Subdivision Termination Criteria in Subdivision Multivariate Solvers

Iddo Hanniel and Gershon Elber

Department of Computer Science,
Technion, Israel Institute of Technology,
Haifa 32000, Israel
{iddoh,gershon}@cs.technion.ac.il

Abstract. The need for robust solutions for sets of non-linear multivariate constraints or equations needs no motivation. Subdivision-based multivariate constraint solvers [1–3] typically employ the convex hull and subdivision/domain clipping properties of the Bézier/B-spline representation to detect all regions that may contain a feasible solution. Once such a region has been identified, a numerical improvement method is usually applied, which quickly converges to the root. Termination criteria for this subdivision/domain clipping approach are necessary so that, for example, no two roots reside in the same sub-domain (root isolation). This work presents two such termination criteria. The first theoretical criterion identifies sub-domains with at most a single solution. This criterion is based on the analysis of the normal cones of the multiviarates and has been known for some time [1]. Yet, a computationally tractable algorithm to examine this criterion has never been proposed. In this paper, we present such an algorithm for identifying sub-domains with at most a single solution that is based on a dual representation of the normal cones as parallel hyper-planes over the unit hyper-sphere. Further, we also offer a second termination criterion, based on the representation of bounding parallel hyper-plane pairs, to identify and reject sub-domains that contain no solution.

We implemented both algorithms in the multivariate solver of the IRIT [4] solid modeling system and present examples using our implementation.

1 Introduction

The robust solution of a univariate non-linear equation is considered a difficult problem. The simultaneous solution of sets of multivariate non-linear constraints is typically far more difficult. In recent years, solvers that draw from geometric design developments and are based on recursive subdivision of the constraints represented in the Bézier and/or B-spline forms were developed [1–3]. Coming from the geometric design field, these solvers are very much suited to solving non linear constraints in geometric design. One additional motivation for the use of subdivision based solvers in geometric design may be found in the fact that the domain is typically limited to the domain of curves or surfaces. In other words, the search for simultaneous zeros is conducted in a limited domain only.

Geometric constraint solvers play a crucial role in geometric modeling environments. The basic problem such solvers encounter is computing the simultaneous roots of a set of non-linear equations. Following the work of Lane and Riesenfeld [5], who used a Bernstein-Bézier subdivision scheme for univariate root-finding, many efficient methods have been developed for a large variety of root-finding problems in geometric design and modeling. Examples include ray-surface, curve-curve and surface-surface intersections. Sherbrook and Patrikalakis [3] extended this subdivision approach to solving a set of multivariate polynomial equations given in the Bernstein-Bézier form. A similar approach can also be applied to the more general B-spline representation [1].

These solvers employ subdivision or domain clipping to reduce the size of the domain that is suspected to contain roots [1–3]. This process is combined with the convex hull property of the Bézier/B-spline form to discard sub-domains in which no solution exists. In other words, a constraint whose coefficients are all positive (or all negative) cannot have zeros. This is a simple yet first example of a termination criterion for this type of subdivision solver.

In every subdivision step, the domain that may contain a feasible solution becomes smaller. Nonetheless, the whole-positivity criterion cannot differentiate between sub-domains that hold one-solution or more. Typically, the process is terminated at a pre-defined subdivision depth, by a-priori fixing the minimal sub-domain to be considered. Furthermore, the subdivision approach is relatively slow, and therefore, reducing the number of subdivision steps, whenever possible, is highly desirable. For example, in our solver we are able to eliminate unnecessary subdivisions in the final stage once we can guarantee that there is at most one root in each sub-domain. If a sub-domain is known to contain at most one solution, we can apply a numerical improvement step (e.g., Newton-Raphson iterations) that quickly converges to the root, if it exists. Furthermore, such a condition will guarantee that we have not prematurely terminated the subdivision process before all roots have been isolated correctly, i.e., two ”close” roots will not be considered as a single root.

To the best of our knowledge, this is the first subdivision-based multivariate solver that can guarantee all roots have been isolated, based on geometric conditions, and not terminate at a predefined subdivision depth.

The condition that a sub-domain contains at most one root is thus an important component in a subdivision based multivariate solver. In [6, 7], Sederberg et al. introduced the concepts of a normal cone and a surface bounding cone for curve-curve intersections and loop detection in surface intersections. Elber and Kim [1] generalized these concepts to higher dimensions and used them to define a condition for isolating single solutions. However, their condition was based on the intersection of d cones in $\mathbb{R}^d$, d being the dimension of the problem, which is a computationally difficult problem for which no efficient algorithm has so far been presented.

In this paper, an efficient algorithm is presented for cone intersections in $\mathbb{R}^d$ and hence for identifying sub-domains with at most a single root, following [1]. This algorithm is based on a dual representation of the normal cones using hyper-planes and the unit hyper-sphere in $\mathbb{R}^d$. This representation leads to a simple and efficient algorithm for solving the problem.

We also present an additional sufficient criterion that ensures that a sub-domain contains no roots. The additional condition, and the algorithm for testing it, exploit pairs of parallel bounding hyper-planes on the constraints.

The rest of this paper is organized as follows. In Section 2, we define the problem and summarize the main results obtained in [1]. In Section 3, we present our first criterion and algorithm for identifying intersections of bounding cones at a single point, which correspond to isolations of sub-domains that contain at most a single solution. In Section 4, we present the second criterion and algorithm for purging away sub-domains with a zero-solution. In Section 5, we show examples from our implementation of the algorithm, and finally, we conclude the paper in Section 6.

2 Background

We consider the problem of identifying sub-domains with no or at most a single solution for a set of implicit non-linear multivariate functions in the Bézier/B-spline representation, assuming the dimension of the solution space to be zero (thus, forming a set of discrete isolated roots, in the general, non-degenerate, case). Given d implicit algebraic equations in d variables,

$$\mathcal{F}_i(u_1, u_2, \cdots, u_d) = 0,\ i = 1, \cdots, d, \tag{1}$$

we seek all $\boldsymbol{u} = (u_1, u_2, \cdots, u_d)$ that simultaneously satisfy Equation (1). We will assume that $\mathcal{F}_i$, $i = 1, \cdots, d$, are represented as B-spline or Bézier multivariate scalar functions, i.e.,

$$\mathcal{F}_i = \sum_{i_1} \cdots \sum_{i_d} P_{i_1,\cdots,i_d} B_{i_1,k_{i_1}}(u_1) \cdots B_{i_d,k_{i_d}}(u_d), \tag{2}$$

where $B_{i_j,k_{i_j}}$ are the i'th k_{i_j}-degree Bézier/B-spline basis functions. We repeat, for completeness, the result from Elber and Kim [1] of the condition for the uniqueness of a solution in a sub-domain.

Sederberg and Meyers [6] used the Hodograph as the basis of a termination condition in the intersection of two planar Bézier curves. Two planar curves intersect at most once if their Hodographs share no common direction vector. Sederberg et al. [6, 7] also developed a similar condition for the intersection of two parametric surfaces, where the surface bounding (or tangent) cone plays an important role in loop detection. Elber and Kim [1] generalized this approach to the higher-dimensional problem of intersecting d implicit hyper-surfaces $\mathcal{F}_i(\boldsymbol{u}) = 0$, for $i = 1, \cdots, d$, in $\boldsymbol{R}^d$ (i.e., Equation (1)). Being the gradient, the normal space of an implicit hyper-surface is considerably easier to compute than that of a general parametric hyper-surface. This fact greatly simplifies the computation of the normal bounding cones.

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

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

Furthermore, let $\mathcal{C}^{in}$ denote the set of vectors on or in the cone $\mathcal{C}$, and let $\mathcal{C}^{out}$ denote the set of vectors on or out of the cone $\mathcal{C}$. That is:

$$\mathcal{C}^{in}(\overline{\boldsymbol{v}}, \alpha) = \{\boldsymbol{u} \mid \langle \boldsymbol{u}, \overline{\boldsymbol{v}} \rangle^2 \geq \|\boldsymbol{u}\|^2 \cos^2 \alpha\}.$$
$$\mathcal{C}^{out}(\overline{\boldsymbol{v}}, \alpha) = \{\boldsymbol{u} \mid \langle \boldsymbol{u}, \overline{\boldsymbol{v}} \rangle^2 \leq \|\boldsymbol{u}\|^2 \cos^2 \alpha\}.$$

Note that if $\boldsymbol{u}$ is a vector on $\mathcal{C}$ (and similarly for $\mathcal{C}^{in}$ and $\mathcal{C}^{out}$), then so is the vector $c\boldsymbol{u}$, where $c \in \mathbb{R}^+$. This is a direct consequence of the cone definition as the linearity of the inner product $\langle \boldsymbol{u}, \overline{\boldsymbol{v}} \rangle^2 = \|\boldsymbol{u}\|^2 \cos^2 \alpha$ implies that $\langle c\boldsymbol{u}, \overline{\boldsymbol{v}} \rangle^2 = \|c\boldsymbol{u}\|^2 \cos^2 \alpha$.

For an implicit hyper-surface $\mathcal{F}_i(\boldsymbol{u}) = 0$, we define its normal cone $\mathcal{C}_i^n = \mathcal{C}^n(\overline{\boldsymbol{v}}_i^n, \alpha_i^n)$ in $\mathbb{R}^d$ (see Figure 1 (a)) as the set of all possible normal vectors, $\nabla \mathcal{F}_i(\boldsymbol{u})$, and their scalar multiples. Clearly, we have,

$$\begin{aligned}\nabla \mathcal{F}_i(\boldsymbol{u}) &= \left(\frac{\partial \mathcal{F}_i}{\partial u_1}(\boldsymbol{u}), \frac{\partial \mathcal{F}_i}{\partial u_2}(\boldsymbol{u}), \cdots, \frac{\partial \mathcal{F}_i}{\partial u_d}(\boldsymbol{u}) \right) \\ &= \sum_{i_1} \sum_{i_2} \cdots \sum_{i_d} \left(P^{u_1}_{i_1,i_2,\cdots,i_d}, P^{u_2}_{i_1,i_2,\cdots,i_d}, \cdots, P^{u_d}_{i_1,i_2,\cdots,i_d} \right) \\ &\qquad B_{i_1,k_{i_1}}(u_1) B_{i_2,k_{i_2}}(u_2) \cdots B_{i_d,k_{i_d}}(u_d), \end{aligned} \tag{3}$$

where the terms $P^{u_j}_{i_1,i_2,\cdots,i_d}$ denote the coefficients of the partial derivative of $\mathcal{F}_i$ with respect to u_j, elevated to a common subspace. Then, the normal cone to $\mathcal{F}_i(\boldsymbol{u})$, $\mathcal{C}_i^n = \mathcal{C}(\overline{\boldsymbol{v}}_i^n, \alpha_i^n)$ can be derived, for example, by letting $\overline{\boldsymbol{v}}_i^n$ be the average (normalized to unit-length) of the vectors $\left(P^{u_1}_{i_1,i_2,\cdots,i_d}, P^{u_2}_{i_1,i_2,\cdots,i_d}, \cdots, P^{u_d}_{i_1,i_2,\cdots,i_d} \right)$, $\forall i_1, i_2, \cdots, i_d$, and α_i^n be the maximum angle between $\overline{\boldsymbol{v}}_i^n$ and these vectors.

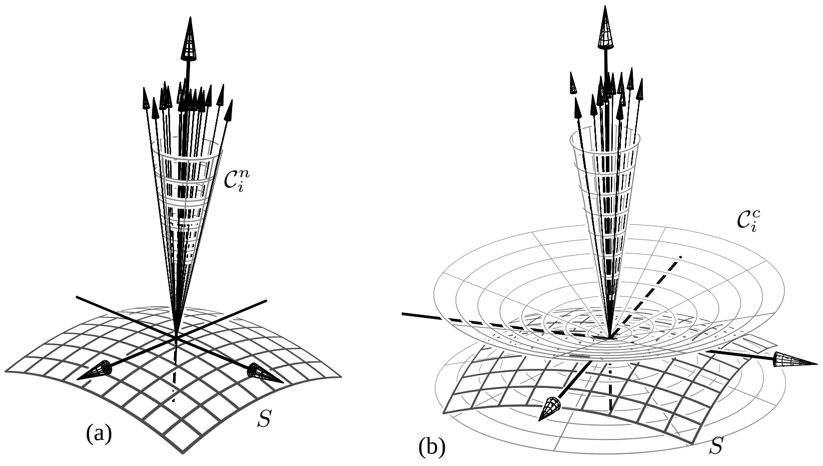

Fig. 1. (a) The normal cone, $\mathcal{C}_i^n$ (in gray) and (b) the complementary tangent cone, $\mathcal{C}_i^c$ (in gray) of a freeform surface, S.

The bounding normal cone obtained by the procedure described above is, in general, not optimal. In order to get an optimal cone, Barequet and Elber [8] proposed an algorithm based on the expected linear time algorithm for minimal spanning spheres [9].

Given the normal cone, $\mathcal{C}_i^n$, we can also define its complementary cone, $\mathcal{C}_i^c$ (see Figure 1 (b)), which contains all vectors that are orthogonal to vectors in $\mathcal{C}_i^n$:

$$\mathcal{C}_i^c = \left\{ \boldsymbol{w} \in \boldsymbol{R}^d \mid \exists \boldsymbol{u} \in \mathcal{C}_i^n \text{ such that } \langle \boldsymbol{u}, \boldsymbol{w} \rangle = 0 \right\}. \tag{4}$$

$\mathcal{C}_i^c$ is also called the tangent cone [6] and it contains the tangent space or all possible tangent directions of the implicit hyper-surface $\mathcal{F}_i(\boldsymbol{u}) = 0$. $\mathcal{C}_i^c$ can easily be derived from $\mathcal{C}_i^n$ as follows (see Figure 1):

$$\mathcal{C}_i^c = \mathcal{C}^{out}(\overline{\boldsymbol{v}}_i^c, \alpha_i^c) = \mathcal{C}^{out}(\overline{\boldsymbol{v}}_i^n, 90 - \alpha_i^n), \tag{5}$$

where $\overline{\boldsymbol{v}}_i^c = \overline{\boldsymbol{v}}_i^n$. In other words, $\mathcal{C}_i^c$ and $\mathcal{C}_i^n$ share the same axis, but have complementary angles. Finally, let $\mathcal{C}_i^c[\boldsymbol{u}^0] \subset \boldsymbol{R}^d$ denote the translation of $\mathcal{C}_i^c$ by $\boldsymbol{u}^0$, and from [1], 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 $\mathcal{F}_i(\boldsymbol{u}^0) = 0$.

The main result obtained in this section, following [1], is:

Theorem 1. *Given d implicit hyper-surfaces $\mathcal{F}_i(\boldsymbol{u}) = 0,\ i = 1, \cdots, d$, in $\boldsymbol{R}^d$, there exist at most one common solution to $\mathcal{F}_i(\boldsymbol{u})$ if*

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

where $\boldsymbol{0}$ is the origin of the coordinate system, and $\mathcal{C}_i^c$ is the complementary tangent cone of $\mathcal{F}_i$.

Proof: Assume that $\boldsymbol{u}^0 \in \boldsymbol{R}^d$ is a common solution of the d equations $\mathcal{F}_i(\boldsymbol{u}) = 0,\ i = 1, \cdots, d$, and 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

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

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

Direct attempts to compute $\bigcap_{i=1}^{d} \mathcal{C}_i^c$ are bound to be highly inefficient. We are now ready to examine a tractable alternative.

3 Identifying Intersections of Bounding Cones

In this section, we present the first result of this paper — an algorithm, based on a dual representation, for identifying intersections of bounding cones in $\boldsymbol{R}^d$, following Theorem 1. Section 3.1 describes our dual representation of the bounding cones in $\boldsymbol{R}^d$, and Section 3.2 presents the algorithm that is based on this representation.

3.1 The Dual Representation of the Bounding Cones

The condition derived in Theorem 1 enables us, in theory, to detect when a sub-domain has at most a single solution. However, in order to verify this condition we have to

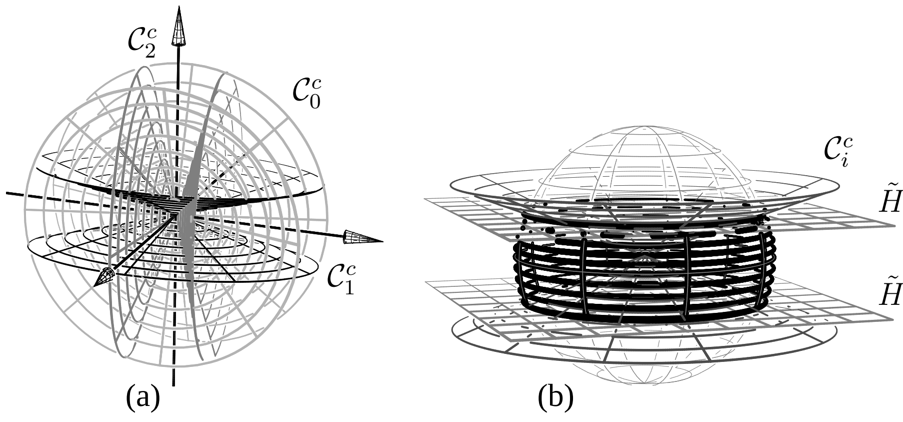

Fig. 2. (a) The intersection of three complementary cones in $\mathbb{R}^3$, $\mathcal{C}_i^c,\ i = 0, 1, 2$. (b) The (thick) strip on the unit sphere is the intersection of the unit sphere and the complementary cone, $\mathcal{C}_i^c$. It is also the intersection of the unit sphere and the region between the parallel delimiting planes, $\tilde{H}_i^+$ and $\tilde{H}_i^-$.

check the intersections of the complementary tangent cones — a difficult problem even in $\mathbb{R}^3$ (see Figure 2(a)). In this section, we consider a dual representation of the tangent cone that enables one to reduce the query of whether the tangent cones intersect at a single point to the much simpler query of intersections of hyper-planes.

Let $\mathcal{C}_i^c(\overline{\boldsymbol{v}}_i, \alpha_i)$ be a given complementary tangent cone and let S^{d-1} denote the unit hyper-sphere in $\mathbb{R}^d$, i.e., $S^{d-1} = \{\boldsymbol{u} \in \mathbb{R}^d \mid \|\boldsymbol{u}\|^2 = 1\}$. Assigning S^{d-1} into the tangent cone definition, we get the intersection of the tangent cone and S^{d-1}:

$$\langle \boldsymbol{u}, \overline{\boldsymbol{v}}_i \rangle^2 = \cos^2 \alpha_i,$$

or

$$\langle \boldsymbol{u}, \overline{\boldsymbol{v}}_i \rangle = \pm |\cos \alpha_i|.$$

In other words, the intersection of the tangent cone $\mathcal{C}_i^c$ and the unit hyper-sphere is delimited by the symmetric parallel hyper-planes $\tilde{H}_i^+ : \langle \boldsymbol{u}, \overline{\boldsymbol{v}}_i \rangle = |\cos \alpha_i|$ and $\tilde{H}_i^- : \langle \boldsymbol{u}, \overline{\boldsymbol{v}}_i \rangle = -|\cos \alpha_i|$ (see Figure 2(b)). In explicit form, these hyper-planes will be written as

$$\tilde{H}_i^{\pm}(u_1, \cdots, u_d) : \ \ v_1^i u_1 + v_2^i u_2 + \cdots + v_d^i u_d = \pm |\cos \alpha_i|,$$

where $\overline{\boldsymbol{v}}_i = (v_1^i, \cdots, v_d^i)$ is the normalized cone axis, α_i is the complementary cone's angle, and $\boldsymbol{u} = (u_1, \cdots, u_d)$ are the unknowns.

Thus, given a complementary tangent cone, its delimiting hyper-planes can be computed as described above. The converse is also true, given a delimiting hyper-plane

$$v_1^i u_1 + v_2^i u_2 + \cdots + v_d^i u_d \pm c = 0,$$

where the coefficient vector $\overline{\boldsymbol{v}}_i$ is normalized and $\pm c \in [-1, 1]$, the complementary tangent cone is simply the cone $\mathcal{C}_i^c(\overline{\boldsymbol{v}}_i, \arccos(c))$. Therefore, there is a one-to-one mapping between the complementary tangent cones and their dual delimiting hyper-planes, over S^{d-1}.

The duality between the tangent cone and its delimiting hyper-planes over S^{d-1} enables us to represent the complementary cones by their delimiting hyper-planes. More precisely, the complementary cone is represented as the strip on the unit hyper-sphere, which is the intersection of the unit hyper-sphere and the region bounded between the delimiting hyper-planes (note the thick lines' strip of S^2 in Figure 2(b)).

Let H_i^+ be the half-space bounded by $\tilde{H}_i^+$ and containing the origin (i.e., the half-space defined by the equation $\langle \boldsymbol{u}, \overline{\boldsymbol{v}}_i \rangle \le |\cos \alpha_i|$), and similarly let H_i^- be the half-space bounded by $\tilde{H}_i^-$ and containing the origin. The strip on the unit hyper-sphere is, therefore, the intersection $H_i^+ \bigcap H_i^- \bigcap S^{d-1}$. The intersection of the d cones can, consequently, be represented as the intersection of the unit hyper-sphere with the regions bounded by the delimiting hyper-planes:

$$\bigcap_{i=1}^{d} \{H_i^+ \bigcap H_i^- \bigcap S^{d-1}\} = S^{d-1} \bigcap_{i=1}^{d} \{H_i^+ \bigcap H_i^-\}.$$

3.2 Algorithm for Identifying the Uniqueness of a Solution

In Section 3.1, we introduced a dual representation of the complementary tangent cones by the intersection of their delimiting hyper-planes and the unit hyper-sphere. This representation enables one to represent the intersection of a set of complementary cones as the intersection of the regions bounded by their delimiting hyper-planes (i.e., an intersection of half-spaces) and the unit hyper-sphere. If the axes of the d complementary cones are linearly independent, then the intersection between the d infinite regions bounded by the delimiting hyper-planes is a bounded convex polytope (see Figure 3 for examples in $\boldsymbol{R}^3$), where the vertices of the polytope are the d-dimensional points of intersection between subsets of d hyper-planes.[1] The following gives the correspondence between the intersection of complementary cones and the intersection of the convex polytope and S^{d-1}.

Lemma 1. *$\bigcap_{i=1}^{d} \mathcal{C}_i^c$ contains a vector other than $\mathbf{0}$, if and only if the intersection $S^{d-1} \bigcap_{i=1}^{d} \{H_i^+ \bigcap H_i^-\}$ is not empty, where $\mathcal{C}_i^c$ are the complementary cones and H_i^+ and H_i^- are the bounding half-spaces.*

Proof: If there is a non-trivial intersection between the complementary cones, then there exists a vector $\boldsymbol{u} \neq \mathbf{0}$ such that $\boldsymbol{u} \in \mathcal{C}_i^c$ for all i. Therefore, for all $c \in R$, $c\boldsymbol{u} \in \mathcal{C}_i^c$ for all i. In particular, there is a unit-length vector $\overline{\boldsymbol{u}}$ (i.e., $\overline{\boldsymbol{u}} \in S^{d-1}$), such that $\overline{\boldsymbol{u}} \in \mathcal{C}_i^c$ for all i. However, by duality, $S^{d-1} \bigcap \mathcal{C}_i^c = S^{d-1} \bigcap \{H_i^+ \bigcap H_i^-\}$, and so we have $\overline{\boldsymbol{u}} \in S^{d-1} \bigcap \{H_i^+ \bigcap H_i^-\}$ for all i. Thus, $S^{d-1} \bigcap_{i=1}^{d} \{H_i^+ \bigcap H_i^-\}$ is not empty.

On the other hand, if $S^{d-1} \bigcap_{i=1}^{d} \{H_i^+ \bigcap H_i^-\}$ is not the empty set, then there exists a unit vector $\overline{\boldsymbol{u}} \in S^{d-1} \bigcap_{i=1}^{d} \{H_i^+ \bigcap H_i^-\}$. This means that $\overline{\boldsymbol{u}} \in S^{d-1} \bigcap \mathcal{C}_i^c$ for all i,

[1] In the degenerate case where the axes are not linearly independent, the polytope will be unbounded and thus $S^{d-1} \bigcap_{i=1}^{d} \{H_i^+ \bigcap H_i^-\}$ will not be empty, therefore, by Lemma 1, $\bigcap_{i=1}^{d} \mathcal{C}_i^c$ can contain more than a single point and the termination condition will fail.

and therefore $\bigcap_{i=1}^{d} \mathcal{C}_i^c$ contains a vector other than **0**. ■

Since $\bigcap_{i=1}^{d}\{H_i^+ \bigcap H_i^-\}$ is the bounded convex polytope, then from Lemma 1 it follows that there is an intersection of the complementary cones at a location other than the origin if and only if there is an intersection of the convex polytope and S^{d-1} in $\mathbb{R}^d$. While computing the intersection of a convex polytope with S^{d-1} in $\mathbb{R}^d$ can be a non-trivial task, for our needs a smaller task is actually required. We are only interested in knowing whether the intersection of the strips on the hyper-sphere is an empty set. If the intersection is an empty set, then the complementary cones intersect only at the origin. But, because of convexity, an empty set can result only if all vertices of the convex polytope are inside the unit hyper-sphere (see Figure 3(b)). Thus, herein, we simply need to compute all the vertices of the polytope and check whether any of them is outside the unit hyper-sphere (i.e., if their distance from the origin is larger than one).

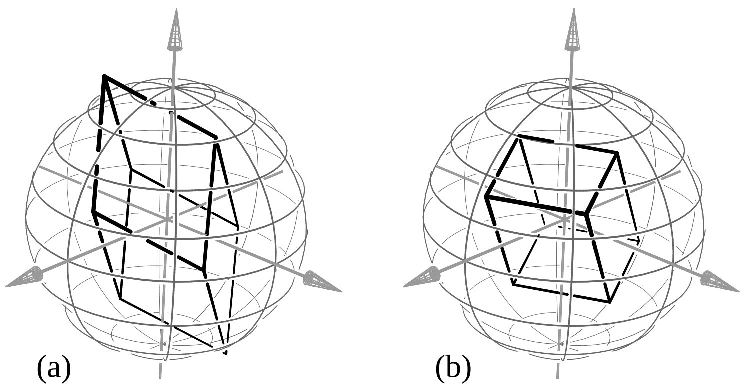

Fig. 3. (a) A non-empty intersection corresponding to a vertex of the polytope outside the unit sphere. (b) All vertices of the polytope are inside the unit sphere and hence the complementary cones intersect only at the origin.

The number of vertices of the convex polytope is the number of possible combinations of d intersecting hyper-planes. In the general case of $2d$ hyper-planes, the number of vertices is $\binom{2d}{d}$. However, since in our case the hyper-planes are arranged in parallel pairs, only 2^d intersections can occur.[2] Furthermore, because of symmetry, if a vertex $\boldsymbol{u}$ is inside (outside) the unit hyper-sphere, then so will its antipodal vertex, $-\boldsymbol{u}$. Thus, it is sufficient to check at most 2^{d-1} combinations of hyper-plane intersections.

Computing the intersection of d hyper-planes amounts to solving the set of d linear equations:

$$v_1^i u_1 + v_2^i u_2 + \cdots + v_d^i u_d = \pm|\cos\alpha_i|, \quad i = 1 \cdots d \tag{6}$$

where the signs of the terms on the right change according to the combination of hyper-planes we are checking.

[2] This can be viewed as a d-digit binary number, each digit representing a cone, where one can choose either a positive or a negative delimiting hyper-plane but not both.

Solving Equation (6) can be done using any standard method such as LU or QR factorization [10]. Note that the matrix inversion/factorization itself needs to be performed only once, and then for each permutation only backtracking is performed.

If we assume factorization of a $d \times d$ matrix takes $O(d^c)$ operations, where $2 < c < 3$ (and for practical purposes $c = 3$), then computing all the 2^{d-1} solutions takes $O(d^c + 2^{d-1}d^2)$ operations.[3] This can be improved, however, using the symmetry of the solution vector. Let $b_i = |\cos\alpha_i|$, for $i = 1, \cdots, d$. Then, the system of equations $V\boldsymbol{u} = \{\pm b_i\}$, presented in Equation (6), can be rewritten as:

$$V\boldsymbol{u} = \Sigma_{i=1}^{d} \pm b_i \boldsymbol{e}_i,$$

where $\boldsymbol{e}_i$ are the standard base vectors. The solution is thus:

$$\boldsymbol{u} = \Sigma_{i=1}^{d} \pm b_i V^{-1} \boldsymbol{e}_i.$$

Therefore, if we solve for $\boldsymbol{c}_i = V^{-1}\boldsymbol{e}_i$ once, we can encode all 2^{d-1} solutions as the combinations:

$$+b_1\boldsymbol{c}_1 \cdots \pm b_i\boldsymbol{c}_i \cdots \pm b_d\boldsymbol{c}_d.$$

We can code the 2^{d-1} solutions by their sequences of pluses and minuses. Assigning 1 for a $+$ and a 0 for a $-$, we get a d-digit binary number. For example, the solution, $+b_1\boldsymbol{c}_1 \cdots - b_i\boldsymbol{c}_i \cdots - b_d\boldsymbol{c}_d$, corresponds to the d-digit number $100...00$.

Now with the solution $\boldsymbol{u}$, computing a solution with an identical code except for one bit takes only $O(d)$ operations. For example, for $d = 3$ computing $\boldsymbol{u}_{101}$, after we have already computed $\boldsymbol{u}_{100}$, amounts to computing $\boldsymbol{u}_{100} + 2b_3\boldsymbol{c}_3$, i.e., a single vector addition, taking $O(d)$ operations. Thus, if we order the solutions according to Gray coding [11], where adjacent binary numbers differ by a single bit, the overall time for computing all the solutions will be reduced to $O(d^c + 2^{d-1}d)$.

We can now summarize the full algorithm for identifying whether a sub-domain contains at most a single solution:

1. Compute the normal cones, $\mathcal{C}_i^n$, $i = 1, \cdots, d$, for each hyper-surface as described in Section 2.
2. For each cone, extract the pair of delimiting hyper-planes as described in Section 3.1.
3. For each of the 2^{d-1} combinations in Equation (6):
   (a) Solve the set of equations. Let $\boldsymbol{u}$ be the d-dimensional solution vector.
   (b) If $\|\boldsymbol{u}\| \geq 1$ return False, i.e., the sub-domain is not guaranteed to have at most a single solution.
4. Return True. The sub-domain is guaranteed to have at most a single solution.

4 Purging Away Zero-solution Domains

The algorithm presented in Section 3 identifies whether a sub-domain has at most a single solution. However, it does not guarantee that a solution exists in the sub-domain.

[3] 2^{d-1} times doing back substitution of the solution, where each back substitution takes $O(d^2)$ operations.

Thus, we might terminate the subdivision and start the numerical iterations only to find that no root exists in this sub-domain. Such a numerical step, starting at a sub-domain with no zeros, can require a relatively large number of iterations, since the initial point might be far from a root. Therefore, we seek to purge away, as much as possible, sub-domains that contain no solution. In this section, we present a second criterion for identifying sub-domains with no solution, in addition to the whole-positivity criterion presented in the introduction.

Recall the scalar Bézier/B-spline form of $\mathcal{F}_i$, in Equation (2). We consider them as hyper-surfaces in $\mathbb{R}^{d+1}$, and use the Bézier/B-spline representation to bound the hyper-surface with two parallel hyper-planes. We denote by *promotion* the process of converting the *scalar* function $\mathcal{F}_i(u_1, u_2, \cdots, u_d)$ to its *vector* function counterpart $\hat{\mathcal{F}}_i : \mathbb{R}^d \rightarrow \mathbb{R}^{d+1}$, $\hat{\mathcal{F}}_i = (u_1, u_2, \cdots, u_d, \mathcal{F}_i)$. Promotion is performed by employing the nodal points, also known as the Greville abssica [12], for the first d dimensions. For the Bézier case, the nodal points are simply $(i_1/k_{i_1}, i_2/k_{i_2}, \cdots, i_d/k_{i_d})$.

As the subdivision process progresses, the smaller and smaller sub-domains are likely to become almost (hyper-) planar. Hence, we can bound $\hat{\mathcal{F}}_i$ by two parallel hyper-planes.[4] These two bounding hyper-planes are constructed as follows. Compute the unit normal, $\overline{\boldsymbol{n}}_i = (n_1, \cdots, n_{d+1}) \in \mathbb{R}^{d+1}$, of $\hat{\mathcal{F}}_i$ at the midpoint of the sub-domain. Then, project all control points of $\hat{\mathcal{F}}_i$ onto $\overline{\boldsymbol{n}}_i$.

Denote by $\boldsymbol{u}_{max} \in \mathbb{R}^{d+1}$ (resp. $\boldsymbol{u}_{min} \in \mathbb{R}^{d+1}$) the point on $\overline{\boldsymbol{n}}_i$ that is the maximal (resp. minimal) projection of the control points onto $\overline{\boldsymbol{v}}_i$. Then, the $(d+1)$-dimensional parallel bounding hyper-planes of $\mathcal{F}_i$ are:

$$\langle \boldsymbol{u} - \boldsymbol{u}_{min}, \overline{\boldsymbol{n}}_i \rangle = 0, \qquad \langle \boldsymbol{u} - \boldsymbol{u}_{max}, \overline{\boldsymbol{n}}_i \rangle = 0,$$

where $\boldsymbol{u} \in \mathbb{R}^{d+1}$.

Since we are only interested in bounding $\mathcal{F}_i = 0$, we only need the intersection of these two $(d+1)$-dimensional hyper-planes with the $u_{d+1} = 0$ hyper-plane. Eliminating the u_{d+1} coordinate, we remain with the d-dimensional hyper-planes bounding $\mathcal{F}_i = 0$:

$$\tilde{K}_i^{min} : \ \ \langle \boldsymbol{u}, \overline{\boldsymbol{n}}_i \rangle|_{u_{d+1}=0} = u_1 n_1^i + u_2 n_2^i + \cdots + u_d n_d^i = b_i^{min},$$

and

$$\tilde{K}_i^{max} : \ \ \langle \boldsymbol{u}, \overline{\boldsymbol{n}}_i \rangle|_{u_{d+1}=0} = u_1 n_1^i + u_2 n_2^i + \cdots + u_d n_d^i = b_i^{max},$$

where $b_i^{min} = \langle \boldsymbol{u}_{min}, \overline{\boldsymbol{n}}_i \rangle$ and $b_i^{max} = \langle \boldsymbol{u}_{max}, \overline{\boldsymbol{n}}_i \rangle$.

As in Section 3.1, we denote by K_i^{min} and K_i^{max} the half-spaces bounded by $\tilde{K}_i^{min}$ and $\tilde{K}_i^{max}$, oriented so that $\mathcal{F}_i = 0$ is on their positive side. $\mathcal{F}_i = 0$ is thus bounded in the region $K_i^{min} \bigcap K_i^{max}$. Given a pair of bounding hyper-planes $\tilde{K}_i^{min}$ and $\tilde{K}_i^{max}$ for each constraint $\mathcal{F}_i = 0$, we have to determine whether the d-dimensional polytope defined by $\bigcap_{i=1}^{d} \{K_i^{min} \bigcap K_i^{max}\}$ is entirely outside of the sub-domain. Being outside the sub-domain, no zeros can exist in this sub-domain.

This intersection problem can be solved using linear programming methods (see, for example, [13][Chapter 29] or [9][Chapter 4]), by adding the $2d$ half-spaces $\tilde{K}_i^{min}$ and $\tilde{K}_i^{max}$, for $i = 1, \cdots, d$, to the $2d$ hyper-planes of the sub-domain itself. If there is no

[4] These two parallel hyper-planes are unrelated to the pairs of hyper-planes used in Section 3.

feasible solution to the linear programming problem defined by these $4d$ hyper-planes, the solution is guaranteed to be entirely outside of the domain.

While the problem can indeed be solved using general linear programming methods, in our implementation and for reasons of efficiency, we chose to solve it in a different way. The configuration of the problem resembles the configuration in Section 3.2, namely identifying whether an intersection of parallel-hyper-plane pairs is inside or outside some region. Therefore, we computed the 2^d vertices of the polytope using the same procedure that was presented in Section 3.2. Here instead of switching between $+b_i$ and $-b_i$, we now switch between b_i^{min} and b_i^{max}. If all the vertices of the polytope are outside of one half-space bounding the sub-domain, then the solution is verified to be entirely outside of the sub-domain.

From our experience, solving the problem in the same manner as in Section 3.2 creates little overhead, since most of the algorithm's time is spent on constructing the pairs of hyper-planes from the $\mathcal{F}_i$ constraints. On the other hand, computing all the vertices has the advantage that now we can use all these intersection locations for better clipping of the sub-domain (see possible future extensions in Section 6).

5 Examples

Figures 4 and 5 present a simple example of the solution of two bivariate constraints. The two constraints are expressed as bicubic Béziers:

$$\mathcal{F}_1(u,v) = \sum_{i=0}^{3}\sum_{j=0}^{3} P_{ij}B_i(u)B_j(v), \quad P_{ij} = \begin{bmatrix} 0 & 2.2 & 1.1 & 1.1 \\ -1 & -1 & -1 & 1 \\ -1 & 1 & 1 & 1 \\ -1 & -1 & -2 & 0 \end{bmatrix}, \quad u,v \in [0,1], \tag{7}$$

and $\mathcal{F}_2(u,v) = \mathcal{F}_1(v,u)$ (reflected along the diagonal). The simultaneous solution of $\mathcal{F}_1(u,v) = \mathcal{F}_2(u,v) = 0$, for $u,v \in [-1,2]$ yields seven roots. Figure 4 (a) presents the entire subdivision tree for a subdivision tolerance of 10^{-3}. In (b), a zoom-in of the thick square area near the center of Figure 4 (a)) is presented. The whole region between these two (close) roots is finely subdivided, only to determine that no other root exists in between.

Figure 5 (a) presents the same subdivision tree, but this time with the introduced single solution cone test, presented in Section 3. The subdivision tree is now (compared to Figure 4 (a)) much smaller. A zoom-in on the region between the two closest roots in Figure 5 (b) reveals how efficiently the roots were isolated, compared to Figure 4. Only in the middle, between the two roots in Figure 5 (a), were some excessive subdivisions applied. This is due to the fact that the two zero sets of $\mathcal{F}_1$ and $\mathcal{F}_2$ are almost parallel, rendering the cone test invalid.

In Figure 6, the same subdivision tree with cells that passed both the single solution test and the parallel hyper-plane test (Section 4) are presented. The numerical step will be performed only on the marked cells. We can see that the parallel hyper-plane test not only significantly reduced the total number of cells in the subdivision tree but also reduced the number of cells on which a numerical step is performed. Moreover, the sub-domains where the two zero sets of $\mathcal{F}_1$ and $\mathcal{F}_2$ are almost parallel, rendering the cone

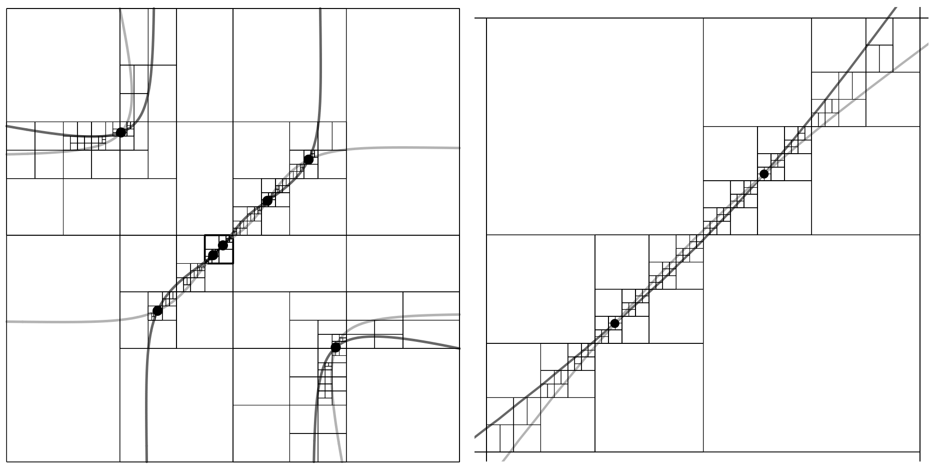

Fig. 4. (a) The zeros of two implicit bi-cubics, $\mathcal{F}_1$ and $\mathcal{F}_2$ (in two shades of gray) along with their simultaneous zero points (see Equation (7)), without the single solution cone test. In (a), the seven zero points are presented along with the subdivision tree. (b) shows a zoom-in over the thick square region near the center of (a). See also Figure 5.

test invalid, were successfully detected and purged away by the parallel hyper-planes test.

All examples presented in this section used an implementation that incorporated the presented algorithms into the multivariate solver of the IRIT [4] solid modeling system.

6 Conclusions

In this paper, we have presented efficient and simple algorithms for identifying no and single solution sub-domains in a set of d implicit constraints in d unknowns. The algorithms are based on a dual representation of the normal cones as hyper-planes over the unit hyper-sphere in $\mathbb{R}^d$ and on pairs of bounding hyper-planes. These algorithms offer ways to improve the performance of subdivision based solvers such as the ones presented in [1–3].

While the single solution test provides a guarantee that all separable roots in the solution set have been isolated correctly, the computational costs of computing these termination criteria are not trivial and effort should be made to optimize these costs. We plan to compare our single solution test to other root isolation approaches such as multivariate Sturm sequences [14] and the Kantorovich theorem [15]. The ideas presented in this paper can be extended for further improvement of subdivision-based solvers. The solution is guaranteed to be within the polytope bounded by the hyper-plane pairs. Thus, the vertices of the bounding polytope, which were computed in Section 4, can be further used for clipping the domain, and not just for purging away zero-solution domains, as was presented.

The proper handling of under- (and over-) constrained systems, when the number of degrees of freedom is larger (smaller) than the number of constraints and the zero-set

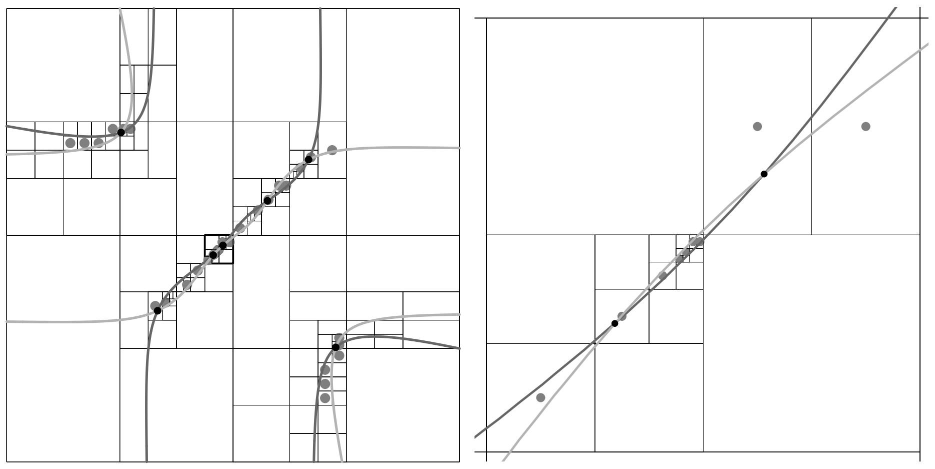

Fig. 5. (a) The zeros of two implicit bi-cubics, $\mathcal{F}_1$ and $\mathcal{F}_2$ along with their simultaneous zero points (see Equation (7)), with the introduced single solution test. The zero points are marked by black dots. In (a), the seven zero points are presented along with the subdivision tree. (b) shows a zoom-in over the thick square region near the center of (a). Compare with Figure 4. The cells where the subdivision process terminated because of the successful single solution test are marked by gray dots. Compare with Figure 6.

is a k-manifold, $k > 0$, is also of major interest in geometric applications. Surface-surface intersection is just one simple instance. While the extension of Theorem 1 to under-constrained systems is not trivial, the bounding hyper-planes criterion (presented in Section 4) can also be employed in these systems. Given an under-constrained system, the polytope that is the intersection of the parallel hyper-plane pairs is unbounded. Still, by using linear programming methods, we can purge away sub-domains with no solutions. If the sub-domain potentially contains roots, we can intersect the unbounded polytope and the hyper-planes bounding the sub-domain itself to receive a smaller sub-domain. It remains to be seen whether this scheme is beneficial. We leave these extensions for future work.

7 Acknowledgment

The authors are grateful to Avraham Sidi and Moshe Israeli for their help in reducing the complexity of solving Equation (6). This research was supported in part by the Israel Science Foundation (grant No. 857/04) and in part by European FP6 NoE grant 506766 (AIM@SHAPE).

References

1. Elber, G., Kim, M.S.: Geometric constraint solver using multivariate rational spline functions. In: Proceedings of the Symposium on Solid Modeling and Applications 2001, Ann Arbor, Michigan (2001) 1–10

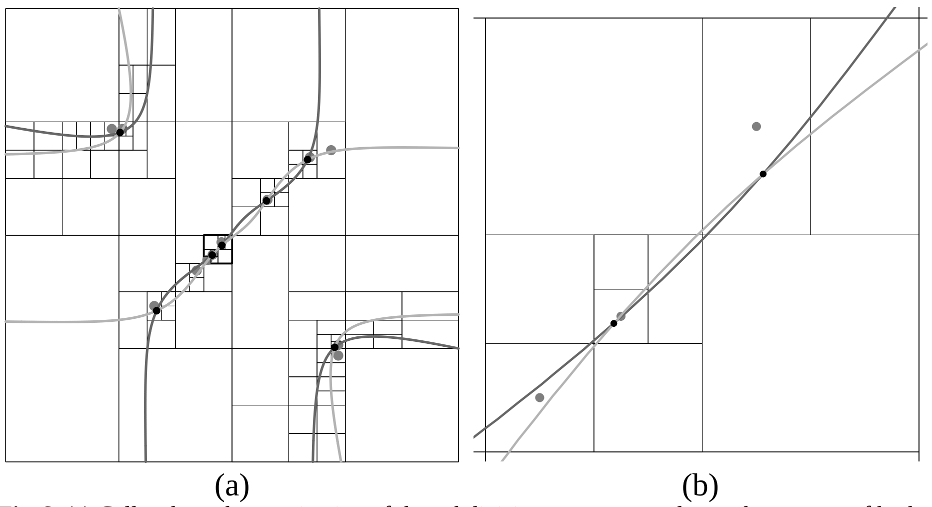

Fig. 6. (a) Cells where the termination of the subdivision process was due to the success of both the single solution test and the parallel hyper-plane test. Again, cells where the single solution test was successful are marked by gray dots. (b) shows a zoom-in over the thick square region near the center of (a). Compare with Figure 5.

2. Mourrain, B., Pavone, J.P.: Subdivision methods for solving polynomial equations. Technical report, INRIA Sophia-Antipolis (2005)
3. Sherbrooke, E.C., Patrikalakis, N.M.: Computation of the solutions of nonlinear polynomial systems. Computer Aided Geometric Design **10**(5) (1993) 279–405
4. Elber, G.: The IRIT 9.5 User Manual. (2005) http://www.cs.technion.ac.il/ irit.
5. Lane, J., Riesenfeld, R.: Bounds on a polynomial. BIT **21** (1981) 112–117
6. Sederberg, T.W., Meyers, R.J.: Loop detection in surface patch intersections. Computer Aided Geometric Design **5**(2) (1988) 161–171
7. Sederberg, T.W., Zundel, A.K.: Pyramids that bound surface patches. Graphical Models and Image Processing **58**(1) (1996) 75–81
8. Barequet, G., Elber, G.: Optimal bounding cones of vectors in three and higher dimensions. Information Processing Letters **93** (2005) 83–89
9. de Berg, M., van Kreveld, M., Overmars, M., Schwarzkopf, O.: Computational Geometry, Algorithms and Applications. 2nd edn. Springer, New York (1998)
10. Golub, G.H., Loan, C.F.V.: Matrix Computation. 3rd edn. The Johns Hopkins University Press, Baltimore and London (1996)
11. Weisstein, E.W.: (Gray code) From MathWorld — A Wolfram Web Resource, http://mathworld.wolfram.com/GrayCode.html.
12. Farin, G.E.: Curves and Surfaces for Computer Aided Geometric Design: A Practical Guide. 4th edn. Academic Press (1996)
13. Cormen, T.H., Leiserson, C.E., Rivest, R.L., Stein, C.: Introduction to Algorithms. 2nd edn. MIT Press and McGraw-Hill (2001)
14. Milne, P.S.: On the solutions of a set of polynomial equations. In: Symbolic and Numerical Computation for Artificial Intelligence. Academic Press (1992) 89–102
15. Dennis, J.E., Schnabel, R.B.: Numerical Methods for Unconstrained Optimization and Non-linear Equations. Prentice Hall Series in Computational Mathematics. Prentice Hall Inc. (1983)