1988

Applying Algebraic Geometry to Surface Intersection Evaluation

Christoph M. Hoffmann
Purdue University, cmh@cs.purdue.edu

Report Number:
88-772

APPLYING ALGEBRAIC GEOMETRY TO SURFACE INTERSECTION EVALUATION

Christoph M. Hoffmann*

Computer Sciences Department
Purdue University
Technical Report CSD-TR-772
CAPO Report CER-88-26
June, 1988

* Research supported in part by NSF grants CCR-8619817 and ONR contract N0014-86-K-0465, and a grant from the AT&T Foundation.
† Computer Science Department, Cornell University, Ithaca, NY 14853, USA. Supported in part by NSF Grant DMC 86-17355, and ONR Contract N00014-86-K-0281.

Applying Algebraic Geometry to Surface Intersection Evaluation

Christoph M. Hoffmann*

Notes for a course on Algebraic Geometry
at SIGGRAPH 1988

Abstract

Focusing on surface intersection tracing, we review the familiar numerical approach. We demonstrate that it actually applies more generally than commonly assumed in the literature. We then study how to combine it with methods from algebraic geometry in an attempt to overcome the shortcomings of purely numerical techniques.

These notes provide a guide to the practicing engineer for seeing how his familiar tools relate to the larger picture. They provide a hands-on guide for understanding and applying some of the more esoteric methods proposed by algebraic geometry, and assess their practical potential.

Keywords: Parametric and implicit surfaces, surface intersection, plane and space curves, singularities. Numerical analysis, least-squares problems, singular value decomposition. Taylor series, Newton's theorem, birational maps, projection. Algebraic geometry, symbolic computation, Gröbner bases, monoids, resultants.

*Computer Science Department, Purdue University, West Lafayette, Ind. 47907. Supported in part by NSF Grant CCR 86-19817, ONR Contract N00014-86-K-0465, and a grant from the ATT Foundation.

1 Introduction

Evaluating the intersection of two surfaces is a recurring operation in geometric modeling and in computer graphics, [23]. This operation is performed repeatedly as part of the conversion from Constructive Solid Geometry (CSG) to Boundary Representation (B-rep). It is used when implementing Boolean operations on B-rep objects. And it is used when rendering curved surfaces in order to locate contours.

Surface intersection evaluation is not an easy problem, and continues to be an active topic of research. Some of the reasons for this continued activity are not hard to identify: A good surface intersection technique has to balance three conflicting goals: *efficiency*, *robustness*, and *accuracy*. Typically, a numerical algorithm is efficient, but is not fully robust. Algorithms based on exact arithmetic are fully robust and accurate, but are slow. We suspect that these demands cannot be met simultaneously without compromise, and that they must be negotiated judiciously, as appropriate for the application at hand. These notes discuss some of the tools that could be used to strike a reasonable balance.

In his thesis [15], Geisow distinguishes four separate methods for evaluating the intersection of two surfaces in three space:

1. Trace the intersection by a marching scheme: Beginning at a starting point p, a local approximation of the intersection is constructed, typically the curve tangent at p. By stepping along the approximant a specific distance, an estimate of a next curve point is obtained that is then refined using iteration.

2. Map the intersection to a plane curve $f(u, v) = 0$, e.g., through elimination of variables from a system of algebraic equations, and then evaluate f.

3. Subdivide and approximate the surfaces by planar facets. Then construct a piecewise linear approximation of the intersection curve.

4. Evaluate the surfaces at discrete points in three space. By interpolation of the resulting function values, localize the intersection.

Of these four techniques, we will consider here the first two, representing, so to speak, opposite extremes: While intersection evaluation by tracing is

typically a purely numerical technique, the map from a three dimensional space curve to a plane curve involves symbolic computation. We thus consider to what extent the two techniques can be combined and integrated, to overcome specific weaknesses they exhibit as separate methods.

We structure these notes as follows: Beginning with the purely numerical approach, we review the relevant formulas needed to implement a marching scheme that constructs a local curve approximant of arbitrary degree. We show how to interpret some of the choices arising, in terms of intrinsic properties, such as arc length, curvature, and torsion.

Next, we show how the numerical formulation can be applied to construct *directly* higher order curve approximants to the intersection of two parametric surfaces. This obviates the need to stick with piecewise linear approaximants constructed by the many subdivision techniques, e.g., [6]. Our technique also side-steps attempts to locally approximate the intersecting surfaces with higher order approximants, which appears to be costly and difficult [4].

The intersection of two algebraic surfaces, whether given implicitly or parametrically, is an algebraic space curve. Every such curve can be mapped *birationally*, i.e., bijectively except for finitely many points, onto a plane algebraic curve. Postponing for a moment how to construct this map, we show how to reliably analyze all singularities of plane algebraic curves, and how to use this knowledge so as to trace *through* any curve singularity. Among the issues to be settled for this are locating an impending sigularity, transforming the curve so as to resolve the singularity, and preserve the direction of tracing.

We then discuss ways for mapping a surface intersection to a plane algebraic curve. Some of these methods are natural, viz., when intersecting a parametric and an implicit surface. Others require fairly substantial symbolic computations, effecting an elimination of variables in a system of algebraic equations. These computations can be carried out through resultant computations or through Gröbner base techniques. Since the latter are fairly unknown in the modeling community, yet could contribute valuable ideas, we discuss Gröbner base techniques in a separate section.

There is a progression in these notes from the tried-and-true to the highly experimental: Sections 2 and 3, following [5], apply the mature technology from numerical analysis. Although not fully robust, these methods work

very flexibly in many situations, incuding nodal curve singularities. Section 4, also adapted from [5], consider the planar case and explore augmenting the numerical approach with symbolic computation capable of resolving all types of singularities. This combination of numerical and symbolic computation offers many promises for increasing the scope of curve tracing.

By now, the community is well aware of resultant computations, thanks to works such as [25]. Repeated resultant computations, however, have the problem of introducing extraneous factors that can complicate variable elimination. Gröbner base techniques, on the other hand, do not have this shortcoming and are relatively unknown. So, in Section 5, we give a brief survey of some of the algorithms possible using Gröbner bases, and how they apply to the surface intersection problem. This material is adapted from the excellent surveys [7, 9], and from Buchberger's lecture [8]. Finally, in Section 6, we consider the reduction of space curve tracing to plane curve tracing. Here, some highly experimental methods are surveyed, including [16, 3, 14].

2 Numerical Tracing of Implicit Surface Intersection

We address the following problem:

Given two implicit surfaces, $f(x,y,z) = 0$ and $g(x,y,z) = 0$, and a point $p = (p_x, p_y, p_z)$ on their intersection. Trace the intersection curve, beginning at p.

We recall that the *gradient* of f at the point p is the vector

$$\nabla f = (f_x(p), f_y(p), f_z(p))$$

where the subscripts denote partial derivates. Moreover, $\times$ denotes the cross product of vectors. Unless p is an isolated point, we need to clarify in which *direction* the trace is to be done. Since there is no natural intrinsic orientation of the intersection $f \cap g$, we will arbitrarily assume at p that the direction $\nabla f \times \nabla g$ is the *positive* tracing direction, whereas $\nabla g \times \nabla f$ would be the *negative* tracing direction, at p. We will see in Section 4 that this convention is *local*, that is, in the presence of singularities tracing may have to alternate between locally positive and locally negative directions.

Throughout this section we assume that at each point p the surface gradients ∇f and ∇g are linearly independent. That is, the intersection curve is regular at p. If the gradients vanish or are linearly dependent, then the curve has a singularity at p and this approach cannot be used without considerable enhancements.

2.1 Structure of the Method

The tracing procedure developed here repeats the following steps:

1. At the curve point p, construct a local approximant $\mathbf{r}(s)$, to some order.
2. Using this approximant and a step value s_0, determine the next point $q = \mathbf{r}(s_0)$.
3. By Newton iteration, bring q closer to the intersection of f and g.

Typically, the order of approximation is fixed. First order approximations use the curve tangent at p as the local approximant. This is often implemented because the tangent is so easy to compute, as $\mathbf{t} = \nabla f \times \nabla g$. Higher order approximants allow larger steps. There is a trade-off between the added time needed to compute a higher order approximant, and the time saved by the ability to take larger steps. Degree three approximants seem to provide a good balance: The determination of the approximant is not too costly, and contains both the local curvature and torsion.

2.2 The Curve Approximant at p

We write the intersection of f and g as a vector function $\mathbf{r}$, parameterized by s, with $p = \mathbf{r}(0)$:

$$\mathbf{r}(s) = \mathbf{r}(0) + s\mathbf{r}'(0) + \frac{s^2}{2}\mathbf{r}''(0) + \frac{s^3}{6}\mathbf{r}'''(0) + \cdots$$

The approximant is determined by finding the components of the derivatives of $\mathbf{p}$. Technically, this involves the formulation of a linear system, that always has the following structure:

$$\begin{aligned} \nabla f \cdot \mathbf{r}^{(m)}(0) &= b_{f,m} \\ \nabla g \cdot \mathbf{r}^{(m)}(0) &= b_{g,m} \end{aligned} \tag{1}$$

The coefficients $b_{f,m}$ and $b_{g,m}$ depend on the partial derivatives of f and g at p. Since the system is underdetermined, there is not a unique solution, and we must make certain choices. These choices will be interpreted geometrically, and will result in an approximant whose parameter s is the arc length and whose derivatives are explicitly related to curvature and torsion at p.

2.2.1 Setting up the Linear System

We determine the derivative values, $\mathbf{r}'(0)$, $\mathbf{r}''(0)$, $\mathbf{r}'''(0)$, from the partial derivatives of f and of g. By Taylor's theorem, we have

$$f(x,y,z) = f(x_0+\Delta x, y_0+\Delta y, z_0+\Delta z) = \sum_{i,j,k} f_{i,j,k}\Delta x^i \Delta y^j \Delta z^k$$

with the partials

$$f_{i,j,k} = \frac{1}{i!j!k!}\frac{\partial^{i+j+k}}{\partial x^i \partial y^j \partial z^k} f(x_0,y_0,z_0).$$

We set

$$\begin{aligned}
\Delta x &= \mathbf{r}'_x s + \mathbf{r}''_x s^2/2 + \mathbf{r}'''_x s^3/6 + \cdots \\
\Delta y &= \mathbf{r}'_y s + \mathbf{r}''_y s^2/2 + \mathbf{r}'''_y s^3/6 + \cdots \\
\Delta z &= \mathbf{r}'_z s + \mathbf{r}''_z s^2/2 + \mathbf{r}'''_z s^3/6 + \cdots
\end{aligned}$$

where $\mathbf{r}'_x$ denotes the x-component of the vector $\mathbf{r}'$, etc. Then we have

$$\begin{aligned}
(\Delta x)^2 &= (\mathbf{r}'_x)^2 s^2 + \mathbf{r}'_x \mathbf{r}''_x s^3 + \cdots \\
(\Delta x)^3 &= (\mathbf{r}'_x)^3 s^3 + \cdots \\
\Delta x \Delta y &= \mathbf{r}'_x \mathbf{r}'_y s^2 + (\mathbf{r}''_x \mathbf{r}'_y + \mathbf{r}'_x y''_x) s^3/2 + \cdots \\
\Delta x \Delta y \Delta z &= \mathbf{r}'_x \mathbf{r}'_y \mathbf{r}'_z s^3 + \cdots,
\end{aligned}$$

and so on. Substituting into the Taylor series for f and equating to zero the coefficients of s^m, where $m = 1, 2, 3$, we get the equations

$$f_{1,0,0}\mathbf{r}'_x + f_{0,1,0}\mathbf{r}'_y + f_{0,0,1}\mathbf{r}'_z = b_{f,1}$$

$$f_{1,0,0}\mathbf{r}''_x + f_{0,1,0}\mathbf{r}''_y + f_{0,0,1}\mathbf{r}''_z = b_{f,2}$$

$$f_{1,0,0}\mathbf{r}'''_x + f_{0,1,0}\mathbf{r}'''_y + f_{0,0,1}\mathbf{r}'''_z = b_{f,3}$$

where

$$b_{f,1} = 0,$$

$$\begin{aligned} b_{f,1} = -2[&f_{2,0,0}(\mathrm{r}'_x)^2 + f_{0,2,0}(\mathrm{r}'_y)^2 + f_{0,0,2}(\mathrm{r}'_z)^2 \\ &+ f_{1,1,0}\mathrm{r}'_x\mathrm{r}'_y + f_{1,0,1}\mathrm{r}'_x\mathrm{r}'_z + f_{0,1,1}\mathrm{r}'_y\mathrm{r}'_z], \end{aligned}$$

$$\begin{aligned} b_{f,3} = -6[&f_{2,0,0}\mathrm{r}'_x\mathrm{r}''_x + f_{0,2,0}\mathrm{r}'_y\mathrm{r}''_y + f_{0,0,2}\mathrm{r}'_z\mathrm{r}''_z + f_{1,1,0}(\mathrm{r}''_x\mathrm{r}'_y + \mathrm{r}'_x\mathrm{r}''_y)/2 \\ &+ f_{1,0,1}(\mathrm{r}''_x\mathrm{r}'_z + \mathrm{r}'_x\mathrm{r}''_z)/2 + f_{0,1,1}(\mathrm{r}''_y\mathrm{r}'_z + \mathrm{r}'_y\mathrm{r}''_z)/2 \\ &+ f_{3,0,0}(\mathrm{r}'_x)^3 + f_{0,3,0}(\mathrm{r}'_y)^3 + f_{0,0,3}(\mathrm{r}'_z)^3 \\ &\quad + f_{2,1,0}(\mathrm{r}'_x)^2\mathrm{r}'_y + f_{1,2,0}\mathrm{r}'_x(\mathrm{r}'_y)^2 + f_{2,0,1}(\mathrm{r}'_x)^2\mathrm{r}'_z \\ &\quad + f_{1,0,2}\mathrm{r}'_x(\mathrm{r}'_z)^2 + f_{0,2,1}(\mathrm{r}'_y)^2\mathrm{r}'_z + f_{0,1,2}\mathrm{r}'_y(\mathrm{r}'_z)^2 + f_{1,1,1}\mathrm{r}'_x\mathrm{r}'_y\mathrm{r}'_z]. \end{aligned}$$

In vectorial notation, we can rewrite these equations as

$$\begin{aligned} \nabla f \cdot \mathrm{r}' &= b_{f,1} \\ \nabla f \cdot \mathrm{r}'' &= b_{f,2} \\ \nabla f \cdot \mathrm{r}''' &= b_{f,3} \end{aligned}$$

where the scalars $b_{f,m}$ are the righthand sides. The equations for g are developed analogously. In particular, we have

$$b_{g,1} = 0,$$

$$\begin{aligned} b_{g,2} = -2[&g_{2,0,0}(\mathrm{r}'_x)^2 + g_{0,2,0}(\mathrm{r}'_y)^2 + g_{0,0,2}(\mathrm{r}'_z)^2 \\ &+ g_{1,1,0}\mathrm{r}'_x\mathrm{r}'_y + g_{1,0,1}\mathrm{r}'_x\mathrm{r}'_z + g_{0,1,1}\mathrm{r}'_y\mathrm{r}'_z], \end{aligned}$$

$$\begin{aligned} b_{g,3} = -6[&g_{2,0,0}\mathrm{r}'_x\mathrm{r}''_x + g_{0,2,0}\mathrm{r}'_y\mathrm{r}''_y + g_{0,0,2}\mathrm{r}'_z\mathrm{r}''_z + g_{1,1,0}(\mathrm{r}''_x\mathrm{r}'_y + \mathrm{r}'_x\mathrm{r}''_y)/2 \\ &+ g_{1,0,1}(\mathrm{r}''_x\mathrm{r}'_z + \mathrm{r}'_x\mathrm{r}''_z)/2 + g_{0,1,1}(\mathrm{r}''_y\mathrm{r}'_z + \mathrm{r}'_y\mathrm{r}''_z)/2 \\ &+ g_{3,0,0}(\mathrm{r}'_x)^3 + g_{0,3,0}(\mathrm{r}'_y)^3 + g_{0,0,3}(\mathrm{r}'_z)^3 \\ &+ g_{2,1,0}(\mathrm{r}'_x)^2\mathrm{r}'_y + g_{1,2,0}\mathrm{r}'_x(\mathrm{r}'_y)^2 + g_{2,0,1}(\mathrm{r}'_x)^2\mathrm{r}'_z \\ &+ g_{1,0,2}\mathrm{r}'_x(\mathrm{r}'_z)^2 + g_{0,2,1}(\mathrm{r}'_y)^2\mathrm{r}'_z + g_{0,1,2}\mathrm{r}'_y(\mathrm{r}'_z)^2 + g_{1,1,1}\mathrm{r}'_x\mathrm{r}'_y\mathrm{r}'_z]. \end{aligned}$$

2.2.2 Solving the Linear System Numerically

We have formulated a linear system $A\mathrm{r}^{(m)} = \mathrm{b}_m$ for $m = 1, 2, 3$. Although it is only a 2 by 3 system, solving it without giving attention to the numerical properties will waste accuracy in the solution. Hence, we should carefully choose a numerically stable solution technique. A good choice is to link in a standard technique from one of the numerical analysis packages, e.g., the *singular value decomposition* routines of Linpack, [11].

Briefly, a singular value decomposition of A derives three matrices U, S, and V such that

$$A = VSU^T$$

and the matrices have the following properties:

1. The matrices U and V are orthogonal, i.e., UU^T and VV^T are the identity matrices.
2. The matrix S is diagonal and the diagonal entries are nonnegative and in decreasing magnitude.

In our case, the matrix V is 2 by 2, U is 3 by 3, and S is

$$S = \begin{pmatrix} \sigma_1 & 0 & 0 \\ 0 & \sigma_2 & 0 \end{pmatrix}$$

where $\sigma_1 > \sigma_2 > 0$. The solution of the system is

$$\mathbf{r}^{(m)} = \alpha'_m U_1 + \beta'_m U_2 + \gamma'_m U_3$$

where U_i is the i^{th} column in U, and

$$\begin{aligned} \alpha'_m &= (V^T \mathbf{b}_m)_1/\sigma_1 \\ \beta'_m &= (V^T \mathbf{b}_m)_2/\sigma_2 \end{aligned}$$

The last coefficient γ'_m is arbitrary, and the column vector U_3 is tangent to the curve at $p = \mathbf{r}(0)$.

2.2.3 Choosing γ'_m

The system is underdetermined, and has an infinity of solutions. So, choices must be made for the γ'_m to arrive at a canonical solution, and these choices can be understood geometrically. From differential geometry, we recall that at the point p of a space curve the *accompanying triad* forms a natural local coordinate system. The triad consists of three orthonormal vectors, the *tangent* $\mathbf{t}$, the *principal normal* $\mathbf{n}$, and the *binormal* $\mathbf{b}$. Note that $\mathbf{n} = \mathbf{b} \times \mathbf{t}$. Their directions are defined by the tangent, the curvature, and the twist of the space curve. As the point p moves on the space curve these vectors vary obeying the *Frenet-Serret formulas*, [13]:

$$\frac{d\mathbf{t}}{ds} = \kappa \mathbf{n}, \qquad \frac{d\mathbf{b}}{ds} = -T\mathbf{n}, \qquad \frac{d\mathbf{n}}{ds} = T\mathbf{b} - \kappa \mathbf{t},$$

where s is the arc length, $\kappa = 1/\rho$ is curvature and $T = 1/\tau$ is torsion.

Since $\mathbf{t} = U_3$ is a unit vector, we choose $\gamma_1' = \pm 1$, hence $\mathbf{r}' = \pm U_3$. Next, we choose $\gamma_2' = 0$, so that $\mathbf{r}''$ is perpendicular to $\mathbf{r}'$. In fact, since

$$\mathbf{r}''(s) = \frac{d\mathbf{t}}{ds} = \kappa \mathbf{n},$$

we know that then $\mathbf{r}''$ is collinear with the principal normal and that

$$\kappa = \sqrt{(\alpha_2')^2 + (\beta_2')^2}$$

Finally, we have

$$\mathbf{r}'''(s) = \left[\frac{d}{ds}(\kappa \mathbf{n}) = \frac{d\kappa}{ds}\mathbf{n} + \kappa \frac{d\mathbf{n}}{ds}\right] = \kappa' \mathbf{n} + \kappa T \mathbf{b} - \kappa^2 \mathbf{t}.$$

By orthogonality, therefore, we have $\gamma_3' = -\kappa^2$.

2.3 Step Length

We have constructed an approximant $r(s)$ to the curve at p and now choose a step length s_0 so as to obtain a subsequent curve point estimate $\mathbf{r}(s_0)$. A safe step length requires understanding the radius of convergence of the full Taylor series. This is a nontrivial undertaking, and we opt for a simpler heuristics in which the contribution of the higher order terms to the next point estimate is kept small.

We choose s such that both

$$\|s^2 \mathbf{r}''(0)/2\| \qquad \text{and} \qquad \|s^3 \mathbf{r}'''(0)/6\|$$

are smaller than $\|s\mathbf{r}'(0)\|/10 = |s|/10$. Since the step sizes could become arbitrarily small, a minimum step size should also be specified. In our applications, this strategy has performed very well.

2.4 Newton Approximation

At the point p, we have constructed a third order approximant $\mathbf{r}(s)$, we have determined adaptively a step length s_0, and now have a new point estimate $q = \mathbf{r}(s_0)$. Using Newton iteration, we refine this estimate until we are on the intersection with acceptable accuracy.

The iteration is based on the following, first-order approximation:

$$\begin{aligned} \nabla f(q_k) \cdot \Delta_k &= -f(q_k) \\ \nabla g(q_k) \cdot \Delta_k &= -g(q_k) \end{aligned}$$

where $\Delta_k = (\delta_x, \delta_y, \delta_z)$. Note that this system has the same structure as system (1). The next point estimate is then

$$q_{k+1} = q_k + \Delta_k$$

As in the approximant construction, we solve the linear system using singular value decomposition. For the solution Δ_k, we set the coefficient of U_3 to zero, since it represents lateral movement that will not improve the quality of the new estimate significantly. We continue with the iteration until

$$\|q_{k+1} - q_k\| < 10^{-t}\|q_k\|$$

where t is a precision parameter. Typically, we have $t = -10$ for double-precision floating-point computations and require two or three iterations to achieve this accuracy.

3 Numerical Tracing of Parametric Surface Intersection

The traditional approach to intersecting two parametric surfaces is by subdivision and piecewise linear approximation. See, e.g., [6, 10, 18, 19, 21, 22]. When an initial intersection point p is known, however, the numerical approach above becomes directly applicable.

Let the surfaces be given as

$$\begin{aligned} x &= G_{1,1}(u_1, v_1) \\ y &= G_{1,2}(u_1, v_1) \\ z &= G_{1,3}(u_1, v_1) \end{aligned}$$

and

$$\begin{aligned} x &= G_{2,1}(u_2, v_2) \\ y &= G_{2,2}(u_2, v_2) \\ z &= G_{2,3}(u_2, v_2) \end{aligned}$$

Then the intersection is given by

$$\begin{aligned} F_1(u_1, v_1, u_2, v_2) = G_{1,1}(u_1, v_1) - G_{2,1}(u_2, v_2) &= 0 \\ F_2(u_1, v_1, u_2, v_2) = G_{1,2}(u_1, v_1) - G_{2,2}(u_2, v_2) &= 0 \\ F_2(u_1, v_1, u_2, v_2) = G_{1,3}(u_1, v_1) - G_{2,3}(u_2, v_2) &= 0 \end{aligned}$$

that is, by three equations in the four unknowns u_1, v_1, u_2, and v_2. These equations define a curve $\mathbf{R}(s)$ in four dimensional space:

$$\mathbf{R}(s) = \begin{pmatrix} u_1(s) \\ v_1(s) \\ u_2(s) \\ v_2(s) \end{pmatrix}$$

from which the intersection curve is recovered via

$$\mathbf{r}(s) = \begin{pmatrix} G_{1,1}(u_1(s), v_1(s)) \\ G_{1,2}(u_1(s), v_1(s)) \\ G_{1,3}(u_1(s), v_1(s)) \end{pmatrix} = \begin{pmatrix} G_{2,1}(u_2(s), v_2(s)) \\ G_{2,2}(u_2(s), v_2(s)) \\ G_{2,3}(u_2(s), v_2(s)) \end{pmatrix} \tag{2}$$

We proceed exactly as in the implicit surface case. We construct a third order approximant, solving a 3 by 4 linear system. We determine a safe step length s_0, and refine the next point estimate by Newton iteration. Then the new curve point is mapped to three space using equation (2). See also [5].

4 Tracing Plane Algebraic Curves

We consider how to trace a plane algebraic curve $f(x, y) = 0$, dealing with singularities in a comprehensive manner. Tracing a plane algebraic curve is fundamental because every algebraic space curve can be mapped birationally to a plane algebraic curve. This observation has been used in various ways in the surface intersection problem, e.g., [12], and continues to be researched as a promising approach to intersection evaluation. We will review several methods for mapping a surface intersection to a plane algebraic curve later.

For implicit surface intersection, we have solved numerically a system of two algebraic equations in three unknowns. For parametric surface intersection, we solved a system of three algebraic equations in four unknowns. Clearly the techniques are applicable to solving $n - 1$ equations in n unknowns. With $n = 2$, we have the planar case.

The numerical tracing routine performs very well, except at singularities. Hence we seek methods for dealing with singularities. They will involve transforming the curve f to a different curve g such that g does not have singularities. So, instead of tracing f, we can trace g, and map back the points of g to corresponding points of f.

It would be nice if we only needed to trace g. Unfortunately, g cannot be so used since we would have to pass through infinity. So, we trace f whenever possible and trace only critical segments of f on g, i.e., segments containing singularities.

4.1 Birational Maps

A birational map between, e.g., x, y, z space and u, v space has the form

$$\begin{array}{ll} x = G_1(u,v) & u = H_1(x,y,z) \\ y = G_2(u,v) & v = H_2(x,y,z) \\ z = G_3(u,v) & \end{array}$$

where the G_i and H_i are *rational* functions, i.e., the ratio of two polynomials. Such maps are bijective except for points at which the denominator vanishes. An example is the map between the u, v plane and x, y, z space defining a (rational) parametric surface. The functions G_i determine a surface point for parameter space points, whereas the functions H_j give the inversion of the surface. Birational maps are an important tool for analyzing curves and surfaces since they preserve many intrinsic curve properties, such as curve genus. In this section we will use them to analyze plane curve singularities.

4.2 Place of a Curve

An algebraic curve $f(x,y) = 0$ can be investigated locally by means of power series. At a point $p = (a_0, b_0)$, we can write

$$\begin{array}{rcl} x(s) & = & a_0 + a_1 s + a_2 s^2 + \cdots \\ y(s) & = & b_0 + b_1 s + b_2 s^2 + \cdots \end{array} \tag{3}$$

Then this pair of power series constitutes a *local parameterization*. It is called a *place* of f at p. A place is *regular*, if a_1 and b_1 are not simultaneously zero, otherwise it is *singular*.

The notion of place is more specific than that of a curve point. At ordinary, or *regular*, curve points f has only one place, but at singular points it could have several. The place at an ordinary point is the Taylor series.

Intuitively, at singularities with two or more places, the curve is self intersecting. At singularities at which f has only one place, the place itself is singular. At other singularities, the various places may be, but need not be, singular. In [5], Section 4.1, we have shown that the derivation of the linear equations for the curve approximant at p formally agrees with the concept of place. However, it leads to nonlinear equations which are then more difficult to solve.

4.3 Dealing with Singularities

All purely numerical tracing routines fail at a singular curve point for the same reason: Technically, the routines depend on the underlying assumption that there exists a system of linear equations that determines the local structure of the curve. At a singular point, however, these equations are nonlinear, [5], Sec. 4.1. Rather than dealing directly with nonlinear equations, we plan to exploit a classical result from algebraic geometry that states that every curve $f(x,y) = 0$ can be transformed birationally into a curve $g(x,y) = 0$ that is devoid of singularities. See, e.g., [1]. Thus, we plan to trace g in the vicinity of singular points of f, and map the points of g back to corresponding points of f.

4.3.1 Desingularization Transformations

The birational map effecting a resolution of the singularity is constructed incrementally from two quadratic transformations T_1 and T_2:

$$\begin{aligned} T_1: \quad & x_1 = x \\ & y_1 = y/x \\ T_2: \quad & x_2 = x/y \\ & y_2 = y \end{aligned}$$

T_1 is 1-1 on all (x,y) points with $x = 0$, and maps the points $(0,y)$ with $y \neq 0$ to infinity. As we approach the origin on a curve branch, the limit of the image points is the image of the origin on the branch. This limit depends

on the direction of approach: If the branch has a tangent with slope m at the singularity, then that branch will intersect the y_1 axis at $(0, m)$. Since T_1 maps a branch with the y-axis as tangent to infinity, such a branch must be desingularized using T_2. Likewise, T_1 must be applied to branches each of whose tangent is the x-axis. For the intermediate tangent positions we choose T_1 if the slope of the tangent has magnitude 1 or less; otherwise T_2 is chosen.

4.3.2 Locating the Singularity

When numerically tracing f, an impending singularity is detected from the condition number of the matrix

$$\begin{pmatrix} f_x & -f_y \\ f_y & f_x \end{pmatrix}$$

The singular point is the simultaneous intersection of $f = 0$, $f_x = 0$, and $f_y = 0$, and can be found iteratively or by direct methods.

An iterative approach can be based on a least-squares formulation as follows: Beginning with a nearby curve point p_0, we construct a sequence of points $p_0, p_1, p_2, \ldots$ converging to the singularity. Let $p_{i+1} = p_i + (\delta_x, \delta_y)$. Then we solve the linear system

$$\begin{pmatrix} f_x & f_y \\ f_{xx} & f_{xy} \\ f_{xy} & f_{yy} \end{pmatrix} \begin{pmatrix} \delta_x \\ \delta_y \end{pmatrix} = \begin{pmatrix} -f \\ -f_x \\ -f_y \end{pmatrix}$$

This overconstrained system corresponds to the least-squares problem

$$A^T A \Delta = A^T \mathbf{b},$$

where A is the coefficient matrix of the overconstrained system, $\mathbf{b}$ the right-hand side, and $\Delta = (\delta_x, \delta_y)^T$.

For higher order singularities, higher order partials may also vanish. Thus, if A does not have full rank, we extend the system by adding, for each vanishing partial h, the equation

$$h_x \delta_x + h_y \delta_y = -h$$

In this manner a matrix $A^T A$ of full rank is obtained.

4.3.3 Curve Transformation

We translate the coordinate system so as to bring the singularity to the origin. Since the validity of the quadratic transformations T_1 and T_2 depends on the singularity being exactly at the origin, we must counteract numerical inaccuracy incurred by this coordinate translation. If the partial $f_{x^j y^k}$ vanishes at the singularity p, it follows that f cannot contain the term $cx^j y^k$, $c \neq 0$, after the origin has been translated to p. Hence, all such terms are eliminated.

4.3.4 Direction of Traversal

At nonsingular points, we locally orient f by the tangent vector $(-f_y, f_x)$. At a singularity, curve segments locally belonging to the same analytic branch may be oriented in an opposite direction, necessitating a reversal in the tracing direction. We establish a relationship between the orientation of the curve f and the orientation of its proper transform g with the goal of recognizing when to reverse the tracing direction after having passed through a singular point.

Let $p = (a_0, b_0)$ be a nonsingular point of f, where $a_0 \neq 0$. Let

$$x(s) = a_0 + a_1 s + a_2 s^2 + ...$$
$$y(s) = b_0 + b_1 s + b_2 s^2 + ...$$

be the place of f centered at p. The place defines a branch orientation by increasing s. This orientation agrees with the standard orientation $(-f_y, f_x)$ whenever

$$\text{sign}(f_x) = \text{sign}(b_1) \qquad \text{and} \qquad \text{sign}(f_y) = \text{sign}(-a_1)$$

otherwise it is opposite. At the corresponding point $p_1 = (a_0, b_0/a_0)$ of g, the transformed curve g has the place

$$\begin{aligned} x_1(s) &= x(s) \\ y_1(s) &= y(s)/x(s) = c_0 + c_1 s + c_2 s^2 + ... \end{aligned}$$

Since $x(s) = x_1(s)$, the curve and its transform are oriented the same way, by increasing s. Moreover,

$$\begin{aligned} c_0 &= b_0/a_0 \\ c_1 &= (b_1 a_0 - a_1 b_0)/a_0^2 \end{aligned}$$

and so on. Hence, relating this to the standard orientation by a proportionality constant α, we have

$$\begin{aligned} g_y &= \alpha f_y \\ g_x &= \alpha(x f_x + y f_y)/x^2 \end{aligned}$$

If $\text{sign}(\alpha) = 1$, then both f and g are traced in the same direction, relative to the standard orientation, otherwise we trace g in the opposite direction:

1. We traverse f in the direction $u(-f_y, f_x)$, where $u = 1$ or $u = -1$.
2. When changing over to the proper transform g of f, the sign of α is computed for the two corresponding points p of f and p_1 of g at which we switch.
3. If $\alpha > 0$, the transform g is traversed in the direction $u(-g_y, g_x)$; otherwise, it is traversed in the opposite direction.

The same traversal correlation is established when leaving the vicinity of the singularity, reestablishing the proper traversal direction on f from the traversal direction on g.

5 Polynomial Ideals

We present a collection of techniques for symbolic computation with polynomial ideals, based on the concept of *Grōbner bases*, which have many useful geometric applications.

A surface intersection is naturally described as a set of polynomials. In the case of implicit surfaces, we have two polynomials:

$$\begin{aligned} f(x, y, z) &= 0 \\ g(x, y, z) &= 0 \end{aligned}$$

in the case of parametric surfaces, we have three:

$$\begin{aligned} F_1(u_1, v_1, u_2, v_2) &= 0 \\ F_2(u_1, v_1, u_2, v_2) &= 0 \\ F_3(u_1, v_1, u_2, v_2) &= 0 \end{aligned}$$

However, this description is not unique, since these polynomials can be replaced by others. If a *unique* algebraic representation of a curve is wanted, we must consider infinite sets of polynomials, called *ideals*. Formally, an ideal *generated* by the polynomials $\{f_1, \ldots, f_k\}$ consists of all polynomials of the form

$$\{h_1 f_1 + h_2 f_2 + \cdots + h_k f_k\}$$

where the h_i are arbitrary polynomials. It is written $I[f_1, \ldots, f_k]$.

When given an algebraic curve, the ideal describing it is the set of all polynomials whose *common* zeros are the curve points. For example, the ideal of the unit circle in the x, y plane includes the cylinder $f = x^2 + y^2 - 1$, the plane $g = z$, and any polynomial that contains f or g as a factor. In this example, the ideal is $I[f, g]$. Intuitively, a set of generators suffices to define unambiguously the space curve.

Generator sets are usually not unique. This fact has been exploited in a number of surface intersection methods. For example, [17] computes the intersection of two quadrics f and g by first replacing one of them with a ruled surface $f' = \lambda f + \mu g$, where λ and μ are suitable numbers, and then computes $f' \cap g$. Thus, the generators $\{f, g\}$ are replaced with the generators $\{f', g\}$. Conceivably, then, some generator sets allow better intersection algorithms than others, and we should investigate whether there exist especially 'nice' generator sets.

5.1 Ideal Membership

We seek a solution to the following problem:

Given a polynomial g, is it in the ideal generated by $F = \{f_1, \ldots, f_k\}$, i.e., can it be written in the form $g = h_1 f_1 + h_2 f_2 + \cdots + h_k f_k$, where the h_i are some polynomials.

The problem will be solved by repeatedly rewriting f until f has been simplified to the point where the original question can be answered by inspection. The success of this process depends on specific properties of the generators.

5.1.1 Leading Terms and Orderings

We need a notion of 'this polynomial is simpler than that one.' We develop it from an ordering on terms and by declaring f more complicated than g if the most complicated term in f is later in this ordering than the most complicated term of g.

Let $x_1, \ldots, x_n$ be the variables of the polynomials. A product of the form

$$x_1^{e_1} x_2^{e_2} \cdots x_n^{e_n}$$

is a *power product*. We define a *lexicographic ordering*, written $\prec$, of the power products as follows:

1. $1 \prec x_1 \prec x_2 \prec \ldots \prec x_n$.
2. If $u \prec v$ then $uw \prec vw$ for all power products w.
3. If u and v are not yet ordered by the above rules, then order them lexicographically as strings.

For example, with $n = 2$, setting $x_1 = x$ and $x_2 = y$, we have the following ordering:

$$1 \prec x \prec x^2 \prec \ldots \prec x^k \prec \ldots \prec y \prec xy \prec x^2y \prec \ldots \prec y^2 \prec xy^2 \prec \ldots$$

Every term in a polynomial g consists of a coefficient and a power product. The term whose power product is largest with respect to the ordering $\prec$ is called the *leading term* of g, written $lt(g)$. The leading term consists of the *leading coefficient*, $lcf(g)$, and the *leading power product*, $lpp(g)$.

Assuming the lexicographic ordering, we say that the polynomial f is *simpler* than the polynomial g if $lpp(f) \prec lpp(g)$. From now on, we will use only the lexicographic ordering, although other orderings are possible [7].

5.1.2 Rewriting and Normal Form Algorithms

We are given a polynomial g, and a set of polynomials $F = \{f_1, \ldots, f_k\}$. We plan to rewrite g using the polynomials in F, simplifying it at each step, until it cannot be further simplified. The rewriting is as follows; let $g = lcf(g) \cdot lpp(g) + g_r$ and repeat the following steps:

1. Find a polynomial $f \in F$ such that $u \cdot lpp(f) = lpp(g)$, i.e., $lpp(f)$ divides $lpp(g)$.

2. Replace g with $g_1 = g - b \cdot u \cdot f$, where $b = lcf(g)/lcf(f)$.

3. If no such f exists, then stop; g is now in normal form.

Since all terms in g_1 are simpler than the leading term of f, and since the leading term of f has been cancelled, it follows that g_1 is simpler than g. This also implies that the algorithm will always terminate. Note that the algorithm does not necessarily result in a unique normal form. The reason is that more than one f could be found in Step 1, leading to very different sequences of rewriting steps.

As example, let $F = \{y^2 + x^2 - 1, xy - x^2 + 1\}$, $g = 2y^3 + x^2 - xy^2$, and let $x \prec y$. The leading terms are, respectively, y^2, xy, and $2y^3$. We rewrite g in the following steps:

$$\begin{array}{rcll} g & = & 2y^3 - xy^2 + x^2 & \\ \rightarrow g_1 & = & g - 2y(y^2 + x^2 - 1) & = \quad -xy^2 - 2x^2y + 2y + x^2 \\ \rightarrow g_2 & = & g_1 - (-y)(xy - x^2 + 1) & = \quad -3x^2y + 3y + x^2 \\ \rightarrow g_3 & = & g_2 - (-3x)(xy - x^2 + 1) & = \quad 3y - 3x^3 + x^2 + 3x \end{array}$$

The polynomial g_3 is a normal form of g with respect to F.

5.1.3 Gröbner Bases

We would like to use the following method for deciding whether g is in the ideal generated by F:

1. Rewrite g using F until we have obtained a normal form g_n.

2. If $g_n = 0$, then g is in the ideal generated by F. Otherwise, g is not in the ideal.

The problem with this algorithm is that the conclusion 'g is not in the ideal' is in doubt: Since there may be different sequences to rewrite g, we are not assured that all yield the same normal form. Fortunately, there always exists a set G of polynomials that generates the same ideal as F and that has the property that our algorithm above is correct. Such a set is called a *Gröbner Base* of the ideal $I[F]$.

There are several standard algorithms for constructing a Gröbner base from a given set F. Technically, they all rewrite the input polynomials, simplifying them and adding certain polynomials that are, roughly speaking, least common multiples of the input polynomials. The time required to find the Gröbner base varies a great deal, and we give some figures later on.

For example, consider the ideal generated by

$$F = \{f_1, f_2\} = \{(x-1)^2 + y^2 - 2, (x+1)^2 + y^2 - 2\}$$

where $x \prec y$. We ask whether $x - y$ is in the ideal generated by F. The Gröbner base for the ideal is

$$G = \{-x, y^2 - 1\}$$

The normal form of $x - y$ is y, hence $x - y$ is not in the ideal generated by F. On the other hand, G implies that the points in which the two circles $f_1 = 0$ and $f_2 = 0$ intersect are just the intersection points of the line $x = 0$ with the two parallel lines $(y-1)(y+1) = 0$.

5.2 Solving Algebraic Equations with Gröbner Bases

Gröbner bases can be used to solve systems of algebraic equations. The key to this application is contained in the following theorem [7]:

Theorem: Let F be a set of polynomials in the variables $x_1, ..., x_n$, and G a Gröbner base for the ideal generated by F with respect to the lexicographic ordering based on $x_1 \prec ... \prec x_n$. Then, for all i, the polynomials of the base G that do not contain the variables $x_i, ..., x_n$ generate all polynomials in the ideal $I[F]$ that do not contain the variables $x_i, ..., x_n$.

This theorem implies, roughly, that a Gröbner base is a triangular system of polynomial equations. We use it as follows to solve the system $F = \{f_1 = 0, ..., f_k = 0\}$ of algebraic equations.

1. Construct a Gröbner base G for $I[F]$.
2. If $1 \in G$, then stop: F does not have any solution.
3. If G does not contain a univariate polynomial in x_1 then stop: The solution to F does not consist of a finite set of points.

4. Let g_1 be a polynomial of lowest degree in x_1 only, and let $X_1 = \{\alpha_j\}$ be the roots of g_1.

5. Find a polynomial g_2 in G, with simplest leading term, and consider $g_2(\alpha_j)$, obtained by substituting each root α_j for x_1 in g_2. $g_2(\alpha_j)$ is a polynomial in x_2 only, and we find its roots $\beta_{j,l}$. For each $\alpha_j \in X_1$, we thus obtain several roots $\beta_{j,l}$.

6. We next find a trivariate polynomial, in x_1, x_2, and x_3, with simplest leading term, and substitute all combinations $(\alpha_j, \beta_{j,l})$ for x_1 and x_2, followed by solving for x_3. This is repeated until we have found all assignments to the $x_1, ..., x_n$. Each complete assignment is a solution of the system.

We illustrate this algorithm with two examples.

Example 1: We compute the intersection of the three circular cylinders

$$\begin{aligned} x^2 + y^2 - 1 &= 0 \\ x^2 + z^2 - 1 &= 0 \\ y^2 + z^2 - 1 &= 0 \end{aligned}$$

assuming $x \prec y \prec z$. The Gröbner base is

$$\{x^2 - 2, y^2 - 2, z^2 - 2\}$$

This system is especially simple since it is diagonal. The roots of the first polynomal are $x = \pm 1/\sqrt{2}$. Substitution into later equations is not necessary, and the roots of the second and third equation are $y = \pm 1/\sqrt{2}$ and $z = \pm 1/\sqrt{2}$. So, we have eight different solutions, given by the eight combinations of solutions to the three equations.

Example 2: Consider the cubic curve $f = 28y^3 + 26xy^2 + 28y^2 + 7x^2y + 16xy + 7y + x^3/2 + 3x/2$. We find its singular points by solving the system

$$\{f = 0, f_x = 0, f_y = 0\}$$

where f_x and f_y are the partials by x and y, respectively. With the ordering $y \prec x$ we obtain the Gröbner base

$$\{2y + 1, x\}$$

Hence f has one singular point, at $(0, -1/2)$.

5.3 Applications of Gröbner Bases to Elimination and Inversion

In general, the conversion between implicit and parametric surface representations involves complex computations. If a surface is given parametrically as

$$\begin{aligned} x &= h_1(s,t) \\ y &= h_2(s,t) \\ z &= h_3(s,t) \end{aligned}$$

then its implicit form can be determined by elimination of s and t. Sederberg [24] advocates the use of resultants to do the elimination. The difficulty is that a resultant computes a projection, and therefore introduces extraneous factors. For polynomial functions h_i, the Gröbner base approach achieves simultaneously the elimination of s and t, as well as a surface inversion. We demonstrate the procedure with an example.

Let the surface be given as

$$\begin{aligned} x &= st \\ y &= st^2 \\ z &= s^2 \end{aligned}$$

The elimination approach, with the Sylvester resultant, proceeds by eliminating one of the parameters first, say s,

$$\begin{vmatrix} t & -x \\ t^2 & -y \end{vmatrix} = xt^2 - yt = 0$$

$$\begin{vmatrix} t & -x & 0 \\ 0 & t & -x \\ 1 & 0 & -z \end{vmatrix} = x^2 - zt^2 = 0$$

Next, t is eliminated from the two equations so derived:

$$\begin{vmatrix} x & -y & 0 & 0 \\ 0 & x & -y & 0 \\ -x & 0 & x^2 & 0 \\ 0 & -z & 0 & x^2 \end{vmatrix} = x^6 - x^2y^2z = 0$$

We thus obtain the implicit equation

$$x^2(x^4 - y^2z) = 0$$

The factor x^2 is superfluous and should be deleted.

We now construct the Gröbner base for this surface, using

$$F = \{x - st, y - st^2, z - s^2\}$$

Ordering the variables $z \prec y \prec x \prec t \prec s$, we obtain the Gröbner base

$$\begin{aligned} G = \{ \quad & x^4 - y^2 z, \\ & tx - y, \\ & tyz - x^3, \\ & t^2 z - x^2, \\ & sy - x^2, \\ & sx - tz, \\ & st - x, \\ & s^2 - z\} \end{aligned}$$

In G, the first polynomial, $x^4 - y^2 z$, is the implicit surface form. Note the absence of extraneous factors. Polynomials $tx - y$ and $sy - x^2$ are the first polynomials in the base that introduce the variables t and s, respectively. They provide an *inversion* of the surface, that is, given a point (x, y, z) on the surface, we can determine its parametric coordinates (s, t) from these polynomials. Moreover, the linearity of these two polynomials in t and s implies that the surface parameterization is *faithful*, that is, to each point (x, y, z) there corresponds only one point (s, t) in parameter space.

5.4 Complexity

Construction of a Gröbner base is a potentially time consuming process. We give below the timings for several experiments done on a Symbolics 3650 Lisp machine using the Gröbner base implementation provided with Macsyma 412.45. The first set of experiments demonstrates that the running time for the Gröbner base construction is sensitive to the variable ordering used. Consider the cubic function in Example 2 above. Construction of the Gröbner base took 0.226 sec with the variable ordering $y \prec x$, yielding $\{2y + 1, x\}$ as base. With $x \prec y$, the algorithm took 0.402 sec, yielding $\{-x, -2y - 1\}$.

Gröbner base algorithms assume exact arithmetic and are therefore implemented using rational numbers. Much of the observed time can depend

on the size of the rationals manipulated. Consider again the three cylinder intersection of Example 1 above. For constructing the Gröbner base we order the variables $x \prec y \prec z$. Then the base construction takes 0.09 sec. Next, we rotate the cylinders, about the z axis by an arc of 1, then about the x axis by an arc of 1/2, and finally, again about the z axis by an arc of 1, and construct the Gröbner base for the cylinders in the new position. Now the computation takes 405 sec, i.e., 5000 times as long as in the first position, yielding the base:

$$G_1 = \{h_1(x), y - h_2(x), z - h_3(x)\}$$

where h_1 has degree 8, and h_2 and h_3 both have degree 7. The longer running time is largely due to the huge rationals involved that have enumerators and denominators with a magnitude of about 10^{700}.

6 Mapping Surface Intersections to Plane Curves

A general approach to surface intersection evaluation is to map the surface intersection to a plane curve $g(u, v) = 0$. The approach is appealing for a number of reasons. For one, we have seen in Section 4 that plane curves can be traced through singularities. So far, an analogous process for surface intersections does not exist due to the need to map simultaneously both intersecting surfaces. Note, however, that certain singularities can be treated by locally approximating both surfaces with degree two approximants [20].

On the other hand, the approach has a number of difficulties that cause problems. These include the cost of constructing the map, inaccuracies that might arise in the substitution process, and, finally, the high degree of g, that is in general the product of the surface degrees and leads to numerical difficulties. We discuss several techniques that can be applied in various situations. None of them avoid all the problems mentioned.

6.1 Substitution Maps

In substitution maps we attempt to substitute the parametric form of one surface into the implicit form of the other, thereby obtaining a plane curve in the parameter space of the first surface. If we intersect a parametric with an implicit surface, the cost of constructing the map is just the cost

of doing the substitution. Otherwise, we need to add the cost of converting the representation of one of the surfaces from parametric to implicit, or vice versa. In general, the cost of the representation conversion will dominate.

6.1.1 Intersecting two Parametric Surfaces

When intersecting two parametric surfaces, we must implicitize one of them. Implicitization is always possible, and can be done either by resultant computations or by Gröbner base techniques. The resultant based computations might be somewhat faster, but suffer from the extraneous factor problem.

If the implicitized surface is

$$f(x, y, z) = 0$$

and the parametric surface is

$$\begin{aligned} x &= h_1(u, v) \\ y &= h_2(u, v) \\ z &= h_3(u, v) \end{aligned}$$

then the plane algebraic curve is

$$g(u, v) = f(h_1(u, v), h_2(u, v), h_3(u, v)) = 0$$

We can then trace $g = 0$ in the u, v space, and map each point via the rational functions h_i. See also [12].

6.2 Intersecting two Implicit Surfaces

When intersecting two implicit surfaces, one of them must be parameterized. Unfortunately, not every implicit surface possesses a rational parametric form, so this approach needs to be modified. It is known that the intersection of two implicit surfaces always lies on a parameterizable surface; [26, 16]. That is, given the surfaces

$$\begin{aligned} f(x, y, z) &= 0 \\ g(x, y, z) &= 0 \end{aligned}$$

there is a surface

$$h(x, y, z) = h_1 f + h_2 g = 0$$

that is parameterizable and contains the intersection of f and g. Moreover, the computation for obtaining h is simple, as is its parameterization. Note that h will be a *monoid*, that is, a surface of degree n with a singular point of order $n-1$.

We describe the derivation of h. First, we *homogenize* f and g, obtaining $F(w,x,y,z)$ and $G(w,x,y,z)$. As long as $w \neq 0$, the curve $F \cap G$ is identical to $f \cap g$. We select one of the variables as *main variable*, and rewrite F and G as polynomials in this variable, say w:

$$\begin{aligned} F &= u_n w^n + u_{n-1} w^{n-1} + \cdots + u_1 w + u_0 \\ G &= v_{n'} w^{n'} + v_{n'-1} w^{n'-1} + \cdots + v_1 w + v_0 \end{aligned}$$

Without loss of generality we may assume that $n \geq n' > 1$, and determine the polynomials

$$\begin{aligned} F_1 &= u_n w^{n-n'} G - v_{n'} F \\ G_1 &= (u_0 G - v_0 F)/w \end{aligned}$$

Note that both F_1 and G_1 contain the intersection curve of F and G.

Both F_1 and G_1 have degree at most $n-1$ in w. If one of them is linear in w, then we stop; we have found the desired surface. If neither is linear, then we repeat the calculation using F_1 and G_1 in place of F and G. Since at each step the maximum degree in w is lowered by at least one, the computation derives the desired monoid equation after at most n steps, in the form

$$w H_{m-1}(x,y,z) + H_m(x,y,z) = 0$$

This surface is parameterized by

$$\begin{aligned} w(u,v,s) &= -H_{m-1}(u,v,s)/H_m(u,v,s) \\ x(u,v,s) &= u \\ y(u,v,s) &= v \\ z(u,v,s) &= s \end{aligned}$$

This parameterization is *projective*, that is, (u,v,s) are the coordinates of a two-dimensional projective parameter space. The parametric forms are now substituted into the equation of G and give the desired plane curve, in homogeneous form, which is then dehomogenized and is the desired plane curve.

We illustrate the method with an example. Consider the intersection curve of the cylinder $f = x^2 + (z+1)^2 - 1 = 0$ and the sphere $g = x^2 + y^2 + (z-2)^2 - 4 = 0$. Homogenizing, we obtain $F = x^2 + z^2 + 2zw$ and $G = x^2 + y^2 + z^2 + 4zw$. The intersection curve is an irreducible degree 4 space curve with a nodal singularity at the origin. We select z as the main variable. Accordingly, we compute

$$\begin{aligned} F_1 &= G - F = y^2 + 2zw \\ G_1 &= [(x^2+y^2)F - x^2G]/z = y^2z + 2(y^2 - x^2)w \end{aligned}$$

F_1 is simpler and has the parameterization

$$\begin{aligned} z &= -2u/s^2 \\ w &= u \\ x &= v \\ y &= s \end{aligned}$$

Substitution into G yields the plane curve

$$v^4 + 4u^2(s^2 - v^2) = 0$$

Dehomogenizing with $u = 1$ yields $v^4 - 4(s^2 - v^2) = 0$.

This example is very favorable because the degree of the plane curve is the minimum degree possible. In general, this method will introduce extraneous factors and yield plane curves of higher degree than needed.

6.3 Projection Methods

The second general approach to mapping a space curve to a plane curve is by projection. In principle, the construction of these maps is straightforward, although computation intensive. The main problem, however, is that the point from which to project must be carefully chosen: A poorly chosen point will result in a map that cannot be inverted. Such a projection map would not permit mapping the points of the plane curve back to space curve points, so that the plane curve trace would yield no information.

In [3], a method for projecting implicit surface intersections to plane algebraic curves is proposed. It assumes that the two surfaces intersect transversally, i.e., that the surface gradients are linearly independent, and that the curve itself is irreducible. In that case, a good projection point can be chosen by the following computation:

1. Transform the surface equations by a general linear transformation with symbolic coefficients.

2. Project the intersection by a resultant computation.

3. Choose randomly numeric values for the coefficients and verify that the projection does not degenerate.

Note that Step 3 succeeds with probability one, since the failing assignments correspond to projection directions that lie on a cylinder whose base line is the space curve.

In [3], the method is applied to testing whether a space curve is rational, i.e., whether it has a rational parameterization. Since a projection map must preserve this property, rationality can be tested equivalently by testing the plane curve, [2].

In [14], a different method is proposed for projecting implicit surface intersections to plane algebraic curves. It removes the requirement that the two surfaces must intersect transversally in an irreducible space curve, and is based on a classical theorem that says that all but finitely many points on a line are good projection points provided the line does not intersect the space curve. Hence, there are two parts to the algorithm:

1. Find a suitable line.

2. Pick a point on it that works.

The first part is done constructively, showing that specific choices will work. The second part requires first formulating the projection equations symbolically, from a point χ on the line, and then assigning a value to its coordinates that yields a certain square-free polynomial.

An interesting aspect of the algorithm is that various questions about the curve structure can be answered with additional computation. For example, the equation of the plane curve will contain information about the irreducible components of the surface intersection.

7 References

1. Abhyankar, S., (1983), Desingularization of Plane Curves, *Proc. of Symposia in Pure Mathematics*, AMS, 40, 1, 1-45.

2. Abhyankar, S., and Bajaj, C., (1987), Automatic Rational Parameterization of Curves and Surfaces III: Algebraic Plane Curves, *Comp. Aided Geometric Design*, to appear.

3. Abhyankar, S., and Bajaj, C., (1987), Automatic Rational Parameterization of Curves and Surfaces IV: Algebraic Space Curves, Tech. Rept. 703, Comp. Science, Purdue University.

4. Paul R. Arner (1987), *Another Look at Surface/Surface Intersection*, Ph.D. Thesis, Dept. of Math., University of Utah.

5. C. Bajaj, C. Hoffmann, J. Hopcroft, R. Lynch (1987), "Tracing Surface Intersections," Tech. Rept., Comp. Sci. Dept., Purdue University. To apppear in *Comp. Aided Geometric Design.*

6. R. Barnhill, G. Farin, M. Jordan, B. Piper (1987), "Surface/surface intersection," *Comp. Aided Geometric Design 4*, 3–16.

7. Buchberger, B., (1985), "Gröbner Bases: An Algorithmic Method in Polynomial Ideal Theory," in *Multidimensional Systems Theory*, N. K. Bose, ed., D. Reidel Publishing Co., 184–232.

8. Buchbeger, B., (1987), "Applications of Gröbner Bases," Summer Program in Robotics, Instit. for Math. and Applic., Univ. of Minnesota.

9. Buchberger, B., Collins, G., and Kutzler, B., (1988), "Algebraic Methods for Geometric Reasoning," *Annl. Reviews in Comp. Sci.*, Vol. 3.

10. Cohen, E., Lyche, T., and Riesenfeld, R., (1980), Discrete B-splines and Subdivision Techniques in Computer Aided Geometric Design and Computer Graphics, *Computer Graphics and Image Processing*, 14, 87 - 111.

11. Dongarra, J., Moler, C., Bunch, J., and Stewart, G., (1979), *Linpack User's Guide*, SIAM Publications, Philadelphia

12. Farouki, R., (1986), Trimmed Surface Algorithms for the Evaluation and Interrogation of Solid Boundary Representations, *IBM J. Res. and Dev.*, 31.

13. Franklin, P., (1944), *Methods of Advanced Calculus*, McGraw-Hill

14. Garrity, T., and Warren, J., (1987) On Computing the Intersection of a Pair of Algebraic Surfaces, *Manuscript.*

15. A. Geisow (1983), *Surface Interrogations*, Ph.D. Dissertation, University of East Anglia, School of Computing Studies and Accountancy.

16. Hoffmann, C., (1987), Algebraic Curves, in *Mathematical Aspects of Scientific Software*, J. Rice, ed., IMA Volumes in Math. and Applic., Springer Verlag, 101–122.

17. Levin, J., (1979), Mathematical Models for Determining the Intersections of Quadric Surfaces, *Computer Graphics and Image Processing*, 11, 73 - 87.

18. Montaudouin, Y., (1987) "Criterion for Terminating Subdivision in the Surface/Surface Intersection Problem. The X Algorithm," Manuscript; Center for Appl. of Math., Lehigh University.

19. Nasri, A., (1984) *Polyhedral Subdivision Methods for Free-Form Surfaces*, Ph.D. Diss., Comp. Studies and Accountancy, Univ. of East Anglia.

20. Owen, J., and Rockwood, A., (1987) Intersection of General Implicit Surfaces, in *Geometric Modeling: Algorithms and Trends*, G. Farin, ed., SIAM Publications, Philadelphia

21. Palmer, T., (1982) Sculptured Surfaces in Volume Modeling Systems, Ph.D. Diss., Comp. Studies and Accountancy, Univ. of East Anglia.

22. Pratt, M., (1986), Parametric Curves and Surfaces as used in Computer Aided Design, *The Mathematics of Surfaces*, ed. J. Gregory, Oxford University Press, 117–142.

23. Requicha, A., and Voelcker, H., (1983), Solid Modeling: Current Status and Research Directions, *IEEE Comp. Graphics and Applic.*, 3, 25 - 37.

24. Sederberg, T., (1983), Implicit and Parametric Curves and Surfaces for Computer Aided Geometric Design, Ph.D. Thesis, Mech. Engr., Purdue University.

25. Sederberg, T., and Parry, S., (1986), A comparison of Curve Intersection Algorithms, Comp. Aided Des., 18, 58–63.

26. Snyder, V., and Sisam, C., (1914), *Analytic Geometry of Space*, Henry Holt and Company, New York; Art. 176, p. 219.