The Visual Computer

A projectively invariant intersection test for polyhedra

Federico Thomas,
Carme Torras

Institut de Robòtica i Informàtica Industrial (CSIC-UPC),
Llorens Artigas 4-6, 2 planta, 08028 Barcelona, Spain
E-mail: {fthomas,ctorras}@iri.upc.es

Published online: ?? ?? 2002

Although intersection relations are projectively invariant, most existing intersection detection tests for arbitrary polyhedra can give different results before and after a nonsingular arbitrary projective transformation of the polyhedra under test. This paper presents a projectively invariant intersection test for general polyhedra whose only numerical part is the computation of 4×4 determinants of homogeneous vertex coordinates. Degeneracies are resolved using a technique of symbolic infinitesimals which also reduces to the computation of 4×4 determinants. This greatly simplifies the implementation of the test in hardware. Moreover, its projective invariance permits applying it at any point in the graphics pipeline. Since no auxiliary geometric entities need to be computed, the presented test can be concisely expressed as a Boolean formula, instead of a procedure.

Key words: Intersection detection – Projective invariance – Degenerate configurations – 4×4 determinant method

Correspondence to: F. Thomas

1 Introduction

Intersection detection is a fundamental geometric operation that arises in many computer graphics and robotics applications (O'Rouke 1998). With the rise of new applications in virtual reality, simulation, and physically based animation it is becoming more important, as it lies at the innermost part of algorithms performing collision detection (Jiménez et al. 2001). Most collision detection libraries available nowadays require that objects be modeled in terms of convex polyhedra, since efficient algorithms exist for detecting intersections between such entities (Mount 1997). Nonconvex objects are then dealt with by decomposing them into convex polyhedral pieces. A first drawback of this approach is that the decomposition introduces many fictitious edges and faces that need to be checked for interference. But a second, more fundamental, drawback is that algorithms based on such decompositions do not always operate properly on objects obtained as a result of projective transformations, because the result of applying such transformations on a convex shape is not necessarily convex (see Fig. 1 for examples that will become clear in Sect. 2.1).

An alternative object representation used by other collision detection libraries consists of describing object boundaries as collections of polygonal faces, which are specified by the coordinates of their vertices ordered always either clockwise or counterclockwise as seen from the outer side. Interference detection algorithms based on this representation may also fail to operate properly. The reason is simple: after a reflection, edges oriented counterclockwise around a face will appear oriented clockwise and vice versa.

Thus, assuming convexity or considering polyhedra boundaries as oriented 2D manifolds should be avoided if one looks for a projectively invariant intersection test. Let us now justify why projective invariance is important. Any 3D geometric entity undergoes a certain number of projective transformations before being represented on the screen, according to the desired viewpoint and scale, location of the entity in the scene, etc. A projectively invariant intersection test guarantees that we can perform intersection detection at any convenient part of this process, which is usually implemented in hardware and commonly known as "viewing pipeline".

In addition to projective invariance, one would like to avoid computing auxiliary geometric entities, such as intersection points or fictitious edges and faces,

The Visual Computer (2002) 18:???–???
Digital Object Identifier (DOI) 10.1007/s003710200158

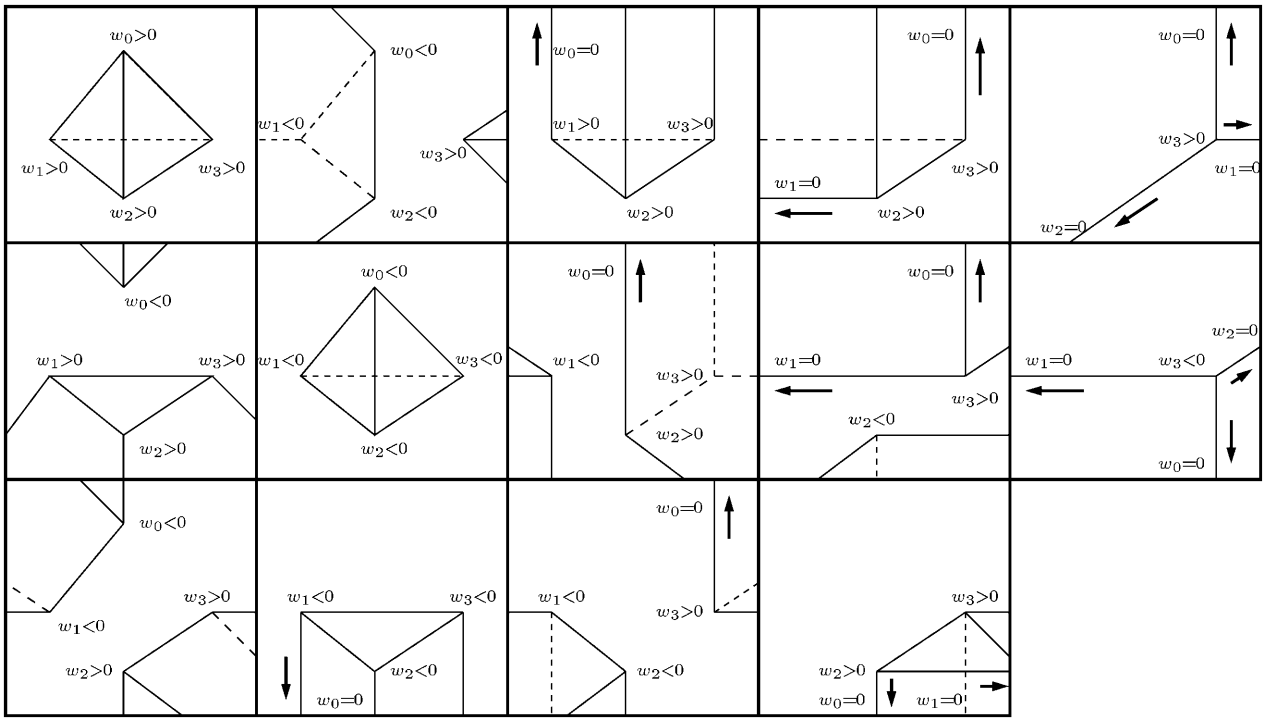

Fig. 1. Homogeneous tetrahedra (Niizeki and Yamaguchi 1994)

since this increases the cost of processing and may introduce round-off errors.

Algorithms for intersection detection between polyhedra are based on interference tests between lower-dimensional entities. Table 1 lists all possible pairs of polytopes embedded in the 1D, 2D, and 3D Euclidean space, together with references to the tests proposed for the corresponding intersection relation. Depending on the application, particularities of the involved polytopes can be exploited to attain some degree of simplicity or efficiency. For example, very efficient algorithms can be designed by constraining polytopes to be simplices (i.e., segments, triangles, and tetrahedra) and/or flats (i.e., point, lines, and planes) (Niizeki and Yamaguchi 1994; Yamaguchi and Niizeki 1997; Möller 1997; Held 1997; Yamaguchi 1998).

All the general polyhedron–polyhedron tests reduce to a series of point-polyhedron and segment-polygon interference tests. One of them (Boyse 1979) further reduces both latter tests to a series of point-polygon tests. Since the efficient resolution of the point-polygon problem is an ubiquitous need in many geometric applications, it has received a lot of attention in the literature (Haines 1994). Effective projectively invariant algorithms, based on the computation of 3×3 determinants, have been proposed for this test (Niizeki and Yamaguchi 1994). Nevertheless, the reduction of polyhedral interference detection to point-polygon tests requires computing intersection points between edges of one polyhedron and faces of the other, something which we would like to avoid.

A way to avoid the computation of auxiliary geometric entities is to formulate polyhedral interference detection as the evaluation of a Boolean formula that depends only on the features in the boundary of the polyhedra. Canny (1987) proposed a polyhedron–polyhedron test of this kind for polyhedra with convex faces, which was generalized by Thomas and Torras (1994) to handle general polyhedra. This latter test, which reduces to a Boolean combination of signs of vertex determinants, constitutes the basis of the present work.

Before discussing this test further, let us briefly mention that Boolean operations on polyhedral features were studied in the 1970s and 1980s in the context of boundary evaluation. Requicha and Voelcker (1982) provide a review of several possible approaches. Later work in this area has mainly focussed on making these approaches robust, i.e., dealing with degen-

Table 1. Pairs of polytopes of different dimensions embedded in 1D, 2D, and 3D space. If their relative configuration allows them to be embedded in a lower-dimensional linear subspace, a degeneracy arises (simple, if the difference between the dimension of the ambient space and that of the subspace is one, and double, if this difference is two)

Dimension	1	2	3
Embedded polytopes	point segment	point segment polygon	point segment polygon polyhedron
Intersection tests	(segment, point)	(segment, point)* (Paeth 1990)	(segment, point)**
	(segment, segment)	(segment, segment) (Prasad 1991; Antonio 1992; Niizeki and Yamaguchi 1994; O'Rouke 1998)	(segment, segment)*
		(polygon, point) (Haines 1994; Horn and Taylor 1989; Niizeki and Yamaguchi 1994; O'Rouke 1998)	(polygon, point)*
		(polygon, segment)	(polygon, segment)
		(polygon, polygon)	(polygon, polygon)
			(polyhedron, point) (Kalay 1982; Lane et al. 1984; Pinto-Carvalho and Roma-Cavalcanti 1994; Thomas and Torras 1994; O'Rouke 1998)
			(polyhedron, segment) (Thomas and Torras 1994)
			(polyhedron, polygon)
			(polyhedron, polyhedron) (Boyse 1979; Canny 1987; Thomas and Torras 1994)
Numerical part	2×2 determinants (signed length of projective segments)	3×3 determinants (signed areas of projective triangles)	4×4 determinants (signed volumes of projective tetrahedra)

* Simple degeneracy ** Double degeneracy

eracies and controlling numerical errors, by either using exact numeric computation (Fortune 1997) or increasing the reliability of inexact methods (Hoffmann et al. 1989; Chubarev 1999).

Tests based on the computation of determinant signs can be easily proved to be projectively invariant. Now, all interference tests for the bidimensional case can be expressed in terms of 3×3 determinant signs, as shown in Niizeki and Yamaguchi (1994), while the polyhedron–polyhedron, point–polyhedron, and edge–polyhedron interference tests can be solved by computing 4×4 determinant signs (Thomas and Torras 1994).

The problem with these two last works is that, in the former, degenerate situations were handled on an ad hoc basis, while, in the latter, they were simply not treated.

Degenerate situations in interference tests arise when the two involved polytopes can be embedded in a space of lower dimension than that in which they coexist. Note that some tests must always operate on degenerate situations. For example, if a point is within a polygon in space, it necessarily lies on the polygon plane. Handling all degenerate cases is somewhat tricky in 2D and is even less straightforward in 3D.

The elegance of all determinant-based tests is invariably impaired when auxiliary geometric elements need to be computed or when geometric degeneracies must be handled in a different ad hoc way. Here we provide a way around these difficulties and present a projectively invariant intersection test for polyhedra where the only numerical part is the computation of 4×4 determinants, including the resolution of degeneracies.

Concerning computational efficiency, note that many basic geometric tests other than interference detection – such as classification, containment, and depth priority tests – can be performed by computing sets of determinants (Yamaguchi 1988), which has motivated the search for efficient determinant computations using either hardware – i.e., the triangle processor and its successor, the polygon engine (Yamaguchi 1988) – or software schemes (Bronnimann and Yvinec 2000).

This paper is structured as follows. Section 2 introduces the required concepts used throughout this paper and gives a brief description of the functions and predicates associated with the basic contacts between two polyhedra. In Sect. 3 the proposed algorithm for intersection detection is fully described, leaving degenerate situations out of the discussion. How these situations are resolved by simply redefining the basic predicates is explained in Sect. 4. Section 5 summarizes the main points and implementation issues and discusses further research.

2 Preliminaries

Incidence relations are projectively invariant. Actually, the domain of projective geometry is essentially that of incidence relations. Thus, it seems reasonable to work directly in projective space by using the well-known homogeneous coordinates. Below, we briefly review them, and we introduce some notation as well as the two basic predicates that will be the building blocks of our intersection detection test.

2.1 Homogeneous coordinates and projective transformations

A point v_0 in 3D space will be represented by homogeneous coordinates (Bloomenthal and Rokne 1994) by means of a four-component nonzero row vector, written as $\mathbf{v}_0 = (x\ y\ z\ w)$. Any nonzero multiple of this vector $\lambda\mathbf{v}_0 = (\lambda x\ \lambda y\ \lambda z\ \lambda w)$ represents the same point v_0. To obtain the corresponding Cartesian coordinates of this point, we divide each component by w, unless $w = 0$. If $w = 0$, the homogeneous coordinate vector represents a point at infinity in the direction of the 3D vector $(x\ y\ z)$, which is not representable in ordinary Cartesian coordinates. The set of points with $w = 0$ is called the plane at infinity. The w's are called the weights (or scale factors) of the homogeneous coordinate vectors. The set containing the 3D space together with its points at infinity is called the real projective space of dimension 3. Note that anti-podal infinite points in projective space are identified.

Any nonsingular 4×4 matrix for which the product of the $(4, 4)$ element with the determinant of the upper left 3×3 component of the matrix is nonzero can be seen as a transformation in projective space which can always be decomposed into a sequence of scale, shear, rotation, translation, and perspective transformations in Euclidean space (Thomas 1991). We explicitly exclude singular transformations from our analysis, because they correspond to projections onto planes and lines. Actually, we are only concerned with transformations which keep intersections invariant and the projections of two polyhedra might intersect while they are far apart.

Since our basic geometric elements will be tetrahedra, instead of points, we next analyze how they are transformed by projective transformations.

Consider the following linear combination of the homogeneous coordinates of four points (say v_0, v_1, v_2, and v_3):

$$\mathbf{v} = \alpha_0\mathbf{v}_0 + \alpha_1\mathbf{v}_1 + \alpha_2\mathbf{v}_2 + \alpha_3\mathbf{v}_3 = (\mathbf{v}_0\mathbf{v}_1\mathbf{v}_2\mathbf{v}_3)\begin{pmatrix}\alpha_0\\ \alpha_1\\ \alpha_2\\ \alpha_3\end{pmatrix}. \tag{1}$$

When we vary all coefficients so that $\alpha_0 + \alpha_1 + \alpha_2 + \alpha_3 = 1$, and they all remain positive, $\mathbf{v}$ sweeps the tetrahedron defined by the convex hull of v_i, $i = 0, \ldots, 3$. The application of a projective transformation, say $\mathbf{M}$, to $\mathbf{v}$ leads to:

$$\begin{aligned}\mathbf{v}' &= \mathbf{M}\mathbf{v}\\ &= \mathbf{M}\,(\mathbf{v}_0\mathbf{v}_1\mathbf{v}_2\mathbf{v}_3)\begin{pmatrix}\alpha_0\\ \alpha_1\\ \alpha_2\\ \alpha_3\end{pmatrix} = \left(\mathbf{v}'_0\mathbf{v}'_1\mathbf{v}'_2\mathbf{v}'_3\right)\begin{pmatrix}\alpha_0\\ \alpha_1\\ \alpha_2\\ \alpha_3\end{pmatrix}.\end{aligned} \tag{2}$$

It can be checked that when the weights of $\mathbf{v}'_i$, $i = 0, \ldots, 3$, have the same sign (either positive or negative), the region swept by v' is an ordinary tetrahedron. When weights have different signs, or some of the weights are zero, v' does not sweep the convex region defined by v'_i, $i = 0, \ldots, 3$. When this happens, it is said that we have a homogeneous, or external, tetrahedron (Niizeki and Yamaguchi 1994). Figure 1 shows the resulting homogeneous tetrahedra for all possible combinations of weight signs. Clearly, the projective transformation of a homogeneous tetrahedron produces another homogeneous tetrahedron and one can always obtain an arbitrary homogeneous tetrahedron by applying a projective transformation to an ordinary tetrahedron. As a consequence, after a projective transformation is applied on a bounded convex region, the result may not be convex, bounded or singly connectedin Euclidean space.

The 4×4 determinant $| \mathbf{v}_0\mathbf{v}_1\mathbf{v}_2\mathbf{v}_3 |$, after normalizing all weights to 1, is called the oriented volume of the tetrahedron defined by v_0, v_1, v_2, and v_3. This volume is positive if and only if the vectors $\overrightarrow{v_0v_1}$, $\overrightarrow{v_0v_2}$, and $\overrightarrow{v_0v_3}$ define a right-handed coordinate system and negative otherwise. Since, according to Eq. (2), $(\mathbf{v}'_0\mathbf{v}'_1\mathbf{v}'_2\mathbf{v}'_3) = \mathbf{M}(\mathbf{v}_0\mathbf{v}_1\mathbf{v}_2\mathbf{v}_3)$, then

$$| \mathbf{v}'_0\mathbf{v}'_1\mathbf{v}'_2\mathbf{v}'_3 | = \det(\mathbf{M}) \; | \mathbf{v}_0\mathbf{v}_1\mathbf{v}_2\mathbf{v}_3 | \; . \tag{3}$$

This relation will be useful later.

2.2 Topological relations in polyhedra

If f_k stands for face k of a given polyhedron, ∂f_k will denote the set of edges around the face. If e_j represents edge j, ∂e_j will denote the vertices bounding this edge. In both cases ∂ is called the boundary operator. The coboundary operator is the dual operator and it will be denoted by δ. The coboundary of a vertex is a set of edges incident at this vertex, and the coboundary of an edge is its adjacent faces.

Consider an edge common to two adjacent faces. The orientation of each of these faces determines an order for the two end points (vertices) of the edge.

Assuming that the edges of our polyhedra are oriented, an order relationship between the vertices of its boundary and the faces of its coboundary can be established such that $\partial e_j = \{\partial^- e_j, \partial^+ e_j\}$ and $\delta e_j = \{\delta^- e_j, \delta^+ e_j\}$, where ∂^- and ∂^+ denote the halfboundary operators and δ^- and δ^+ the halfcoboundary operators. The adopted convention to choose halfboundaries and halfcoboundaries, depending on the orientation of the edges, is irrelevant to our purposes, as will become clear in Sect. 3.

2.3 Two basic predicates

There are two basic contacts between the elements (faces, edges, and vertices) of two polyhedra in 3D Euclidean space; namely: (a) a face of one polyhedron is in contact with a vertex of the other polyhedron; and (b) an edge of one polyhedron is in contact with an edge of the other polyhedron. These contacts are said to be basic because all other contacts can be expressed as a combination of them (Canny 1987). We next introduce two functions associated with these two incident relations which should be equal to zero for the incidences to occur.

According to Fig. 2a, a type-A function is defined as:

$$A_{v_i,f_j} = |\mathbf{v}_n \; \mathbf{v}_l \; \mathbf{v}_k \; \mathbf{v}_i| \tag{4}$$

where $\{\mathbf{v}_n, \; \mathbf{v}_l, \; \mathbf{v}_k\}$ is an ordered arbitrary representative set of vertices of face f_j. If vertex v_i meets the plane supporting face f_j, then $A_{v_i,f_j} = 0$.

Likewise, according to Fig. 2b, we define a type-B function as:

$$B_{e_i,e_j} = |\mathbf{v}_l \; \mathbf{v}_k \; \mathbf{v}_m \; \mathbf{v}_n| \tag{5}$$

where $\mathbf{v}_k = \partial^+ e_i$, $\mathbf{v}_m = \partial^- e_j$, $\mathbf{v}_l = \partial^- e_i$, and $\mathbf{v}_n = \partial^+ e_j$. According to this definition, if the line supporting edge e_i meets the line supporting edge e_j, then $B_{e_i,e_j} = 0$.

The above determinant-based functions are projective relations which indicate that a point and a plane are incident or two lines are incident. They contain no metrical information relating distance or angle and thus they are clearly independent from any metric. Due to this fact, after applying a projective transformation $\mathbf{M}$ to our polyhedra, these functions can be expressed in terms of their original values using Eq. (3) as follows:

$$A_{v'_i,f'_j} = \det(\mathbf{M}) \cdot A_{v_i,f_j} \tag{6}$$

and

$$B_{e'_i,e'_j} = \det(\mathbf{M}) \cdot B_{e_i,e_j}, \tag{7}$$

respectively.

For the moment, we will assume that the above determinant-based functions never vanish. In other words, we are assuming that no degenerate situations arise (we are assuming, for example, that both polyhedra are not just touching). Under this assumption,

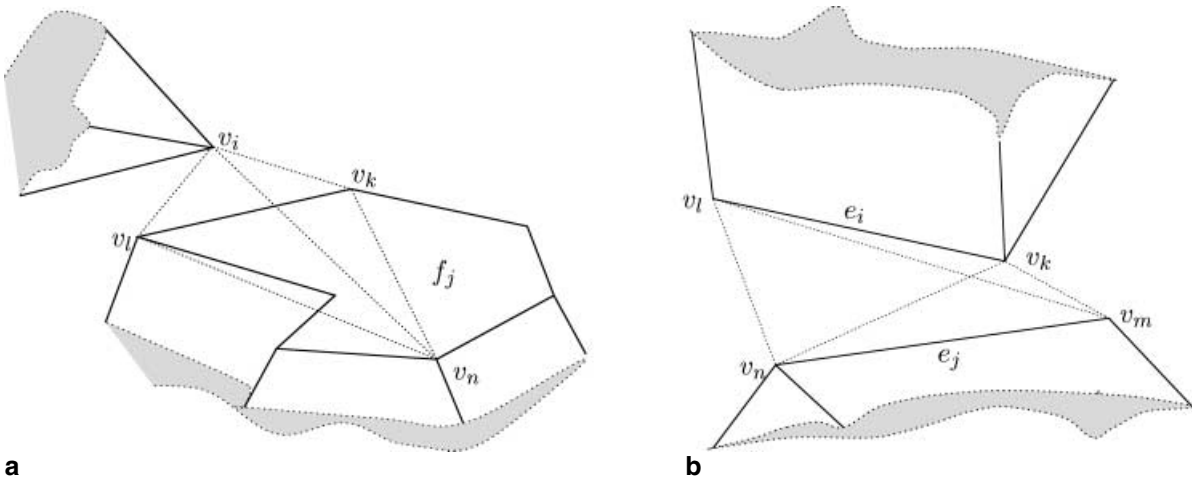

Fig. 2. The two basic contacts between polyhedra and their associated tetrahedra in *dotted lines*

we define the predicate $\mathbf{A}_{v_i,f_j}$, associated with function A_{v_i,f_j}, which is true when $A_{v_i,f_j} > 0$ and false otherwise. Likewise, we define the predicate $\mathbf{B}_{e_i,e_j}$, associated with function B_{e_i,e_j}, which is true when $B_{e_i,e_j} > 0$ and false otherwise.

3 The intersection test

Two polyhedra intersect if, and only if, one of the two following situations arises: (1) an edge of one polyhedron pierces a face of the other polyhedron; or (2) a vertex of one polyhedron is inside the other polyhedron. The second situation must be tested to detect interference when one polyhedron is enclosed entirely within the other one. A necessary and sufficient condition for the first situation to occur is first obtained.

For an edge e_0 to intersect a face f_1 it is necessary that its two endpoints lie on different sides of the plane Π_1 supporting face f_1, i.e.,

$$\mathbf{A}_{\partial^+ e_0,f_1} \otimes \mathbf{A}_{\partial^- e_0,f_1} \tag{8}$$

is true, $\otimes$ being the exclusive or operator (XOR, for short) defined as $(a \otimes b) = (a \wedge \overline{b}) \vee (\overline{a} \wedge b)$.

Assuming that the Boolean formula (8) is true, let us refer to Fig. 3 for further discussion. Let Π_0 be a plane containing edge e_0 which, for convenience, we will assume to be the plane supporting a face f_0, such that $f_0 \in \delta e_0$. The line supporting e_0 divides Π_0 into two half planes, Π_0^- and Π_0^+. Now the number of edges of the face piercing each of these half planes (which determines whether the edge intersects the face) can be obtained as follows. Let $\mathcal{E}^+$ be the set of edges piercing plane Π_0 and pointing *upwards* and $\mathcal{E}^-$, those piercing the same plane and pointing *downwards*, so that

$$\mathcal{E} = \mathcal{E}^+ \cup \mathcal{E}^- = \left\{ e_j \in \partial f_1 \mid \mathbf{A}_{\partial^+ e_j,f_0} \otimes \mathbf{A}_{\partial^- e_j,f_0} \right\}. \tag{9}$$

Then, the function B_{e_0,e_j}, $e_j \in \mathcal{E}^+$, is positive iff the intersection of e_j and Π_0 is located on Π_0^-, and negative iff it is located on Π_0^+. Likewise, the function $B_{e_0,-e_j}$, $e_j \in \mathcal{E}^-$, is positive iff the intersection of e_j and Π_0 is located on Π_0^-, and negative iff it is located on Π_0^+.

Thus, the number of edges piercing half plane Π_0^+ (or half plane Π_0^-) is odd iff

$$\bigotimes_{e_k \in \mathcal{E}} \left[\left(\mathbf{A}_{\partial^+ e_k,f_0} \wedge \mathbf{B}_{e_0,e_k} \right) \vee \left(\mathbf{A}_{\partial^- e_k,f_0} \wedge \mathbf{B}_{e_0,-e_k} \right) \right] \tag{10}$$

is true, where $\bigotimes_{i=1,\ldots,n} a_i = a_1 \otimes a_2 \otimes \cdots \otimes a_n$. Note that $\otimes$ is associative.

According to the definition of the $\otimes$ operator, expression (10) can be rewritten as:

$$\bigotimes_{e_k \in \mathcal{E}} \left(\mathbf{A}_{\partial^- e_k,f_0} \otimes \mathbf{B}_{e_0,e_k} \right). \tag{11}$$

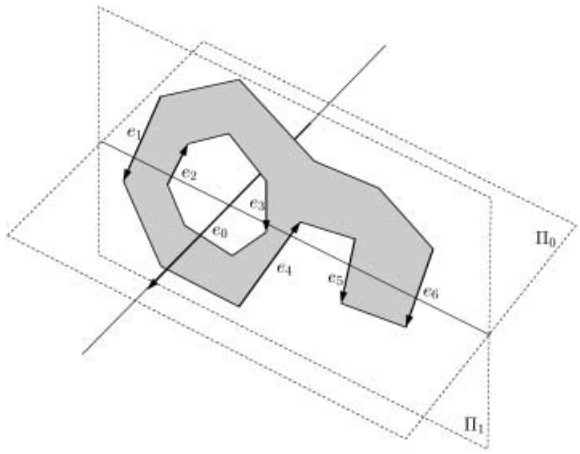

Fig. 3. Detecting whether edge e_0 intersects the shaded face f_1. In this case $\mathcal{E}^+ = \{e_2, e_4\}$ and $\mathcal{E}^- = \{e_1, e_3, e_5, e_6\}$

Then, using Eq. (9), the $\bigotimes$ operation can be extended from $\mathcal{E}$ to the whole ∂f_1:

$$\bigotimes_{e_j \in \partial f_1} \left(\mathbf{A}_{\partial^+ e_j, f_0} \otimes \mathbf{A}_{\partial^- e_j, f_0}\right) \wedge \left(\mathbf{A}_{\partial^- e_j, f_0} \otimes \mathbf{B}_{e_0, e_j}\right). \quad (12)$$

Summarizing, the conjunction of Eqs. (8) and (12) leads to a necessary and sufficient condition for edge e_0 to intersect face f_1, which can be expressed as:

$$(\mathbf{A}_{\partial^+ e_0, f_1} \otimes \mathbf{A}_{\partial^- e_0, f_1}) \wedge \left[\bigotimes_{e_j \in \partial f_1} \left(\mathbf{A}_{\partial^+ e_j, f_0} \otimes \mathbf{A}_{\partial^- e_j, f_0}\right) \wedge \left(\mathbf{A}_{\partial^- e_j, f_0} \otimes \mathbf{B}_{e_0, e_j}\right)\right]. \quad (13)$$

A necessary and sufficient condition to detect whether an edge of one polyhedron is piercing a face of the other polyhedron can easily be obtained by iterating the application of Boolean formula (13) for all edges of one polyhedron and all faces of the other, and vice versa.

Detecting whether a vertex of one polyhedron is inside the other polyhedron (situation 2 at the beginning of this section) can be reduced to the problem of checking whether the number of faces pierced by an edge determined by the considered vertex and a point far enough from the polyhedra is odd or even. Thus the treatment is exactly the same as for the first situation. The predicate that becomes true when vertex v_0 is inside the polyhedron P is then:

$$\bigotimes_{f_i \in P} \left[(\mathbf{A}_{v_0, f_i} \otimes \mathbf{A}_{v_\infty, f_i}) \wedge \bigotimes_{e_j \in \partial f_i} \left(\mathbf{A}_{\partial^+ e_j, f_0} \otimes \mathbf{A}_{\partial^- e_j, f_0}\right) \wedge \left(\mathbf{A}_{\partial^- e_j, f_0} \otimes \mathbf{B}_{e_0, e_j}\right) \right] \quad (14)$$

where v_∞ is a point far from both polyhedra, e_0 is the edge with endpoints v_0 and v_∞, and f_0 is any face containing edge e_0. Note that $\mathbf{v}_\infty$ can be simply obtained by setting the weight of $\mathbf{v}_0$ to 0.

Now notice that, in both Eqs. (13) and (14), all basic predicates are always combined through XOR operators. This is the key point in proving that the test is projectively invariant. Since $(a \otimes b) = (\overline{a} \otimes \overline{b})$, the outcome of the test is exactly the same if the truth values of all basic predicates are simultaneously changed. This is all the change that applying a nonsingular projective transformation $\mathbf{M}$ may bring about, as follows from Eqs. (6) and (7), since $\det(\mathbf{M})$ will multiply all basic functions, possibly changing the signs of all basic predicates at once. Therefore, the outcome of the Boolean test remains invariant after applying any nonsingular projective transformation.

Finally, it is worth mentioning that we have not introduced any constraint on the well-formedness (in terms of connectivity, orientability, and non-self-intersection) of our polyhedra. Actually, the presented test can be applied to self-intersecting polyhedra.

4 Dealing with degeneracies

The correctness of the presented parity-count method for detecting intersections between polyhedra is impaired by degeneracies induced by the problem and the algorithm themselves. Algorithm-induced degeneracies can be avoided. For example, those in which the chosen plane Π_0 contains a vertex of the face against which it is tested can be avoided by selecting a different plane. Nevertheless, handling problem-induced degeneracies would require numerical operations other than 4×4 determinants, which would obscure the initial simplicity of our interference test.

An alternative to handling degeneracies is to remove them by displacing the involved vertices in a consis-

tent manner. Since we have an algorithm that correctly decides the intersection of two polyhedra for almost all inputs, the idea is to redefine the problem that it is supposed to solve to make it work for all inputs. This certainly seems like a dubious way of proceeding, but by elaborating on this idea it is possible to come up with a simple way to resolve degeneracies.

Displacing vertices a finite amount would result in a distortion of our polyhedra. Such a displacement may render faces nonplanar and might be sensitive to numerical imprecisions and round-off errors. To circumvent these shortcomings, we may instead apply an infinitesimal perturbation that will change the original input instance into a nondegenerate one arbitrarily close to it in the Euclidean metric. This is a usual way of dealing with degeneracies in geometric computations (Seidel 1998; Edelsbrunner and Mucke 1990). Adopting the deterministic approach described in (Emiris and Canny 1992), we perturb every point coordinate $v_{i,j}$ to obtain $v_{i,j}(\varepsilon)$ – where ε is an infinitesimal symbolic variable – as follows:

$$v_{i,j}(\varepsilon) = v_{i,j} + \varepsilon(i^j). \tag{15}$$

Let us assume that at some point of our algorithm we get a vanishing determinant, say

$$\Lambda = \left|\mathbf{v}_{i_1}\mathbf{v}_{i_2}\mathbf{v}_{i_3}\mathbf{v}_{i_4}\right| = \begin{vmatrix} v_{i_1,1} & v_{i_1,2} & v_{i_1,3} & 1 \\ v_{i_2,1} & v_{i_2,2} & v_{i_2,3} & 1 \\ v_{i_3,1} & v_{i_3,2} & v_{i_3,3} & 1 \\ v_{i_4,1} & v_{i_4,2} & v_{i_4,3} & 1 \end{vmatrix}. \tag{16}$$

By perturbing the involved point coordinates, according to Eq. (15), we get

$$\Lambda(\varepsilon) = \begin{vmatrix} v_{i_1,1}+\varepsilon i_1 & v_{i_1,2}+\varepsilon i_1^2 & v_{i_1,3}+\varepsilon i_1^3 & 1 \\ v_{i_2,1}+\varepsilon i_2 & v_{i_2,2}+\varepsilon i_2^2 & v_{i_2,3}+\varepsilon i_2^3 & 1 \\ v_{i_3,1}+\varepsilon i_3 & v_{i_3,2}+\varepsilon i_3^2 & v_{i_3,3}+\varepsilon i_3^3 & 1 \\ v_{i_4,1}+\varepsilon i_4 & v_{i_4,2}+\varepsilon i_4^2 & v_{i_4,3}+\varepsilon i_4^3 & 1 \end{vmatrix}. \tag{17}$$

Now, let us define the determinant of a homogeneous perturbation as

$$\Psi = \begin{vmatrix} i_1 & i_1^2 & i_1^3 & 1 \\ i_2 & i_2^2 & i_2^3 & 1 \\ i_3 & i_3^2 & i_3^3 & 1 \\ i_4 & i_4^2 & i_4^3 & 1 \end{vmatrix}. \tag{18}$$

Then, it can be easily checked that

$$\begin{aligned} \Lambda(\varepsilon) = \Lambda + \varepsilon & \left(\begin{vmatrix} i_1 & v_{i_1,2} & v_{i_1,3} & 1 \\ i_2 & v_{i_2,2} & v_{i_2,3} & 1 \\ i_3 & v_{i_3,2} & v_{i_3,3} & 1 \\ i_4 & v_{i_4,2} & v_{i_4,3} & 1 \end{vmatrix} + \begin{vmatrix} v_{i_1,1} & i_1^2 & v_{i_1,3} & 1 \\ v_{i_2,1} & i_2^2 & v_{i_2,3} & 1 \\ v_{i_3,1} & i_3^2 & v_{i_3,3} & 1 \\ v_{i_4,1} & i_4^2 & v_{i_4,3} & 1 \end{vmatrix} \right. \\ & \left. + \begin{vmatrix} v_{i_1,1} & v_{i_1,2} & i_1^3 & 1 \\ v_{i_2,1} & v_{i_2,2} & i_2^3 & 1 \\ v_{i_3,1} & v_{i_3,2} & i_3^3 & 1 \\ v_{i_4,1} & v_{i_4,2} & i_4^3 & 1 \end{vmatrix} \right) \\ + \varepsilon^2 & \left(\begin{vmatrix} v_{i_1,1} & i_1^2 & i_1^3 & 1 \\ v_{i_2,1} & i_2^2 & i_2^3 & 1 \\ v_{i_3,1} & i_3^2 & i_3^3 & 1 \\ v_{i_4,1} & i_4^2 & i_4^3 & 1 \end{vmatrix} + \begin{vmatrix} i_1 & v_{i_1,2} & i_1^3 & 1 \\ i_2 & v_{i_2,2} & i_2^3 & 1 \\ i_3 & v_{i_3,2} & i_3^3 & 1 \\ i_4 & v_{i_4,2} & i_4^3 & 1 \end{vmatrix} \right. \\ & \left. + \begin{vmatrix} i_1 & i_1^2 & v_{i_1,3} & 1 \\ i_2 & i_2^2 & v_{i_2,3} & 1 \\ i_3 & i_3^2 & v_{i_3,3} & 1 \\ i_4 & i_4^2 & v_{i_4,3} & 1 \end{vmatrix} \right) + \varepsilon^3\Psi. \end{aligned} \tag{19}$$

Assuming that ε is an arbitrarily small positive number, the obvious way to obtain a sign for $\Lambda(\varepsilon)$ is to evaluate the terms of the ε expansion Eq. (19) in order of increasing powers of ε. The process stops at the first nonvanishing term and reports its sign. It can be checked that the needed term in expansion Eq. (19) depends on the level of degeneracy: the linear term is needed when the four points lie on a plane, the quadratic term when they lie on a line, and the cubic one when they are all coincident. Clearly, the adopted infinitesimal perturbation does not affect the output for inputs without degeneracies and, since Ψ is always different from zero – because it is a Vandermonde determinant– the proposed procedure always ends up with a sign in case of degeneracy.

Now, we can redefine functions $A_{v_i,f}$ and B_{e_i,e_j} (Eqs. 4 and 5) according to Eq. (16) so that the basic predicates $\mathbf{A}_{v_i,f}$ and $\mathbf{B}_{e_i,e_j}$ – and hence Boolean formulas (13) and (14) – will now be fully valid in the presence of degeneracies.

Let us end this section with a comment on the outcome of our test for polyhedra in contact. It must be clear by now that, because of the way degeneracies are dealt with, the outcome may be arbitrarily that the polyhedra are intersecting or separated. There is

no third possible answer, because the test is binary. However, it should be pointed out that perturbations of the same vertices to resolve different degeneracies will always be consistent, due to the deterministic nature of the perturbation.

5 Conclusions

We have presented a projectively invariant intersection test for polyhedra where the only numerical part is the computation of 4×4 determinants, including the resolution of degeneracies. Since no auxiliary geometric entities need to be computed at any point, it can be summarized as a Boolean formula, instead of a procedure. Actually, it can be seen as a generalization of the parity-count method for the point-polygon test (Haines 1994).

Algorithmic efficiency and implementation issues have deliberately been left out of the discussion, but they deserve some final comments.

Assuming that both the number of vertices and the number of faces of the polyhedra to be checked for intersection are approximately half the number of edges, it can be easily checked that the brute force implementation of the presented intersection test would require the computation of $21mn$ determinants in the worst case, where m and n are the number of edges in the two polyhedra. Since many determinants share the same operations, this fact could be taken into account to obtain a tight complexity bound. Common operations can be detected by reducing determinants to cross and dot products as in Thomas and Torras (1994). Then, the dominating quadratic term for multiplications can be shown to be $4.5mn$ and that for additions $8.5mn$. Nevertheless, the associated coding complexity makes this option impracticable, so that the combination of the brute force algorithm and the technique described in Bronnimann and Yvinec (2000) for the computation of determinants, or its direct implementation in hardware, remain the best choices.

Interference detection libraries can be thought of as consisting of two main ingredients: a basic intersection test and strategies to confine the application of this test to the relevant objects and object parts. This paper is devoted to the former, but we would like to mention that the latter can be used to reduce the number of edge–face pairs to be tested and, therefore, the resulting computational cost. Strategies such as enclosing boxes, space partitioning, hierarchies of bounding volumes, spatial coherence, and orientation bounding are reviewed by Lin and Gottschalk (1998) and Jiménez et al. (2001).

A common problem in intersection detection algorithms is their coding complexity: much effort is devoted to handling degenerate situations correctly. This leads to a lack of robustness. Our algorithm is simple because degenerate situations are handled naturally using the same formalism. This simplicity and homogeneity leads to quick implementations.

Finally, we conjecture that deciding whether two polytopes of arbitrary dimensions intersect can be carried out by analyzing determinant sign sets of their homogeneous vertex coordinates in the smallest space containing both polytopes. This is clearly a point that deserves further research.

References

1. Antonio F (1992) Faster line segment intersection. In: Kirk D (ed) Graphics Gems III. Academic Press, Boston, pp 199–202
2. Bloomenthal J, Rokne J (1994) Homogeneous coordinates. Visual Comput 10:176–187
3. Boyse JW (1979) Interference detection among solids and surfaces. Commun ACM 21:3–9
4. Bronnimann H, Yvinec M (2000) Efficient exact evaluation of signs of determinants. Algorithmica 27:21–52
5. Canny JF (1987) The complexity of robot motion planning. MIT Press, Cambridge, Mass.
6. Chubarev A (1999) Robust set operations on polyhedral solids: a fixed precision approach. Int J Comput Geom Appl 6:187–204
7. Edelsbrunner H, Mucke EP (1990) Simulation of simplicity: a technique to cope with degenerate cases in geometric algorithms. ACM Trans Graph 9:66–104
8. Emiris IZ, Canny JF (1992) An efficient approach to removing geometric degeneracies. In: Proc. 8th ACM Symp. on Computational Geometry, pp 74–82
9. Fortune S (1997) Polyhedral modelling with multiprecision integer arithmetic. Comput Aided Des 29:123–133
10. Haines E (1994) Point in polygon strategies. In: Heckbert P (ed) Graphics Gems IV. Academic Press, Boston, pp 24–46
11. Held M (1997) ERIT. A collection of efficient and reliable intersection tests. J Graph Tools 2:25–44
12. Hoffmann CM, Hopcroft JE, Karasick MS (1989) Robust set operations on polyhedral solids. IEEE Comput Graph Appl 9:50–59
13. Horn WP, Taylor DL (1989) A theorem to determine the spatial containment of a point in a planar polygon. Comput Vis Graph Image Process 45:106–116
14. Jiménez P, Thomas F, Torras C (2001) Collision detection: a survey. Comput Graph 25:269–285
15. Kalay YE (1982) Determining the spatial containment of a point in general polyhedra. Comput Graph Image Process 19:303–334

16. Lane J, Megedson B, Rarick M (1984) An efficient point in polyhedron algorithm. Comput Vis Graph Image Process 26:118–125
17. Lin MC, Gottschalk S (1998) Collision detection between geometric models: a survey. In:
18. Möller T (1997) A fast triangle–triangle intersection test. J Graph Tools 2:25–30
19. Mount DM (1997) Geometric intersection. In: Goodman JE, O'Rourke J (eds) Handbook of Discrete and Computational Geometry. CRC Press, New York, pp 615–630
20. Niizeki M, Yamaguchi F (1994) Projectively invariant intersection detections for solid modelling. ACM Trans Graph 13:277–299
21. O'Rouke J (1998) Computational Geometry in C, 2nd edn. Cambridge University Press, Cambridge
22. Paeth AW (1990) A fast 2D point-on-line test. In: Glassner AS (ed) Graphics Gems I. Academic Press, Boston, pp 49–50
23. Pinto-Carvalho PC, Roma-Cavalcanti P (1994) Point in polyhedron testing using spherical polygons. In: Heckbert PS (ed) Graphics Gems IV. Academic Press, Boston, pp 42–49
24. Prasad M (1991) Intersection of line segments. In: Arvo J (ed) Graphics Gems II. Academic Press, Boston, pp 7–8
25. Requicha AAG, Voelcker HB (1982) Solid modeling: a historical summary and contemporary assessment. IEEE Comput Graph Appl 2:9–24
26. Seidel R (1998) The nature and meaning of perturbations in geometric computing. Discrete Comput Geom 19:1–17
27. Thomas SW (1991) Decomposing a matrix into simple transformations. In: Arvo J (ed) Graphics Gems II. Academic Press, Boston, pp 320–323
28. Thomas F, Torras C (1994) Interference detection between non-convex polyhedra revisited with a practical aim. In: IEEE Int. Conf. on Robotics and Automation, pp 587–594
29. Yamaguchi F (1988) Applications of the 4×4 determinant method and the polygon engine. Visual Comput 4:176–187
30. Yamaguchi F (1998) A shift of playground for geometric processing from Euclidean to homogeneous. Visual Comput 14:315–327
31. Yamaguchi F, Niizeki M (1997) Some basic geometric test conditions in terms of Plücker coordinates and Plücker coefficients. Visual Comput 13:29–41

FEDERICO THOMAS was born in Barcelona, on 5 October 1961. He received a B.Sc. degree in telecommunications engineering in 1984 and a Ph.D. degree (with honors) in computer science in 1988, both from the Polytechnic University of Catalonia. Since March 1990, he has been a research scientist at the Institut de Robòtica i Informàtica Industrial of the Spanish High Council for Scientific Research. His research interests are in geometry and kinematics, with applications to robotics, computer graphics, and machine vision. He has published more than 40 research papers in international journals and conference proceedings (http://www-iri.upc.es/people/thomas).

CARME TORRAS is research professor at the Institut de Robòtica i Informàtica Industrial (CSIC-UPC). She received M.Sc. degrees in mathematics and computer science from the Universitat de Barcelona and the University of Massachusetts, respectively, and a Ph.D. degree in computer science from the Universitat Politècnica de Catalunya. Prof. Torras has published three books and more than 100 papers in the areas of robotics, vision, and neurocomputing. She has been local project leader of several European projects, such as "Planning RObot Motion" (PROMotion), "Behavioural Learning: Sensing and Acting" (B-LEARN), "Robot Control based on Neural Network Systems" (CONNY), and "Self-organization and Analogical Modelling using Subsymbolic Computing" (SUBSYM) (http://www-iri.upc.es/people/torras).