloid of one sheet	(a)	80	0
	(b)	76	0
	(c)-singular	96	76
3. rotational paraboloid + hyperbolic paraboloid	(a)	60	0
	(b)-singular	108	156
	(c)	50	0
4. hyperboloid of two sheets + elliptic cylinder	(a)	80	0
	(b)	80	0
	(c)-singular	88	612

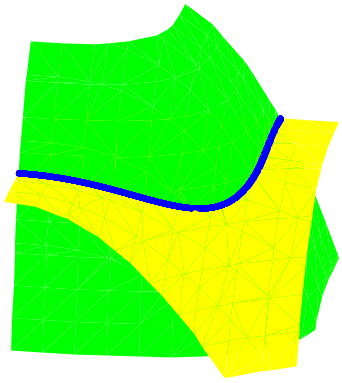


Fig. 4. Approximation of a high-order algebraic space curve by circular arcs (69 segments) and line segments (278 segments), both with tolerance $\varepsilon = 10^{-4}$. Only one picture is shown, since there are no visual differences.

segment is defined then as the intersection curve of the linear Taylor expansions of the reformulated polynomials.

The example surfaces are defined as the zero level set of the polynomials

$$\begin{aligned} 2x^4 + y^3 + z - 1.1 \\ x^3y^2 + z - 0.6 \end{aligned} \tag{23}$$

in the unit cube. The curve is approximated within the tolerance $\varepsilon = 10^{-4}$. The line approximation uses 278 line segments while the arc generation method only 69 arcs to reach the same tolerance level (see Fig. 4).

Example 3. In this example we approximate the isophotes of surfaces for different light directions. Isophotes are curves on a surface, where all points are exposed with equal light intensity from a given light source. An isophote of a surfaces $f = 0$ for a fixed direction vector $\mathbf{d}$ and angle φ traces the point set

$$\mathcal{I}(f, \mathbf{d}, \varphi) = \{\mathbf{p} : f(\mathbf{p}) = 0 \wedge \langle \mathbf{d}, \nabla f(\mathbf{p}) \rangle = \cos(\varphi) \|\nabla f(\mathbf{p})\|\},$$

Table 3. Number of used approximating primitives (in columns # Arcs) in the isophote approximations (see examples in Fig.5 and Fig.6).

$\mathcal{S}_1 : xy - z + 0.5 = 0$						$\mathcal{S}_2 : x^3 + \frac{1}{2}y^3 + z - \frac{1}{2} = 0$			
$(0, 0, -1)$		$(-1, 1, -4)$		$(-2, 0, -3)$		$(-1, -1, -1)$		$(0, -1, -1)$	
$\cos\varphi$	# Arcs	$\cos\varphi$	# Arcs	$\cos\varphi$	# Arcs	$\cos\varphi$	# Arcs	$\cos\varphi$	# Arcs
0.8	**66**	0.7	**19**	0.5	**15**	0.6	**28**	0.3	**16**
0.85	**44**	0.8	**25**	0.65	**18**	0.7	**32**	0.4	**32**
0.9	**48**	0.88	**56**	0.8	**28**	0.75	**58**	0.5	**44**
0.95	**32**	0.95	**54**	0.9	**22**	0.8	**107**	0.7	**70**
0.99	**28**	0.99	**26**	0.97	**31**	0.85	**120**	0.99	**79**

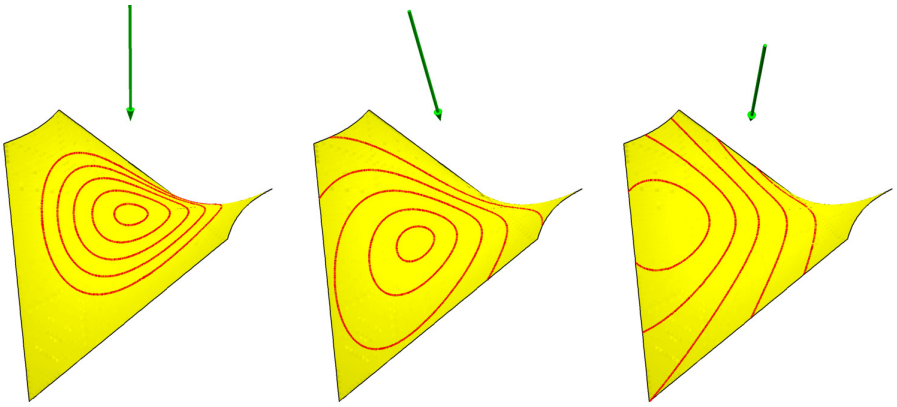


Fig. 5. Approximation of isophotes for different light directions on the surface $\mathcal{S}_1$.

if we suppose that the direction vector is a unit vector. In order to describe an isophote for a given $\mathbf{d}$ and φ we used the algebraic equation system

$$\begin{aligned} f &= 0, \\ (f_x d^x + f_y d^y + f_z d^z)^2 - \cos^2\varphi \left(f_x^2 + f_y^2 + f_z^2\right) &= 0, \end{aligned}$$

where $\mathbf{d} = (d^x, d^y, d^z)$. These two equations allocates the points of the isophotes, which belong to the direction $\mathbf{d}$ and the angles φ and $(\pi - \varphi)$. We apply the arc approximation algorithm to two different surfaces to compute isophotes. For the first one we compute isophotes for three different light directions (see Fig. 5). For the second surface we used two different light directions (see Fig. 6). In Tab. 3 we show the number of used approximating arcs for each isophote for both surfaces, along with the light directions and angles. We approximated the isophotes in the domain $[-1, 1]^3$ within the tolerance $\varepsilon = 0.05$.

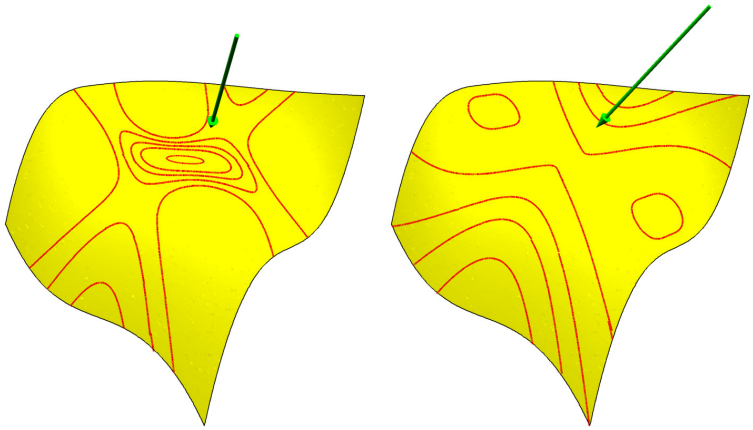


Fig. 6. Approximation of isophotes for different light directions on the surface $\mathcal{S}_2$.

6 Conclusion

We presented a local technique which generates approximating circular arcs for regular segments of algebraic space curves. This technique is based on a reformulation of the problem. It combines the defining polynomials of the algebraic curve, such that the new polynomials define the same algebraic curve but they have special Hessian matrices at a certain point. This local technique can be combined with subdivision method to approximate arbitrary algebraic curves restricted to a domain.

A similar method can be applied also for planar algebraic curves. Moreover it might be possible to use a similar arc generation technique in $\mathbb{R}^n$ to approximate general algebraic curves. This result can be used also for finding roots of polynomial systems [17]. Computing with quadratic approximating primitives provides good convergence properties [22]. Thus it is promising to apply then for curve approximation and for solving polynomial systems.

References

1. Patrikalakis, N.M., Maekawa, T.: Shape interrogation for computer aided design and manufacturing. Springer (2002)
2. Wang, W., Joe, B., Goldman, R.: Computing quadric surface intersections based on an analysis of plane cubic curves. Graphical Models **64** (2002) 335–367
3. Wang, W., Goldman, R., Tu, C.H.: Enhancing Levin's method for computing quadric surface intersections. Computer Aided Geometric Design **20** (2003) 401–422
4. Chau, S., Oberneder M., Galligo A., Jüttler B.: Intersecting biquadratic surface patches. In Piene, R., Jüttler, B. eds.: Geometric Modeling and Algebraic Geometry. Springer (2008) 161–180

5. Tu, C.H., Wang, W., Mourrain, B., Wang, J.Y.: Using signature sequences to classify intersection curves of two quadrics. Computer Aided Geometric Design **26** (2009) 317–335
6. Dupont, L., Lazard, D., Lazard, S., Petitjean, S.: Near-optimal parameterization of the intersection of quadrics. SCG '03: Proceedings of the nineteenth annual symposium on computational geometry. ACM (2003) 246–255
7. Held, M., Eibl, J.: Biarc approximation of polygons within asymmetric tolerance bands. Computer-Aided Design **37** (2005) 357–371
8. Hoschek, J., Lasser, D.: Fundamentals of Computer Aided Geometric Design. AK Peters (1993)
9. Pratt, M.J., Geisow, A.D.: Surface/surface intersection problem. In Gregory, J., ed.: The Mathematics of Surfaces II. Claredon Press, Oxford (1986) 117–142
10. Patrikalakis, N.M.: Surface-to-Surface Intersections. IEEE Comput. Graph. Appl. **13**(1) (1993) 89–95
11. Bajaj, C.L., Hoffmann C.M., Hopcroft J.E., Lynch R.E.: Tracing surface intersections. Comput. Aided Geom. Des. **5**(4) (1988) 285–307
12. Krishnan, S., Manocha, D.: An efficient surface intersection algorithm based on lower-dimensional formulation. ACM Trans. Graph. **16**(1) (1997) 74–106
13. Gonzalez-Vega, L., Necula, I.: Efficient topology determination of implicitly defined algebraic plane curves. Comput. Aided Geom. Design **19**(9) (2002) 719–743
14. Daouda, D.N., Mourrain, B., Ruatta, O.: On the computation of the topology of a non-reduced implicit space curve. Proc. Int. Symp. on Symbolic and Algebraic Computation, ACM (2008) 47-54.
15. Alcazar, J.G., Sendra, J.R.: Computation of the topology of real algebraic space curves. Journal of Symbolic Computation **39**(6) (2005) 719–744
16. Liang, C., Mourrain, B., Pavone, J.-P.: Subdivision methods for the topology of 2d and 3d implicit curves. In Piene, R., Jüttler, B. eds.: Geometric Modeling and Algebraic Geometry. Springer (2008) 199–214
17. Mourrain, B., Pavone, J.P.: Subdivision methods for solving polynomial equations. Journal of Symbolic Computation **44**(3) (2009) 292–306
18. Elber, G., Kim, M.: Geometric constraint solver using multivariate rational spline functions. Proc. Symp. on Solid Modeling and Applications, ACM (2001), 1-10
19. Kreyszig, E.: Differential Geometry. Dover (1991)
20. Cline, R. E., Plemmons, R. J.: l_2-Solutions to Undetermined Linear Systems. SIAM Review **18**(1) (1976) 92–106
21. Jüttler, B., Chalmovianský, P.: A predictor–corrector technique for the approximate parameterization of intersection curves. Appl. Algebra Eng. Comm. Comp. **18** (2007) 151–168
22. Sederberg, T.W., White, S.C., Zundel, A.K.: Fat arcs: a bounding region with cubic convergence. Comput. Aided Geom. Des. **6**(3) (1989) 205–218
23. Sabin, M.A.: The use of circular arcs to form curves interpolated through empirical data points. Technical Report VTO/MS/164, British Aircraft Corporation, (1976)
24. Meek, D.S., Walton, D.J.: Approximating smooth planar curves by arc splines. Journal of Computational and Applied Mathematics **59** (1995) 221–231
25. Song, X., Aigner M., Chen, F., Jüttler B.: Circular spline fitting using an evolution process. Journal of Computational and Applied Mathematics **231** (2009) 423-433
26. Mantzaflaris, A., Mourrain, B.: Deflation and Certified Isolation of Singular Zeros of Polynomial Systems unpublished: http://hal.inria.fr/inria-00556021/en (2011)