Solving Geometric Constraints using Multivariate Rational Spline Functions

Gershon Elber
Department of Computer Science
Technion, Israel Institute of Technology
Haifa 32000, Israel
E-mail: gershon@cs.technion.ac.il

Myung-Soo Kim
School of Computer Science and Engineering
Seoul National University
Seoul 151-742, Korea
E-mail: mskim@cse.snu.ac.kr

Abstract

We present a new approach to building a solver for a set of geometric constraints represented by multivariate rational functions. The constraints are formulated using inequalities as well as equalities. When the solution set has dimension larger than zero, we approximate it by fitting a hypersurface to discrete solution points. We also consider a variety of constraint solving problems common in geometric modeling. These include computing ray-traps, bisectors, sweep envelopes, and regions accessible during 5-axis machining.

1 Introduction

Computing the solution of a set of non-linear equations has been considered a very difficult task. Even for the univariate case, the root-finding procedure can be unstable and prone to errors. Lane and Riesenfeld [Lane81] propose a subdivision-based approach, using the Bernstein-Bézier basis function, for the computation of roots of univariate polynomial functions. The numerical stability of the Bernstein-Bézier basis functions [Farouki88] makes this approach attractive and robust. Following this lead, many efficient methods have been developed for a large variety of root-finding problems in geometric modeling, including ray-surface, curve-curve, and surface-surface intersections. The same approach has also been applied to the more general B-spline representation.

Nishita et al. [Nish90] introduced a *Bézier clipping* technique that can compute the roots of univariate Bézier functions very efficiently. Looking at higher dimensions, Sherbrook and Patrikalakis [Sher93] presented an approach to solving a set of multivariate polynomial equations given in the Bernstein-Bézier form. When the problem has many roots, a large number of subdivision steps are required at an early stage of the algorithm; thus, the cost of each subdivision is an important factor in the overall performance. At each subdivision, Sherbrook and Patrikalakis can reduce the domain to a much smaller subdomain; however, the cost of this domain clipping is high and there is a trade-off against the reduction of domain size.

Although starting with many subdivisions is inevitable, we are able to eliminate redundant subdivisions at the final stage if we can guarantee that there is at most one root in each subdomain. Domain clipping has no explicit mechanism to guarantee such a condition. Sederberg and his colleagues' concepts [Sede88, Sede96] of the normal cone and the surface bounding cone are more suitable to this purpose. In this paper, we generalize these tools to higher dimensions for the intersection of multivariate implicit hypersurfaces. Note that normal cones and surface bounding cones are easier to compute for implicit surfaces than for parametric surfaces; this fact greatly simplifies our algorithm. When we have isolated all the different roots using the surface bounding cones, we terminate the subdivision procedure and switch to Newton-Raphson iterations which approximate each separate root with quadratic convergence.

The test for this termination condition usually fails at the initial stage of the algorithm. So, to start with we subdivide the domain without checking the rather costly termination condition. Moreover, we speed up this procedure by employing a trivial subdivison scheme that checks only the signs of control coefficients of each multivariate function over a domain. When the signs of control coefficients of a multivariate function are all positive (or all negative), by the convex hull property the function is also positive (or negative) throughout the domain. Consequently, we can eliminate such a domain since it contains no common root for a given set of multivariate equations. Many subdomains are thus eliminated early since many of them have at least one function strictly greater than, or strictly less than, zero in the subdomain. This further justifies our use of a simple subdivision scheme.

When the number of equality constraints is smaller than the number of variables, the solution set has dimension larger than zero. We approximate the solution set by fitting a hypersurface to some discrete points sampled from the solution set. These discrete solution points are also generated by recursive subdivision followed by Newton-Raphson iteration with quadratic convergence. Although we cannot separate roots (since they are continuous in this case), the surface bounding cones are still useful to guarantee that the hypersurfaces intersect each other transversally in the domain and thus to ensure that each domain contains a connected component of the solution set. A simple subdivision scheme may also be used efficiently to generate a dense distribution of discrete points across the solution set.

In this paper, we also consider a variety of constraint-solving problems common in geometric modeling. These include computing ray-traps, bisectors, sweep envelopes, and accessible regions in 5-axis machining. (See also the authors' other publications [Heo99, Kim00], which apply a similar approach to computing the intersection of two ruled surfaces, the intersection of two ringed surfaces, and the Minkowski sum of two surfaces.) We show how to convert each of these problems into the search for a solution to a set of multivariate rational equations. To simplify the solution procedure, we eliminate some variables such as x, y, z from the given equations and solve equations in the remaining input parameters. Since the geometric problem is originally given in Euclidean space, the solution set must be computed in this space, and not in the space of other input parameters. When the solution set has dimension larger than zero, we generate some discrete solution points in Euclidean space and fit a curve or surface through these points. Our implementation as well as all the examples shown in this paper are based on the IRIT solid modeling system [Irit00] that has been developed at the Technion, Israel Institute of Technology.

This paper is organized as follows. In Section 2, we present the zero-set finding procedure, including the subdivision as well as the numerical improvement stage. In Section 3, we present some examples, and we conclude the paper in Section 4.

2 Zero-Set Finding for Multivariate Functions

We consider the problem of zero-set finding for a set of multivariate functions. One can subdivide the multivariate function(s) until the function values are bounded within a certain resolution. Section 2.1 discusses this subdivision stage, which ends up with a set of discrete points approximating the zero-set. We then apply a numerical improvement to this set of points, which is explained in Section 2.2. If the dimension of the solution set is greater than zero, then we can fit a multivariate surface to the set of discrete solution points, which produces an approximation to the solution set. This fitting stage is discussed in Section 2.3. Finally, in Section 2.4 we consider the case where the zero-set has dimension zero, thus forming a set of discrete points. We investigate the conditions for which a subdomain of the problem contains a single solution to the multivariate problem.

2.1 Subdivision for Zero-Set Finding

Consider n multivariate (piecewise) rational constraints, $\mathcal{F}_i(u_1, u_2, \cdots, u_{m-1}) = 0$, $(i = 1, \cdots, n)$, in $\boldsymbol{R}^{m-1}$, where $n \leq m-1$. And let a point in the $(m-1)$-dimensional parameter space of $\mathcal{F}_i$ be denoted by $\boldsymbol{u} = (u_1, u_2, \cdots, u_{m-1})$. Now let us consider all simultaneous solution points, $\boldsymbol{u}^s \in \boldsymbol{R}^{m-1}$, such that $\mathcal{F}_i(\boldsymbol{u}^s) = 0$, for all $i = 1, \cdots, n$.

We will assume that $\mathcal{F}_i$, $i = 1, \cdots, n$, are represented as B-spline or Bézier multivariate scalar surfaces. Then, the domain of $\mathcal{F}_i(\boldsymbol{u})$ contains zeros only if the control coefficients of $\mathcal{F}_i$ have different signs; this is because the value of zero must be contained in the convex hull of the control coefficients of $\mathcal{F}_i$. Using this convex hull property, a subdivision-based approach, as described in Algorithm 2.1, will generate a set of discrete points that robustly converges to the simultaneous zero-set of $\mathcal{F}_i$, $i = 1, \cdots, n$.

Let τ be a subdivision tolerance that determines the maximum number of points to be generated in the subdivision stage. We may assume $0 \leq u_i \leq 1$, $i = 1, \cdots, n$; then the parameter domain is a unit hypercube of dimension $m-1$, which is sliced into $\approx \frac{1}{\tau}$ segments in each coordinate direction. The result of this subdivision will be a total of $\left(\frac{1}{\tau}\right)^{m-1}$ cells. Each such cell may generate one point (i.e. the center of each cell); thus, we will end up with at most $\left(\frac{1}{\tau}\right)^{m-1}$ approximate solution points to the zero-set at the end of this subdivision stage. This upper bound is realized only when the entire space is contained in the zero-set, which is unlikely to happen in typical cases.

As we mentioned above, this approach can be easily extended to support constraints represented by inequalities as well as equalities. In Algorithm 2.1, we can handle inequalities properly by slightly modifying line $(*)$ as follows:

if ($\exists i$ such that $\left(p_i^{min} p_i^{max} > 0\right.$ and $\mathcal{F}_i$ is equality constraint$\left.\right)$ or
  $\exists i$ such that $\left(p_i^{min} > 0\right.$ and $\mathcal{F}_i$ is negative constraint$\left.\right)$ or
  $\exists i$ such that $(p_i^{max} < 0$ and $\mathcal{F}_i$ is positive constraint$)$) **then**

2.2 Numerical Improvement for Zero-Set Finding

This section discusses the numerical improvement of the approximated solutions. While this is essentially a Newton-Raphson process, we take a more geometrically intuitive approach. Moreover, only equality constraints are considered in this section, under the assumption that the (strict) inequality constraints may always be satisfied by taking a sufficient number of subdivisions if necessary.

Let $\boldsymbol{u}^0 = (u_1^0, u_2^0, \cdots, u_{m-1}^0)$ be an approximate solution as computed in Section 2.1. That is, $\mathcal{F}_i(\boldsymbol{u}^0) \approx 0$, for $i = 1, \cdots, n$. The normal space of the multivariate graph surface $(\boldsymbol{u}, \mathcal{F}_i(\boldsymbol{u}))$ is a line in $\boldsymbol{R}^m$, and at $(\boldsymbol{u}^0, \mathcal{F}_i(\boldsymbol{u}^0))$ this line is parallel to

$$\boldsymbol{n}_i(\boldsymbol{u}^0) = \left(\frac{\partial \mathcal{F}_i}{\partial u_1}(\boldsymbol{u}^0), \frac{\partial \mathcal{F}_i}{\partial u_2}(\boldsymbol{u}^0), \cdots, \frac{\partial \mathcal{F}_i}{\partial u_{m-1}}(\boldsymbol{u}^0), -1 \right), \quad (1)$$

where $\frac{\partial \mathcal{F}_i}{\partial u_j}(\boldsymbol{u}^0)$ denotes the partial derivative of each $\mathcal{F}_i$ with respect to u_j, evaluated at $\boldsymbol{u}^0$. Complementing the normal's one-dimensional space at $(\boldsymbol{u}^0, \mathcal{F}_i(\boldsymbol{u}^0))$ yields the tangent to $(u_1, u_2, \cdots, u_{m-1}, \mathcal{F}_i(\boldsymbol{u}))$, an m-variate hyperplane with parameters $\overline{\boldsymbol{u}} = (u_1, u_2, \cdots, u_m)$, that is equal to

$$\begin{aligned} \mathcal{P}_i^{\boldsymbol{u}^0}(\overline{\boldsymbol{u}}) \; : \quad & \frac{\partial \mathcal{F}_i}{\partial u_1}(\boldsymbol{u}^0)u_1 + \frac{\partial \mathcal{F}_i}{\partial u_2}(\boldsymbol{u}^0)u_2 + \cdots \\ & + \frac{\partial \mathcal{F}_i}{\partial u_{m-1}}(\boldsymbol{u}^0)u_{m-1} - u_m - \left\langle \boldsymbol{n}_i(\boldsymbol{u}^0), \overline{\boldsymbol{u}}^0 \right\rangle \\ = \quad & \left\langle \boldsymbol{n}_i(\boldsymbol{u}^0), \overline{\boldsymbol{u}} \right\rangle - \left\langle \boldsymbol{n}_i(\boldsymbol{u}^0), \overline{\boldsymbol{u}}^0 \right\rangle = 0, \end{aligned}$$

where $\overline{\boldsymbol{u}}^0 = (u_1^0, u_2^0, \cdots, u_{m-1}^0, u_m^0) = (u_1^0, u_2^0, \cdots, u_{m-1}^0, \mathcal{F}_i(\boldsymbol{u}^0))$.

Using a first-order approximation, each function $\mathcal{F}_i$ constrains the solution to be on the hyperplane $\mathcal{P}_i^{\boldsymbol{u}^0}(\overline{\boldsymbol{u}})$. The planes $\mathcal{P}_i^{\boldsymbol{u}^0}(\overline{\boldsymbol{u}})$ also contain u_m as a free variable and we recall that $\mathcal{F}_i(\boldsymbol{u}) = 0$, $\forall i$. So this requirement can be simply embedded as the additional constraint $u_m = 0$. The final results are $n+1$ linear constraints.

Consider the case that $m = n+1$, which leads to a fully constrained system of m linear equations formulated from the n tangent planes $\mathcal{P}_i^{\boldsymbol{u}^0}(\overline{\boldsymbol{u}})$ of the functions $\mathcal{F}_i(\boldsymbol{u})$, and also one additional constraint formulated as $u_m = 0$, the zero being sought. Therefore, we have m linear equations and m unknowns. Note that m planes, in general position in $\boldsymbol{R}^m$, intersect at a single point, which corresponds to the result of one iteration of a multidimensional Newton-Raphson procedure in $\boldsymbol{R}^m$, with a quadratic convergence rate near simple roots [Luen84].

For example, consider the case of two scalar bivariate constraints in $\boldsymbol{R}^3$, written as $\mathcal{F}_i(x, y) = 0$, $i = 1, 2$; additionally, let $z = 0$ and $\mathcal{P}_i(x, y, z) = 0$, $i = 1, 2$; the whole amounting to three linear constraints and three unknowns. Three planes in general position

```
Algorithm 2.1
Input:
    $\mathcal{F}_i$, $i = 1, \cdots, n$, $n$ multivariate rational constraints;
    $\tau$, tolerance of subdivision process;
Output:
    $\mathcal{S}$, a set of points in the parametric space of $\mathcal{F}_i$ such that each point $\boldsymbol{u}^s \in \mathcal{S}$ yields
      an approximated simultaneous zero over all $\mathcal{F}_i$;
ZeroSetSubdiv( $\{\mathcal{F}_i\}_{i=1}^n$, $\left\{(u_j^{min}, u_j^{max})\right\}_{j=1}^{(m-1)}$, $\tau$ )
begin
    $\Delta^{max} u \Leftarrow \max_j \left(u_j^{max} - u_j^{min}\right)$;
    if ($\Delta^{max} u < \tau$) then begin
        return $\left\{\left(\frac{u_1^{min}+u_1^{max}}{2}, \frac{u_2^{min}+u_2^{max}}{2}, \cdots, \frac{u_{m-1}^{min}+u_{m-1}^{max}}{2}\right)\right\}$;
    end
    else begin
        $p_i^{min}$, $p_i^{max}$ $\Leftarrow$ minimal and maximal control coefficients of $\mathcal{F}_i$;
(*)     if ($\exists i$ such that $p_i^{min} p_i^{max} > 0$) then
            return $\phi$;
        else begin
            $\mathcal{F}_i^1$, $\mathcal{F}_i^2 \Leftarrow \mathcal{F}_i$ subdivided at $\frac{u_j^{min}+u_j^{max}}{2}$, along the $j$th parametric direction,
                  where $\Delta^{max} u$ has been detected, $i = 1, \cdots, n$;
            return ZeroSetSubdiv$\left(\left\{\mathcal{F}_i^1\right\}_{i=1}^n\right.$,
                        $\left.\left\{(u_0^{min}, u_0^{max}), \cdots \left(u_j^{min}, \frac{u_j^{min}+u_j^{max}}{2}\right), \cdots (u_n^{min}, u_n^{max}), \right\}, \tau\right)$
               $\cup$ ZeroSetSubdiv$\left(\left\{\mathcal{F}_i^2\right\}_{i=1}^n\right.$,
                        $\left.\left\{(u_0^{min}, u_0^{max}), \cdots \left(\frac{u_j^{min}+u_j^{max}}{2}, u_j^{max}\right), \cdots (u_n^{min}, u_n^{max}), \right\}, \tau\right)$;
        end
    end
```

in $\boldsymbol{R}^3$ intersect at a single point, which is also the result of one iteration of the Newton-Raphson procedure.

Now let $m > n + 1$, and the linear system of equations will be under-determined. Note that $(n + 1)$ planes intersect in an $(m - n - 1)$-dimensional variety $\mathcal{V}$. At this juncture, we may need to find the solution in $\mathcal{V}$ that minimizes a certain additional optimality criterion. For this purpose, we employ a linear system solver that yields a least-squares solution in an L_2-norm, in order to find the closest solution to the current position $\boldsymbol{u}^0$ from the possible solution set $\mathcal{V}$. Singular-value decomposition and QR factorization [Golu96] are two options we can take for solving this under-determined problem while producing a solution with a minimal L_2-norm, and thus minimizing the displacement from $\boldsymbol{u}^0$.

A QR factorization procedure was used in preparing the examples in this paper. The tolerance employed in the numerical improvement stage is several magnitudes smaller than that of the subdivision stage. In the case of a solution space of dimension greater than zero, the vast majority of points were successfully improved to that tolerance in very few iterations, becoming more than an order of magnitude closer to the solution at each iteration. Thus, we are not required to improve all points from the subdivision stage; those few points that fail to improve can simply be purged away. Examples are presented in Section 3.

2.3 Fitting a HyperSurface to the Solution Set

Once a set of discrete points that satisfy the simultaneous zeros has been constructed (and assuming that the dimension of the solution space is greater than zero), we fit a smooth hypersurface to it, thus approximating the solution set.

In contrast to the problem of regular scattered data interpolation, this problem permits a simple solution to the parameterization of data points. In typical cases, we can easily parameterize the zero-set using the parameterization of one of the given input constraints. This approach has been taken in some examples presented in Section 3.

2.4 Conditions for the Uniqueness of a Solution

One solution point is generated (or sampled) for each subdomain we obtain at the final resolution of the subdivision procedure. If a subdomain contains more than one connected component of the solution set, generating only one sample point may not be sufficient to construct a good continuous approximation. Conversely, if a subdomain at some intermediate resolution contains only one connected component, we may need to stop the subdivision process and switch to a more efficient (numerical) procedure for generating one or more solution points.

An efficient method for checking whether there is more than one connected component in a subdomain is also important when the solution space has dimension zero (thus forming a set of discrete points). When it is known that only one solution exists in a subdomain, we can stop the subdivision and change to a numerical procedure. In many cases, we can stop the subdivision at a higher level (and thus at a lower resolution). Moreover, this capability can be used in isolating different solution points no matter how close together they are. That is, we can repeat the subdivision process until each region contains at most one solution point.

Sederberg and Meyers [Sede88] use the hodograph as the basis of such a termination condition in the intersection of two planar Bézier curves. Two sub-curves intersect at most once if their hodographs share no common direction vector. Sederberg and his colleagues [Sede88, Sede96] also develop a similar condition for the intersection of two parametric surfaces, where the surface bounding (or tangent) cone plays an important role. In this paper, we generalize this approach to the higher-dimensional problem of intersecting $m-1$ implicit hypersurfaces $\mathcal{F}_i(\boldsymbol{u}) = 0$, for $i = 1, \cdots, m-1$, in $\boldsymbol{R}^{m-1}$. The normal vector of an implicit hypersurface is considerably easier to compute than that of a parametric hypersurface. This fact greatly simplifies our algorithm for computing the normal and surface bounding cones.

Let $\mathcal{C}(\boldsymbol{v}, \alpha)$ denote the cone with the axis in the direction of a non-zero vector $\boldsymbol{v}$ and an opening angle α:

$$\mathcal{C}(\boldsymbol{v}, \alpha) = \{\boldsymbol{u} \mid \langle \boldsymbol{u}, \boldsymbol{v}\rangle^2 \le \|\boldsymbol{u}\|^2 \|\boldsymbol{v}\|^2 \cos^2 \alpha\}.$$

The normal space of an implicit hypersurface has dimension one. For an implicit hypersurface $\mathcal{F}_i(\boldsymbol{u}) = 0$, we can define its normal cone $\mathcal{C}_i^n = \mathcal{C}(\boldsymbol{v}_i^n, \alpha_i^n) \subset \boldsymbol{R}^{m-1}$ as the set of all possible normal vectors $\nabla \mathcal{F}_i(\boldsymbol{u})$ and their scalar multiples. Clearly, we have

$$\begin{aligned}
&\nabla \mathcal{F}_i(\boldsymbol{u}) \\
&= \left(\frac{\partial \mathcal{F}_i}{\partial u_1}(\boldsymbol{u}), \frac{\partial \mathcal{F}_i}{\partial u_2}(\boldsymbol{u}), \cdots, \frac{\partial \mathcal{F}_i}{\partial u_{m-1}}(\boldsymbol{u})\right) \\
&= \sum_{i_1}\sum_{i_2}\cdots\sum_{i_{m-1}} \Bigg(P^{u_1}_{i_1,i_2,\cdots,i_{m-1}}, P^{u_2}_{i_1,i_2,\cdots,i_{m-1}}, \cdots, \\
&\qquad\qquad P^{u_{m-1}}_{i_1,i_2,\cdots,i_{m-1}}\Bigg) \\
&\qquad B_{i_1,k_{i_1}}(u_1) B_{i_2,k_{i_2}}(u_2) \cdots B_{i_{m-1},k_{i_{m-1}}}(u_{m-1}), \quad (2)
\end{aligned}$$

where the terms $P^{u_j}_{i_1,i_2,\cdots,i_{m-1}}$ denote the coefficients of the partial derivative of $\mathcal{F}_i$ with respect to u_j, elevated to a common degree. Then, $\mathcal{C}_i^n = \mathcal{C}(\boldsymbol{v}_i^n, \alpha_i^n)$ can be derived by letting $\boldsymbol{v}_i^n$ be the average of the vectors $\left(P^{u_1}_{i_1,i_2,\cdots,i_{m-1}}, P^{u_2}_{i_1,i_2,\cdots,i_{m-1}}, \cdots, P^{u_{m-1}}_{i_1,i_2,\cdots,i_{m-1}}\right)$, $\forall i_1, i_2, \cdots, i_{m-1}$, and α_i^n be the maximum angle between $\boldsymbol{v}_i^n$ and these vectors.

Given $\mathcal{C}_i^n$, we can define its complementary cone $\mathcal{C}_i^c$, which contains all vectors that are orthogonal to vectors in $\mathcal{C}_i^n$:

$$\mathcal{C}_i^c = \left\{\boldsymbol{w} \in \boldsymbol{R}^{m-1} \mid \exists \boldsymbol{u} \in \mathcal{C}_i^n \text{ such that } \langle \boldsymbol{u}, \boldsymbol{w}\rangle = 0\right\}. \quad (3)$$

$\mathcal{C}_i^c$ is also called the tangent cone [Sede88] and it contains all possible tangent directions of the implicit hypersurface $\mathcal{F}_i(\boldsymbol{u}) = 0$. $\mathcal{C}_i^c$ is of dimension $m-1$ and can easily be derived from $\mathcal{C}_i^n$ as follows (see Figure 1):

$$\begin{aligned}
\mathcal{C}_i^c &= \mathcal{C}(\boldsymbol{v}_i^c, \alpha_i^c) \\
&= \mathcal{C}(\boldsymbol{v}_i^n, 90 - \alpha_i^n). \quad (4)
\end{aligned}$$

In other words, $\mathcal{C}_i^c$ and $\mathcal{C}_i^n$ share the same axis, but have complementary angles.

Finally, let $\mathcal{C}_i^c(\boldsymbol{u}^0) \subset \boldsymbol{R}^{m-1}$ denote the translation of $\mathcal{C}_i^c$ by $\boldsymbol{u}^0$. Then we have $\{\boldsymbol{u} \mid \mathcal{F}_i(\boldsymbol{u}) = 0\} \subset \mathcal{C}_i^c(\boldsymbol{u}^0)$, $\forall \boldsymbol{u}^0$ such that $\mathcal{F}_i(\boldsymbol{u}^0) = 0$. (Based on the mean value theorem, we can provide a rigorous proof that is similar to Sederberg and Meyers' [Sede88]; but we omit the details here.)

We are now ready to state the main result of this section:

Theorem 1 *Given $m-1$ implicit hypersurfaces $\mathcal{F}_i(\boldsymbol{u}) = 0$, $i = 1, \cdots, m-1$, in $\boldsymbol{R}^{m-1}$, there is at most one common solution if*

$$\bigcap_{i=1}^{m-1} \mathcal{C}_i^c = \{\boldsymbol{0}\},$$

where $\boldsymbol{0}$ is the origin of the coordinate system.

Proof: Assume that $\boldsymbol{u}^0 \in \boldsymbol{R}^{m-1}$ is a common solution of the $m-1$ equations $\mathcal{F}_i(\boldsymbol{u}) = 0$, $i = 1, \cdots, m-1$. Consider $\mathcal{C}_i^c(\boldsymbol{u}^0)$. From the relation $\{\boldsymbol{u} \mid \mathcal{F}_i(\boldsymbol{u}) = 0\} \subset \mathcal{C}_i^c(\boldsymbol{u}^0)$, we have

$$\bigcap_{i=1}^{m-1} \{\boldsymbol{u} \mid \mathcal{F}_i(\boldsymbol{u}) = 0\} \subset \bigcap_{i=1}^{m-1} \mathcal{C}_i^c(\boldsymbol{u}^0) = \{\boldsymbol{u}^0\}.$$

Thus, there is no other common solution except $\boldsymbol{u}^0$. ■

The fact that one can detect subdomains with at most one root is of great importance. Under this condition, Algorithm 2.1 can be terminated early, and we can significantly improve the performance of our solver.

3 Examples of Geometric Constraint Solving

This section considers a variety of constraint-solving problems that are common in geometric modeling and computation. These problems can greatly benefit from the multivariate rational solver introduced in the previous section. In particular, we present some interesting results from classical geometry (Sections 3.1 and 3.2), from modern computational geometry (Section 3.3), from geometric modeling (Section 3.4), and from accessibility analysis used to support multi-axis NC machining (Section 3.5).

3.1 Ray-Traps or Bouncing Billiard-Balls

Consider n curve segments $C_i(u)$, $i = 0, \cdots, n-1$, in the xy-plane.

Definition 1 *Given a set of n planar curves, its* ray-trap *of length n is a set of n points $\{P_i = C_i(u_i)\}$ such that a ray bouncing from P_i toward $P_{(i+1) \bmod n}$ will be reflected toward $P_{(i+2) \bmod n}$.*

We consider how to compute all ray-traps of length n for a given configuration of n freeform curves. In the context of piecewise linear curves, this problem is quite well-known under various different names. In particular, Tabachnikov [Taba95] discussed the entire topic of *bouncing billiard-balls*, which includes the case of (the inside of) polygonal objects. For example, every triangle has one such ray-trap path of length three (a billiard ball bouncing off every edge once). This path is the pedal triangle that is formed out of the feet of the three altitudes, a fact also pointed out by Wells [Well91].

We now consider the problem of ray-traps in the world of freeform rational curves. The incoming ray from P_{i-1} and the outgoing ray toward P_{i+1} should form an identical angle with the normal of C_i at P_i. This angular equality can be written as

$$\frac{\langle P_{i-1} - P_i, N_i\rangle}{\|P_{i-1} - P_i\|} = \frac{\langle P_{i+1} - P_i, N_i\rangle}{\|P_{i+1} - P_i\|}, \quad (5)$$

or, in terms of the original curves, as follows:

$$\begin{aligned}
&\frac{\langle C_{i-1}(u_{i-1}) - C_i(u_i), N_i(u_i)\rangle}{\|C_{i-1}(u_{i-1}) - C_i(u_i)\|} \\
&= \frac{\langle C_{i+1}(u_{i+1}) - C_i(u_i), N_i(u_i)\rangle}{\|C_{i+1}(u_{i+1}) - C_i(u_i)\|}. \quad (6)
\end{aligned}$$

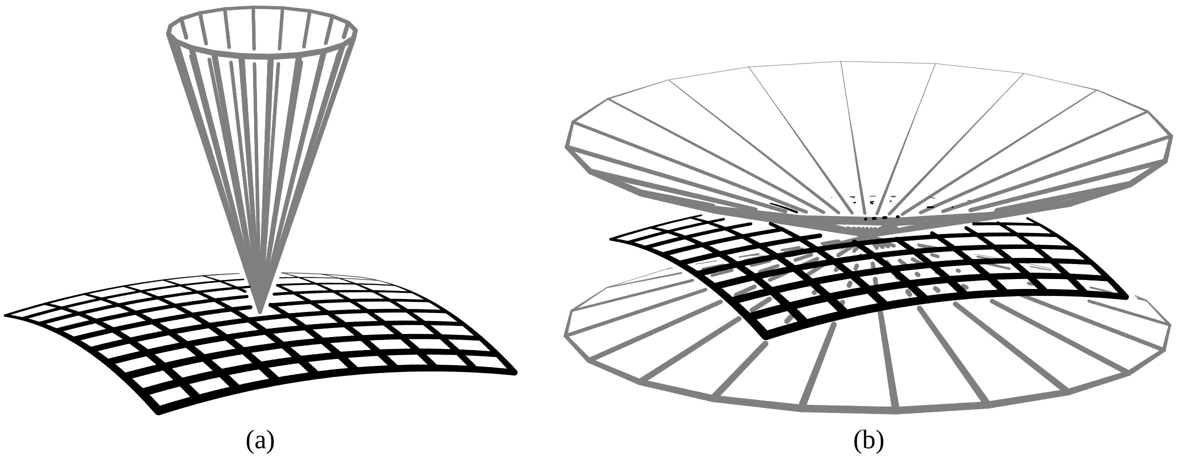

Figure 1: (a) The normal cone, $\mathcal{C}_i^n$ (in gray) and (b) the complementary tangent cone, $\mathcal{C}_i^c$ (in gray) of a freeform surface.

Equation (6) is non-rational; however, its square is rational. Now, let $C_i(u_i) = (x_i(u_i),\ y_i(u_i))$. Because the normal field $N_i(u_i)$ appears in both sides of Equation (6), it need not be normalized, and may be formulated as:

$$N_i(u_i) = (-y_i(u_i),\ x_i(u_i)).$$

The n curves have n degrees of freedom, represented by the u_i parameters. The ray-trap problem imposes n rational constraints on the n curves, each in the form of the square of Equation (6). Hence, we have n degrees of freedom and n constraints, and the solution set is finite. Nevertheless, the solution set of these squared rational constraints may contain some extraneous solutions. For example, if the incoming ray and the outgoing ray lie on the same line (but not on each other's reflected lines), they will satisfy Equation (6), since it is squared. Therefore, once a finite solution set is computed, one must examine and eliminate all extraneous solutions.

Figure 2 shows all ray-traps of length three among three circles. This solution was computed in several seconds on a modern workstation within a tolerance of 10^{-6}.

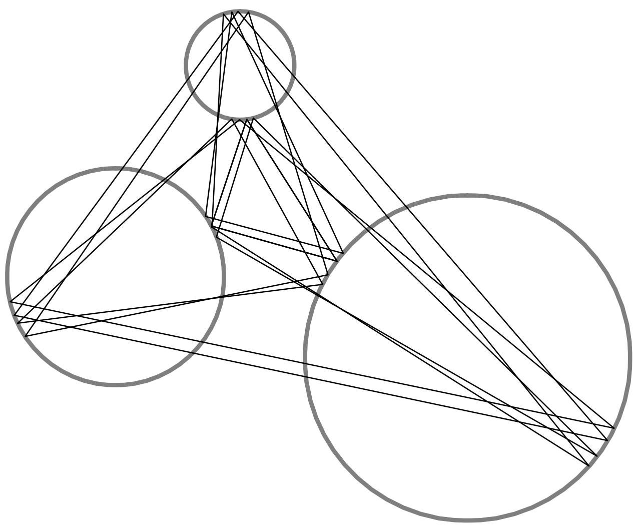

Figure 2: Ray-traps of length three of bouncing bouncing among three circles.

3.2 Singular Skeleton Points in $\boldsymbol{R}^2$

Consider three curves $C_1(u)$, $C_2(v)$, $C_3(w)$ in the xy-plane. We consider how to compute all locations P on the plane from which the distances to the three curves are the same. These locations are, for example, topologically crucial to the formation of the skeleton of the configuration discussed by de Berg et al. [deBerg98].

Point P must satisfy the following five rational constraints:

$$\langle P - C_1(u), P - C_1(u)\rangle = \langle P - C_2(v), P - C_2(v)\rangle, \quad (7)$$

$$\langle P - C_1(u), P - C_1(u)\rangle = \langle P - C_3(w), P - C_3(w)\rangle, \quad (8)$$

$$\left\langle P - C_1(u), \frac{dC_1(u)}{du}\right\rangle = 0, \quad (9)$$

$$\left\langle P - C_2(v), \frac{dC_2(v)}{dv}\right\rangle = 0, \quad (10)$$

$$\left\langle P - C_3(w), \frac{dC_3(w)}{dw}\right\rangle = 0. \quad (11)$$

Equations (7) and (8) ensure that the point P is at an equal distance from the three curve points $C_1(u)$, $C_2(v)$, $C_3(w)$. Equations (9)–(11) guarantee that the distance to the curves is indeed measured along the curves' normals. This problem is also exactly constrained, having five degrees of freedom, one parameter from each curve and the other two as the coordinates of $P = (x, y)$.

The first two constraints of distance equality (Equations (7) and (8)) can be reduced to linear equations in P as follows:

$$2\langle C_2(v) - C_1(u), P\rangle = C_2^2(v) - C_1^2(u), \quad (12)$$

$$2\langle C_3(w) - C_1(u), P\rangle = C_3^2(w) - C_1^2(u). \quad (13)$$

The point P can be obtained as a rational function of the three parameters u, v, and w, which we denote as $P = P(u, v, w) = (x(u, v, w), y(u, v, w))$.

Given three points in general position on the three curves, $C_1(u_0)$, $C_2(v_0)$, $C_3(w_0)$, P provides the point in the plane that is at an equal distance from them. That is, P is the center of the circle passing through $C_1(u_0)$, $C_2(v_0)$, and $C_3(w_0)$. Clearly, the three radii are not normals to the curves: Equations (9)–(11) have

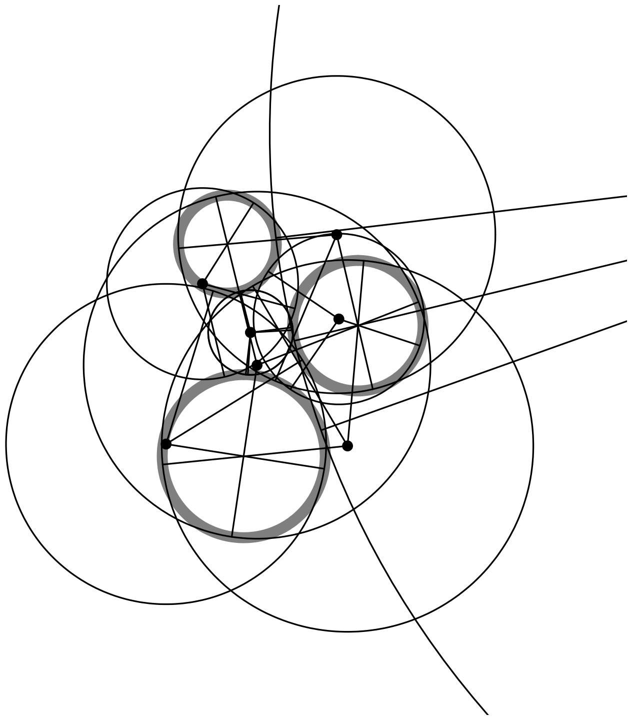

Figure 3: The classical Apollonius problem of finding the circle tangent to three given circles (in gray) is identical to the problem of finding the singular points of the planar skeleton, and the solution of the latter yields the eight circles in the solution set.

not been considered yet.

However, it is now possible to substitute $P = P(u, v, w)$, which already satisfies Equations (7) and (8), into the orthogonality constraints of Equations (9)–(11), yielding three rational (orthogonality) constraints in u, v, and w as follows:

$$\left\langle P(u,v,w) - C_1(u), \frac{dC_1(u)}{du} \right\rangle = 0, \tag{14}$$

$$\left\langle P(u,v,w) - C_2(v), \frac{dC_2(v)}{dv} \right\rangle = 0, \tag{15}$$

$$\left\langle P(u,v,w) - C_3(w), \frac{dC_3(w)}{dw} \right\rangle = 0. \tag{16}$$

We can solve the above system of three constraint equations using the rational constraints solver introduced in Section 2. The resulting solutions provide all locations in the plane that are at equal distance from the three curves.

Figure 3 shows the eight solutions corresponding to all such singular points among three circles. These results are identical to the solution of the classical Apollonius problem of finding a circle that is tangent to given three circles. All eight solutions are computed in a dozen seconds on a modern workstation within a tolerance of 10^{-6}.

3.3 Surface-Surface Bisectors

The bisector surface in $\boldsymbol{R}^3$ between two rational space curves, or between a point and a rational surface is itself rational [Elbe98b, Elbe99]. Nevertheless, in general the bisector surface between two rational surfaces in $\boldsymbol{R}^3$ is not rational.

Let $S_1(u,v)$ and $S_2(s,t)$ be two regular rational surfaces in $\boldsymbol{R}^3$. The points $\mathcal{B} = (x, y, z)$ on the bisector surface of S_1 and S_2 must satisfy the following constraints (see also [Elbe00]):

$$0 = \left\langle \mathcal{B} - S_1(u,v), \frac{\partial S_1(u,v)}{\partial u} \right\rangle, \tag{17}$$

$$0 = \left\langle \mathcal{B} - S_1(u,v), \frac{\partial S_1(u,v)}{\partial v} \right\rangle, \tag{18}$$

$$0 = \left\langle \mathcal{B} - S_2(s,t), \frac{\partial S_2(s,t)}{\partial s} \right\rangle, \tag{19}$$

$$0 = \left\langle \mathcal{B} - S_2(s,t), \frac{\partial S_2(s,t)}{\partial t} \right\rangle, \tag{20}$$

$$\begin{aligned} 0 = {} & \langle \mathcal{B} - S_1(u,v), \mathcal{B} - S_1(u,v) \rangle - \\ & \langle \mathcal{B} - S_2(s,t), \mathcal{B} - S_2(s,t) \rangle . \end{aligned} \tag{21}$$

Equations (17)–(20) mean that the distance is measured along the orthogonal lines at the bisector's foot points. Equation (21) guarantees that the distances to S_1 and S_2 are the same. The problem is formulated using seven variables u, v, s, t, x, y, z, and five constraints. Hence, the solution space forms a bivariate surface.

Let $n_1(u,v) = \frac{\partial S_1(u,v)}{\partial u} \times \frac{\partial S_1(u,v)}{\partial v}$ be the unnormalized normal field of $S_1(u,v)$. Then, if the solution to the bisector surface is of the form $\mathcal{B} = S_1(u,v) + \alpha n_1(u,v)$, ($\alpha$ is a real-valued function), Equations (17) and (18) are automatically satisfied. By substituting this proposed bisector surface point into Equation (21), we have,

$$\begin{aligned} 0 = {} & \langle \alpha n_1(u,v), \alpha n_1(u,v) \rangle \\ & - \langle S_1(u,v) + \alpha n_1(u,v) - S_2(s,t) \\ & \quad S_1(u,v) + \alpha n_1(u,v) - S_2(s,t) \rangle , \end{aligned}$$

or equivalently,

$$\alpha(u,v,s,t) = \frac{\langle S_1(u,v) - S_2(s,t), S_1(u,v) - S_2(s,t) \rangle}{-2 \langle n_1(u,v), S_1(u,v) - S_2(s,t) \rangle}. \tag{22}$$

Finally, we substitute $\alpha(u,v,s,t)$ into $\mathcal{B} = S_1(u,v) + \alpha n_1(u,v)$, and update Equations (19) and (20) with this new representation of $\mathcal{B}$. Let $\Delta(u,v,s,t)$ be defined as follows

$$\begin{aligned} & \Delta(u,v,s,t) \\ & \quad = -2(S_1(u,v) - S_2(s,t)) \langle n_1(u,v), S_1(u,v) - S_2(s,t) \rangle \\ & \qquad + n_1(u,v) \langle S_1(u,v) - S_2(s,t), S_1(u,v) - S_2(s,t) \rangle . \end{aligned}$$

Then, Equations (19)–(20) are reduced to

$$0 = \left\langle \Delta(u,v,s,t), \frac{\partial S_2(s,t)}{\partial s} \right\rangle, \tag{23}$$

$$0 = \left\langle \Delta(u,v,s,t), \frac{\partial S_2(s,t)}{\partial t} \right\rangle. \tag{24}$$

The common zero-set of Equations (23) and (24) satisfies all five constraints of Equations (17)–(21). Therefore, we have reduced the surface-surface bisector problem of $\boldsymbol{R}^3$ into the problem of finding the common zero-set of the two four-variate functions of Equations (23) and (24).

Figure 4 shows a surface that bisects a plane and a biquadratic surface. The points on this bisector surface were computed from the zero-set of Equations (23) and (24). This is an underconstrained

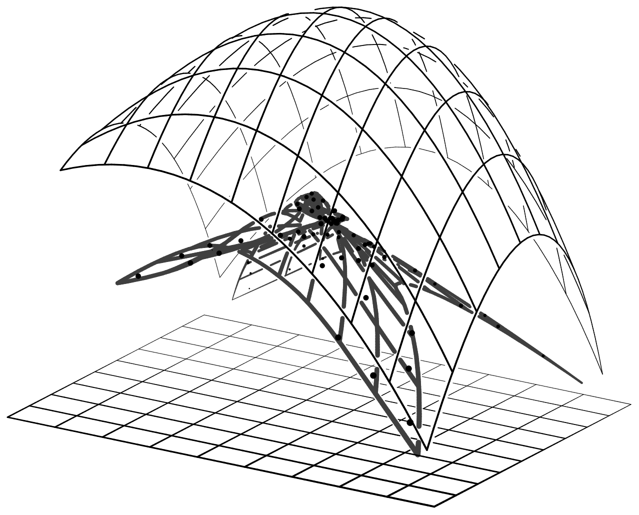
Figure 4: The bisector surface (in gray) between a plane and a quadratic surface. The points on the bisector were derived by a simultaneous solution of Equations (23) and (24).

problem, and so we can fit a bivariate surface to discrete points sampled on the bivariate solution space, which is the bisector. Either the parameterization of $S_1(u,v)$ or that of $S_2(u,v)$ can be employed to define the parameterization of the sampled points. See the authors' previous paper [Elbe00] for more details, including a discussion of curve-surface bisectors in $\boldsymbol{R}^3$.

3.4 Boundary Determination of Trivariate Solids

Consider a freeform surface $S(u,v)$ swept through an affine transformation so as to generate a trivariate function $T : \boldsymbol{R}^3 \rightarrow \boldsymbol{R}^3$, which may be written:

$$T(u,v,t) = A(t)[S(u,v)] + C(t),$$

where $A(t) = (a_{ij}(t))_{3\times 3}$ represents a linear transformation (e.g. a rotation or shearing) and $C(t) = (c_x(t), c_y(t), c_z(t))$ denotes a translation of the coordinate system.

The extraction of the boundary of the sweep is an important problem in geometric modeling and computation with many interesting applications, for example, in computing the swept volume of an NC cutter path in 3- or 5-axis machining.

Assuming $a \leq t \leq b$, the boundary surface of the swept volume consists of the surface patches of $T(u,v,a)$, $T(u,v,b)$, and the envelope surface, which is the set of points $T(u,v,t) = (T_x(u,v,t), T_y(u,v,t), T_z(u,v,t))$ that satisfies the following condition [Abdel97, Joy99, Martin90]:

$$\begin{vmatrix} \frac{\partial T_x}{\partial u} & \frac{\partial T_x}{\partial v} & \frac{\partial T_x}{\partial t} \\ \frac{\partial T_y}{\partial u} & \frac{\partial T_y}{\partial v} & \frac{\partial T_y}{\partial t} \\ \frac{\partial T_z}{\partial u} & \frac{\partial T_z}{\partial v} & \frac{\partial T_z}{\partial t} \end{vmatrix} = 0.$$

That is, the Jacobian of $T(u,v,t)$ must vanish on the envelope surface.

Having a single constraint and three degrees of freedom, the solution space has a dimension of two. Figure 5 shows the envelope surface for a scaled ellipsoid moving along a linear and a circular trajectory. Here, the solution is approximated using an approach similar to the Marching Cubes scheme [Lore87]. Both solutions in Figures 5(a) and 5(b) were computed in few seconds on a modern workstation.

3.5 Accessibility in 5-Axis Machining

Consider a surface $S(u,v)$, and let $O(u,v)$ be the specification of an orientation vector field which will allow an NC machine tool to cut the surface $S(u,v)$ in a 5-axis configuration. In other words, at $S(u_0,v_0)$ the cutter must be positioned so that its axis is parallel to $O(u_0,v_0)$. For example, if the cutter is to be aligned with the surface normal, then $O(u,v) = n(u,v)$, where $n(u,v) = \frac{\partial S}{\partial u} \times \frac{\partial S}{\partial v}$.

Let $K(s,t)$ be a check surface above $S(u,v)$. We consider how to compute the regions of $S(u,v)$ where one can machine the surface S from the direction O without gouging into the check surface K [Elbe98a]. By offsetting K by the radius of the cutter, we can transform the cutter into a line and the check surface $K(s,t)$ into an offset surface $K^o(s,t)$. Hence, we must compute all solutions (u_0,v_0) for which a ray from $S(u_0,v_0)$ in the direction $O(u_0,v_0)$ is tangent to $K^o(s,t)$.

Assume that the direction vector $O(u,v)$ is not parallel to the tangent plane of $S(u,v)$. Then, $O(u,v) \times \frac{\partial S(u,v)}{\partial u} \neq 0$ and we can define

$$\begin{aligned} O_1(u,v) &= O(u,v) \times \frac{\partial S(u,v)}{\partial u}, \\ O_2(u,v) &= O(u,v) \times O_1(u,v), \end{aligned} \tag{25}$$

as two vector fields that span the plane orthogonal to $O(u,v)$ in $\boldsymbol{R}^3$. If $O(u,v) = n(u,v)$, then we can simply choose $O_1(u,v)$ and $O_2(u,v)$ to be the partial derivatives of S, namely $\frac{\partial S}{\partial u}$ and $\frac{\partial S}{\partial v}$.

The necessary and sufficient conditions for isolating the boundary of the accessible regions of $S(u,v)$ in the direction $O(u,v)$ while tangent to $K(s,t)$ can be formulated as follows [Elbe98a]:

$$0 = \langle O_1(u,v), S(u,v) - K^o(s,t) \rangle, \tag{26}$$
$$0 = \langle O_2(u,v), S(u,v) - K^o(s,t) \rangle, \tag{27}$$
$$0 = \langle n_K(s,t), S(u,v) - K^o(s,t) \rangle, \tag{28}$$

where n_K is the normal field of surface K (and also that of K^o). Equations (26) and (27) guarantee that $S(u,v) - K(s,t)$ is in the direction of $O(u,v)$. Equation (28) means that the axis of the cutter (in the direction of $O(u,v)$) is also orthogonal to the normal field of K, namely n_K; equivalently, the axis of the cutter is indeed tangent to the offset surface K^o.

In the case of $O(u,v) = n(u,v)$, we can use the partial derivatives of $S(u,v)$ instead of $O_1(u,v)$ and $O_2(u,v)$. Thus we have

$$\begin{aligned} 0 &= \left\langle \frac{\partial S(u,v)}{\partial u}, S(u,v) - K^o(s,t) \right\rangle, \\ 0 &= \left\langle \frac{\partial S(u,v)}{\partial v}, S(u,v) - K^o(s,t) \right\rangle, \\ 0 &= \langle n_K(s,t), S(u,v) - K^o(s,t) \rangle. \end{aligned}$$

Figure 6 shows regions where Equations (26)–(28) are simultaneously satisfied in the case of a cuboid component to be accessed and a spherical check surface. There are four degrees of freedom and three rational constraints, and thus we get a univariate solution manifold. In Figure 6, these univariate curves are shown as the boundary curves of the accessible region. In a recent paper [Elbe98a], the authors presented an example similar to, but

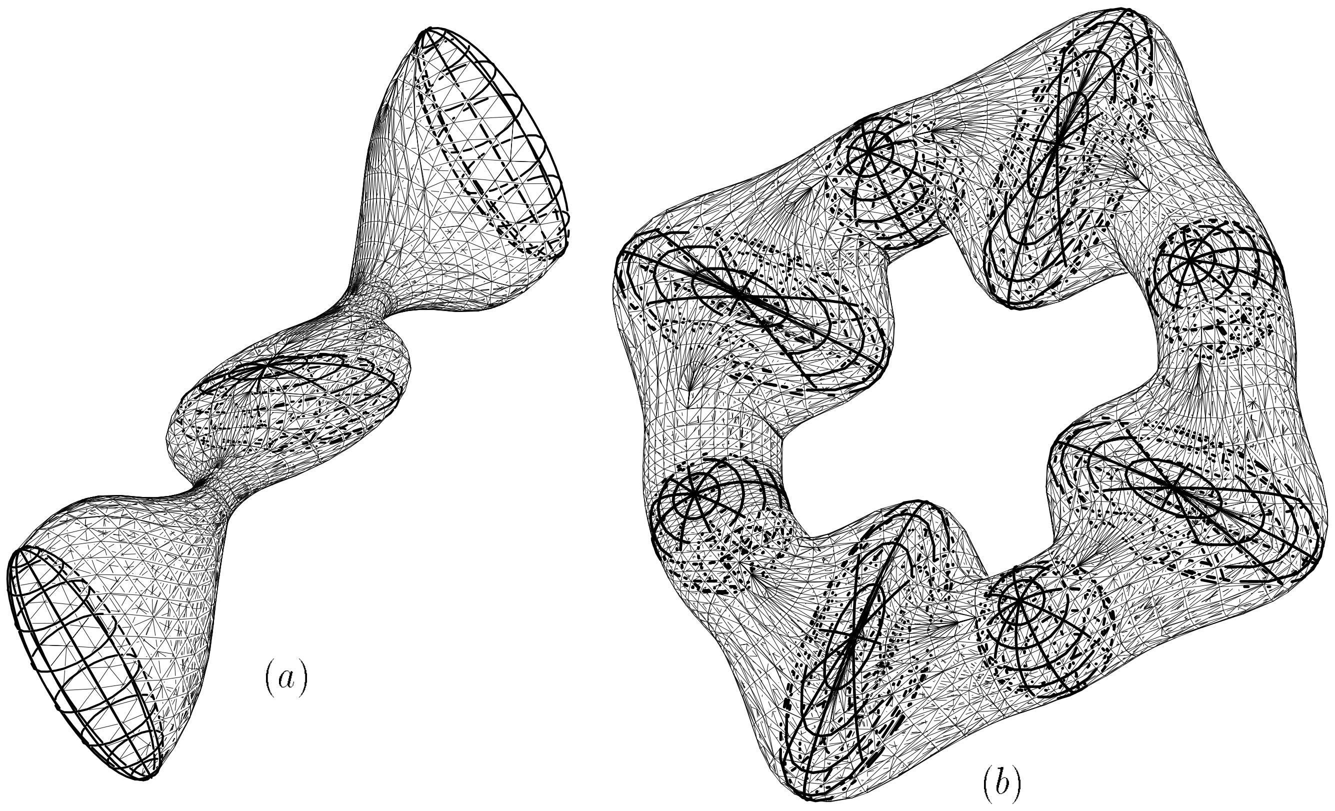

Figure 5: The envelope of a scaled ellipsoid along (a) a linear and (b) a circular trajectory.

somewhat simpler than, that of Figure 6. This previous example required a couple of hours of computation time, whereas the new example shown in Figure 6 took about a minute and a half on a modern workstation. Even if we allow a speed-up of three or four times in computation over the last few years, our new approach provides an improvement that is significantly more than an order of magnitude.

In Figure 7, additional inequality constraints are introduced. Access to the surface is limited to a hemisphere of directions centered along a prescribed vector.

4 Conclusions

In this paper, we have presented a scheme that can robustly solve a set of multivariate rational constraints. The effectiveness of this approach has also been demonstrated through practical examples and we believe that our method has considerable potential as a basic component in geometric modeling systems. Nevertheless, there is also plenty of room for further improvement. In particular, when dealing with geometric problems in higher dimensions, we need to improve the computational efficiency with which the zero-set is found.

Feasible computation of simultaneous solutions of sets of non-linear constraints has largely been restricted to lower-dimensional cases—where it has been attempted at all. Here, as the number of constraints, n, increases, the complexity of the solution rises exponentially with n. Problems with four or five constraints could be practically solved in minutes, but clearly it would be difficult to scale up the method to larger size and higher dimension. We foresee two ways of attacking this scalability problem:

- Decompose a problem of size n into smaller subproblems. Such a decomposition is feasible if one can detect independence among different constraints. An example could be found in the singular skeleton example of Section 3.2, where the first two equations were solved independently of the last three equations.
- Reduce a constraint space into a lower-dimensional unconstrained space. In some cases, we may start with certain partial solutions that satisfy $\mathcal{F}_i(\boldsymbol{u}) = 0$, for some values of i. An example of such a case can be found in the surface-surface bisector of Section 3.3. In this example, we employed an initial solution of $\mathcal{B} = S_1(u,v) + \alpha n_1(u,v)$, which automatically satisfies the first two equations, and greatly simplified the overall computation.

5 Acknowledgment

The authors are grateful to Tony Woo who introduced them the problem of ray-traps.

References

[Abdel97] K. Abdel-Malek and H.-J. Yeh. Geometric representation of the swept volume using Jacobian rank-

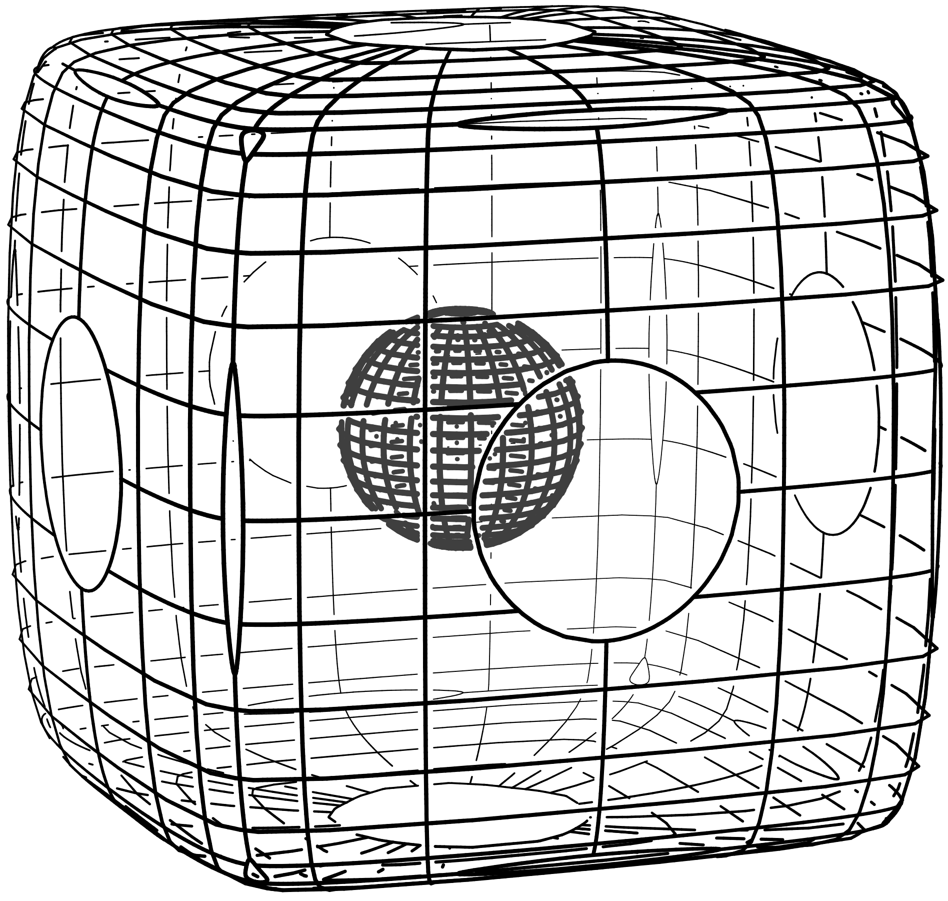

Figure 6: A cuboid surface and a spherical check surface (in gray). The cuboid surface is trimmed using the locus of points that simultaneously satisfy Equations (26)–(28). These 26 trimmed regions are the locations where the normals of the cuboid surface are tangent to the spherical check surface.

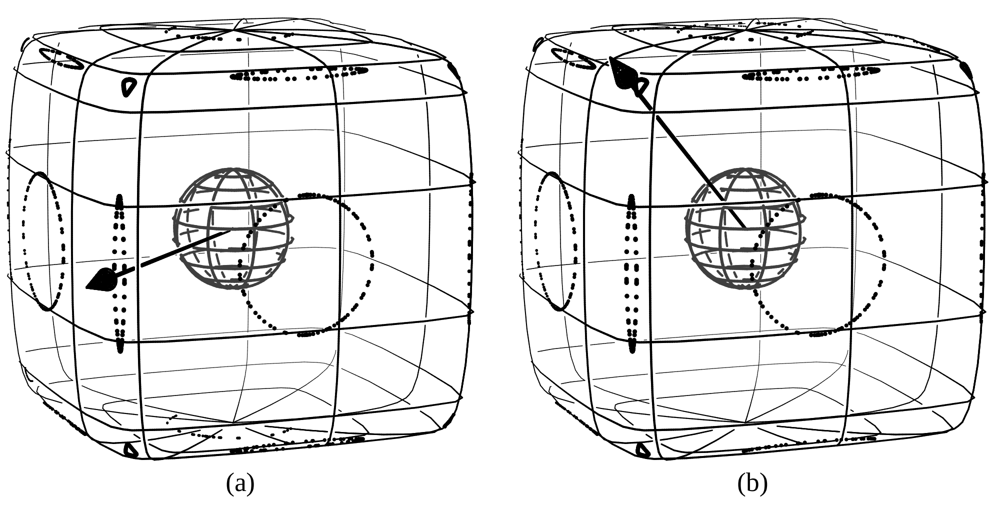

Figure 7: An inequality constraint is added to the accessibility constraints of the configuration in Figure 6. In (a), the access is limited to direction $\vec{V}$ such that $\left\langle \vec{V}, (1,1,0) \right\rangle > 0$ whereas in (b), the access is limited to direction $\vec{V}$ such that $\left\langle \vec{V}, (1,1,1) \right\rangle > 0$.

deficiency conditions. *Computer-Aided Design*, Vol 29, pp 457–468, 1997.

[deBerg98] M. de Berg, M. van Kreveld, M. Overmars and O. Schwarzkopf. *Computational Geometry, Algorithms and Applications*. Springer, New York, Second Edition, 1998.

[Elbe98a] G. Elber and E. Cohen. A unified approach to verification in 5-axis freeform milling environments. *Computer-Aided Design*, Vol 31, No 13, pp 795–804, November 1999.

[Elbe98b] G. Elber and M.-S. Kim. The bisector surface of freeform rational space curves. *ACM Transactions on Graphics*, Vol 17, No 1, pp 32–50, January 1998.

[Elbe99] G. Elber and M.-S. Kim. Computing rational bisectors. *IEEE Computer Graphics & Applications*, Vol 19, No 6, pp 76–81, November-December 1999.

[Elbe00] G. Elber and M.-S. Kim. A computational model for nonrational bisector surfaces: curve-surface and surface-surface bisectors. *Proc. of Geometric Modeling and Processing 2000*, Hong Kong, pp 364–372, April 10-12, 2000.

[Faux85] I. D. Faux and M. J. Pratt. *Computational Geometry for Design and Manufacture.* Halsted Press, New York, 1985.

[Farouki88] R. Farouki and V. Rajan. On the numerical condition of polynomials in Bernstein form. *Computer Aided Geometric Design*, Vol 4, No 3, pp 191–216, 1987.

[Golu96] G. H. Golub and C. F. Van Loan. *Matrix Computation*. The Johns Hopkins University Press, Baltimore and London, Third Edition, 1996.

[Heo99] H.-S. Heo, M.-S. Kim, and G. Elber. The intersection of two ruled surfaces. *Computer-Aided Design*, Vol 31, No 1, pp 33–50, January 1999.

[Kim00] M.-S. Kim and G. Elber. Problem reduction to parameter space. *The Mathematics of Surfaces IX*, R. Cipolla and R. Martin (Eds), Springer, London, 2000, pp 82–98.

[Irit00] IRIT 8.0 User's Manual, September 2000, Technion. `www.cs.technion.ac.il/∼irit`.

[Joy99] K. Joy and M. Duchaineau. Boundary determination for trivariate solid. *Proc. of Pacific Graphics 99*, Seoul, Korea, October 5–7 1999, pp 82–91.

[Lane81] J. Lane and R. Riesenfeld. Bounds on a polynomial. *BIT*, Vol 21, pp 112–117, 1981.

[Lore87] W.E. Lorensen and H.E. Cline. Marching Cubes: a high resolution 3D surface construction algorithm. *Computer Graphics*, Vol 21, No 4 (Proc. ACM Siggraph 87), pp 163–169, July 1987.

[Luen84] D. G. Luenberger. *Linear and Nonlinear Programming*. Second Edition, Addison-Wesley, Menlo Park, California, 1984.

[Martin90] R. Martin and P. Stephenson. Sweeping of three-dimensional objects. *Computer-Aided Design*, Vol 22, pp 223–234, 1990.

[Nish90] T. Nishita, T. W. Sederberg and M. Kakimoto. Ray tracing trimmed rational surface patches. *Computer Graphics*, Vol 24, No 4 (Proc. ACM Siggraph 90), pp 337–345, August 1990.

[Taba95] S. Tabachnikov. *Billiards*. Paris: Societe Mathematique de France, 1995.

[Sede88] T. W. Sederberg and R. J. Meyers. Loop detection in surface patch intersections. *Computer Aided Geometric Design*, Vol 5, No 2, pp 161–171, 1988.

[Sede96] T. W. Sederberg and A. K. Zundel. Pyramids that bound purface patches. *Graphical Models and Image Processing*, Vol 58, No 1, pp 75–81, 1996.

[Sher93] E. C. Sherbrooke and N. M. Patrikalakis. Computation of the solutions of nonlinear polynomial systems. *Computer Aided Geometric Design*, Vol 10, No 5, pp 379–405, October 1993.

[Well91] D. Wells. *The Penguin Dictionary of Curious and Interesting Geometry* Penguin Books, New York, 1991.