



## Signed distance computation using the angle weighted pseudonormal

**Bærentzen, Jakob Andreas; Aanæs, Henrik**

*Published in:*

IEEE Transactions on Visualization and Computer Graphics

*Link to article, DOI:*

[10.1109/TVCG.2005.49](https://doi.org/10.1109/TVCG.2005.49)

*Publication date:*

2005

*Document Version*

Publisher's PDF, also known as Version of record

[Link back to DTU Orbit](#)

*Citation (APA):*

Bærentzen, J. A., & Aanæs, H. (2005). Signed distance computation using the angle weighted pseudonormal. / *IEEE Transactions on Visualization and Computer Graphics*, 11(3), 243 - 253.  
<https://doi.org/10.1109/TVCG.2005.49>

---

### General rights

Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

- Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
- You may not further distribute the material or use it for any profit-making activity or commercial gain
- You may freely distribute the URL identifying the publication in the public portal

If you believe that this document breaches copyright please contact us providing details, and we will remove access to the work immediately and investigate your claim.

# Signed Distance Computation Using the Angle Weighted Pseudonormal

J. Andreas Bærentzen and Henrik Aanæs, *Member, IEEE*

**Abstract**—The normals of closed, smooth surfaces have long been used to determine whether a point is inside or outside such a surface. It is tempting to also use this method for polyhedra represented as triangle meshes. Unfortunately, this is not possible since, at the vertices and edges of a triangle mesh, the surface is not  $C^1$  continuous, hence, the normal is undefined at these loci. In this paper, we undertake to show that the angle weighted pseudonormal (originally proposed by Thürmer and Wüthrich and independently by Séquin) has the important property that it allows us to discriminate between points that are inside and points that are outside a mesh, regardless of whether a mesh vertex, edge, or face is the closest feature. This inside-outside information is usually represented as the sign in the signed distance to the mesh. In effect, our result shows that this sign can be computed as an integral part of the distance computation. Moreover, it provides an additional argument in favor of the angle weighted pseudonormals being the natural extension of the face normals. Apart from the theoretical results, we also propose a simple and efficient algorithm for computing the signed distance to a closed  $C^0$  mesh. Experiments indicate that the sign computation overhead when running this algorithm is almost negligible.

**Index Terms**—Mesh, signed distance field, normal, pseudonormal, polyhedron.

## 1 INTRODUCTION

BY far the most popular way of representing 3D objects in computers is as triangle meshes. Often, these triangle meshes are closed, thus representing polyhedra. When dealing with such 3D objects, it is often crucial to be able to determine how they relate to each other. In particular, we may want to know whether two objects interpenetrate or whether a given path collides with an object. A fundamental prerequisite for such queries is the ability to determine whether a given point in space is inside or outside a 3D object—here assumed to be a triangle mesh.

For objects with closed, smooth, and orientable surfaces, the surface normal is an important tool for determining whether a given point is inside or not. This is done by finding the closest point,  $c$ , on the surface and taking the inner product of the surface normal at  $c$  with the vector between the given point  $p$  and  $c$ , i.e.,  $r = p - c$ . However, an object represented as a polyhedron or mesh is not a smooth surface and, hence, does not have normals defined everywhere on the surface (i.e., the surface is discontinuous at edges and vertices). At the outset, the above simple scheme is therefore not deemed possible.

It is, however, possible to define vectors at edges and vertices which possess some of the properties of normals and these we denote *pseudonormals*. Naturally, there are a variety of these pseudonormals capturing different properties of normals. In the case of determining whether a point is inside a polyhedron, it turns out that the *angle weighted pseudonormal* proposed by Thürmer and Wüthrich [1] and independently

by Séquin [2] captures the necessary properties. Thus, the above-mentioned simple scheme can be used for polyhedra as well. To our knowledge, this has not been observed before.

The main contribution of this paper is proposing and proving that the angle weighted pseudonormal can be used for determining whether a point is inside a polyhedron. Specifically, we show that the sign of the inner product between the angle weighted pseudonormal and the vector to an arbitrary point from its closest point on the surface uniquely determines whether that given point is inside the polyhedron or not. In addition, we argue that this is a rather unique property of the angle weighted pseudonormal in that all other proposed analytic pseudonormals which we have been able to find do not possess this property. Hence, valuable insight into the geometry of the widely used polyhedron or mesh-based object structures is provided. Finally, we argue that the proposed procedure for determining whether a point is inside or outside is also numerically robust.

In order to demonstrate the applicability of this result, it is used to propose a scheme for signed distance to triangle mesh computation. In essence, this scheme consists of extending any *unsigned* distance algorithm to calculate the *signed* distance as well. Fortunately, all *unsigned* distance algorithms find the closest point on the mesh,  $c$ , to all considered points,  $p$ . In conjunction with the angle weighted pseudonormal, this is enough information to compute the sign.

We propose an efficient algorithm for computing the signed distance to a mesh from a point in a general position. The algorithm is similar in structure to other algorithms for computing the unsigned distance [3], [4]. The algorithm requires a hierarchical representation of the mesh and our results show that, once this hierarchy has been built, the

• The authors are with the Informatics and Mathematical Modeling Department, Technical University of Denmark, DK-2800 Kongens Lyngby, Denmark. E-mail: {jab, haa}@imm.dtu.dk.

Manuscript received 1 Oct. 2004; revised 16 Dec. 2004; accepted 5 Jan. 2005; published online 10 Mar. 2005.

For information on obtaining reprints of this article, please send e-mail to: [tcvg@computer.org](mailto:tcvg@computer.org), and reference IEEECS Log Number TVCG-0121-1004.

overhead incurred by computing the sign is negligible compared to the distance computation. The performance of our method is measured by generating 3D grids of signed distances for a number of models and we compare the performance to a method for computing only the sign.

Signed distances are important for a number of applications. For instance, signed distance fields (i.e., grids of voxels holding distances) are often used to initialize the level set method [5], [6] which, in turn, has many applications in, e.g., computer vision and computer graphics [7]. Although beyond the scope of this paper, we envision that the result also has relevance for other applications, most notably path planning, offset surfaces, and collision detection, c.f. [8], [9], all of which can be approached via signed distances or signed distance fields [10].

### 1.1 Overview and Relations to Other Work

The main result of this paper is the proof that the angle weighted pseudonormal can discriminate between points that are inside and points that are outside a triangle mesh or polyhedron.<sup>1</sup> The angle weighted pseudonormal is discussed in detail in Section 3 and the proof is found in Section 3.

Other solutions have been proposed for addressing the point in polyhedron problem. The best known is counting the intersections of a ray going from the given point to infinity since the number of intersections must be odd if the point is inside. A robust way to implement this procedure is due to Linhart [11]. Another scheme based on testing the inclusion of the point in tetrahedra formed between each face and the origin was proposed by Feito and Torres [12]. Finally, a different method based on summing the area of spherical polygons was proposed in [13] by Carvalho and Cavalcanti. For earlier work, see also references in [11] and [12]. In general, the previous methods must visit the entire mesh and their runtime is at best linear in the number of faces. This is certainly true of [12], [13]. Using intersection counting [11], we only need to visit the parts of the mesh that are pierced by a ray going from the given point to infinity. However, our method only needs to find the closest point on the mesh, which can be even faster.

Our primary application for the work in this paper is the generation of discrete signed distance fields from triangle meshes. There is a good deal of literature on this topic and, in the following, we review the most relevant papers.

Payne and Toga [3] discuss many issues pertaining to distance fields. One of these is the generation of distance fields from triangle meshes, optimizations, and the correct computation of signs. Payne and Toga addressed the sign problem by suggesting that the mesh is scan converted separately. Jones [14] also uses scan conversion and computes the distance for voxels only in the vicinity of sign changes. Dachille and Kaufman [15] proposed a method for computing the distance to a triangle using only incremental plane calculations. Their technique is amenable to hardware implementation.

1. In the following, we will use the two terms interchangeably.

Mauch proposed a novel scan conversion method [16]. The idea is to generate a so-called *characteristic* for each mesh feature (i.e., each edge, vertex, and face). A characteristic is essentially the Voronoi neighborhood of the feature. All voxels inside the same characteristic are closest to the same feature. Sigg et al. extended Mauch's method in [17]. In order to make the method sufficiently fast when the final scan conversion is performed on graphics hardware, they reduced the number of characteristics producing only one for each triangle. Another hardware accelerated method for generating distance fields was proposed recently by Sud et al. in [18]. Occlusion queries are used to cull away features that cannot contribute to the distance field and the maximum extent of the contribution of a feature is bounded using observations regarding spatial coherence.

This paper extends unsigned distance algorithms to include sign. If such an algorithm is used to compute signed distance fields, this implies that, unlike in all of the above methods, there is no need for scan conversion [19]. Although scan conversion is not intrinsically difficult, there are some pitfalls mentioned in Section 5. The algorithm for signed distance computation is discussed in Section 4.

Some authors have investigated alternative distance field representations where distances are not stored in a regular voxel grid. For instance, Gibson et al. proposed the use of adaptive distance fields [20]. Huang et al. suggested the notion of *complete distance fields* [21], where information about the initial triangles is kept in a 3D grid. Guéziec [4] used a hierarchical mesh representation to quickly compute distances. His method is especially efficient if the mesh deforms. The work by Guéziec extended earlier work in collision detection by Johnson and Cohen [22] and Larsen et al. [23].

Recently, Wu and Kobbelt pursued a similar strategy [24]. They generate a BSP tree, c.f. [25], where each node contains a linear approximation of the distance field. The distance field is only  $C^{-1}$ , but of bounded error. While a detailed discussion of the methods above is beyond our scope, they are mentioned, in part, because it is likely that they could benefit from the proposed sign computation scheme.

As mentioned, several pseudonormals have been proposed. Probably the earliest is due to Gouraud, who suggested using the (unweighted) sum of face normals as a vertex normal [26]. Thürmer and Wüthrich [1] and Séquin [2] independently proposed the sum of angle weighted face normals and Glassner [27] and Max [28] have proposed yet other ways to compute normals. These techniques are discussed in detail in Section 2.1.

Last, a previous and less complete publication of this work is [29].

## 2 ANGLE WEIGHTED PSEUDONORMAL

First some formalism. Let  $M$  denote a triangle mesh. Assume that  $M$  describes a closed, orientable 2-manifold in

3D Euclidean space, i.e., the signed distance problem is well-defined. Denote by  $\mathcal{M}$  the closure of the interior of  $M$ , i.e.,  $M = \partial\mathcal{M}$ . Define the *unsigned* distance from a point  $\mathbf{p}$  to  $M$  as<sup>2</sup>

$$d(\mathbf{p}, M) = \inf_{\mathbf{x} \in M} \|\mathbf{p} - \mathbf{x}\|, \quad (1)$$

and let the sign be determined by whether  $\mathbf{p}$  is in  $\mathcal{M}$ , positive denoting  $\mathbf{p} \notin \mathcal{M}$ . The optimum or optima in (1) are the *closest point(s)* to  $\mathbf{p}$ . We use  $\mathbf{c}$  to denote a closest point.

Conversely, for a given point  $\mathbf{c}$  on the mesh, there is a set of points  $V(\mathbf{c})$  such that for  $\mathbf{p} \in V(\mathbf{c})$ ,  $\mathbf{c}$  is the surface point closest to  $\mathbf{p}$ . The set  $V(\mathbf{c})$  is the Voronoi region of  $\mathbf{c}$ . For a solid with a *differentiable surface*, the Voronoi regions form line segments from  $\mathbf{c}$  in the positive and negative directions of the normal  $\mathbf{n}$  at  $\mathbf{c}$ . These line segments are either semi-infinite or terminate at points of equal distance to two surface points (i.e., points on the medial axis). Since any  $\mathbf{p} \in V(\mathbf{c})$  lies on a line segment defined by the normal, it is clear that

$$\mathbf{n} \cdot (\mathbf{p} - \mathbf{c}) = \begin{cases} \|\mathbf{p} - \mathbf{c}\| & \text{if } \mathbf{p} \text{ outside surface} \\ -\|\mathbf{p} - \mathbf{c}\| & \text{if } \mathbf{p} \text{ inside surface} \\ 0 & \text{if } \mathbf{p} \text{ on surface.} \end{cases} \quad (2)$$

In the case of piecewise planar surfaces,  $M$ , the Voronoi regions form line segments only if  $\mathbf{c}$  belongs to a face. If  $\mathbf{c}$  belongs to an edge,  $V(\mathbf{c})$  is a wedge delimited by the face normals. In the case of a vertex,  $V(\mathbf{c})$  is a cone whose apex coincides with  $\mathbf{c}$ . The edges of the cone are defined by the face normals.

Unfortunately, this means that (2) no longer holds if the point  $\mathbf{c}$  belongs to an edge or a vertex. Hence, an obvious question is whether we can formulate a pseudonormal for any locus  $\mathbf{c}$  on a mesh such that a modified form of (2) holds. The main contribution of this paper is proving that, for the angle weighted pseudonormal,  $\mathbf{n}_\alpha$ , the following holds:

$$\begin{aligned} \mathbf{n}_\alpha \cdot (\mathbf{p} - \mathbf{c}) &> 0 && \text{if } \mathbf{p} \text{ outside surface} \\ \mathbf{n}_\alpha \cdot (\mathbf{p} - \mathbf{c}) &< 0 && \text{if } \mathbf{p} \text{ inside surface} \\ \mathbf{n}_\alpha \cdot (\mathbf{p} - \mathbf{c}) &= 0 && \text{if } \mathbf{p} \text{ on surface.} \end{aligned} \quad (3)$$

The angle weighted pseudonormal is defined as follows: For a given face, denote the normal<sup>3</sup> as  $\mathbf{n}$ , which is assumed pointing outward, i.e., if the closest point,  $\mathbf{c} \in M$ , to  $\mathbf{p}$  is on a face, then the sign is given by

$$\text{sign}(\mathbf{r} \cdot \mathbf{n}), \quad \mathbf{r} = \mathbf{p} - \mathbf{c}. \quad (4)$$

The angle weighted pseudonormal for a given point,  $\mathbf{x} \in M$ , is then defined as

$$\mathbf{n}_\alpha = \frac{\sum_i \alpha_i \mathbf{n}_i}{\|\sum_i \alpha_i \mathbf{n}_i\|}, \quad (5)$$

where  $i$  runs over the faces incident with  $\mathbf{x}$  and  $\alpha_i$  is the incident angle, c.f. Fig. 1.

Even though Thürmer and Wüthrich [1] only considered this angle weighted pseudonormal for vertices of the mesh, it generalizes nicely to faces and edges. In the face case, there is *one* incident face, namely, the face itself, hence

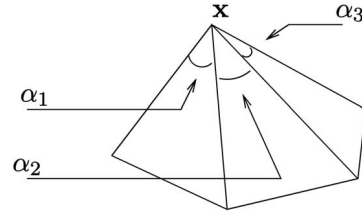


Fig. 1. The incident angles  $\{\alpha_1, \alpha_2, \alpha_3, \dots\}$  of point  $\mathbf{x} \in M$ .

$$\mathbf{n}_1 = \mathbf{n}_\alpha = \frac{2\pi \mathbf{n}_1}{\|2\pi \mathbf{n}_1\|}$$

since the length of the normal is unit. This illustrates that the angle weighted pseudonormal can be seen as a generalization of the standard face normal. On edges, both face normals have weight  $\pi$  and the result is the same as when computing the unweighted average.

## 2.1 Other Pseudonormals

Other pseudonormals have been proposed as extensions to the face normal for meshes, but we are not aware of any other analytic pseudonormal which fulfills (3).

Many other pseudonormals do have closed form expressions, but, unlike the angle weighted pseudonormal, they cannot be used for sign computation in general nor were they proposed for this purpose.

In order to show this, we begin by observing that a point,  $\mathbf{p}$ , which is closest to a vertex,  $\mathbf{x}$ , of a triangle mesh may be outside the mesh but behind a face of the mesh. Thus, in general, a face normal cannot be used as a pseudonormal for our application. By changing the tessellation, examples can be constructed where the pseudonormals discussed below come arbitrarily close to any face normal of any given geometry. Hence, counterexamples can be constructed.

The most obvious pseudonormal, mentioned by Gouraud in [26], is the unweighted mean of normals, i.e.,

$$\frac{\sum_i \mathbf{n}_i}{\|\sum_i \mathbf{n}_i\|}.$$

Glassner proposed a slightly different method [27]: For a given vertex,  $\mathbf{x}$ , we find all neighboring vertices and the plane that best fits this set of points using the method of least squares. Finally, the plane normal is used as the normal at  $\mathbf{x}$ . The method can be modified slightly. Instead of using the neighboring vertices, we intersect all edges incident on  $\mathbf{v}$  with an infinitesimal ball. Now, the points where the edges and the ball intersect are used as the basis for the least squares fit.

None of these pseudonormals can be used for inside outside discrimination since, by a more or less contrived retessellation, it is possible to make them arbitrarily similar to a face normal. This is illustrated for the unweighted average normal in Fig. 2 and for Glassner's modified approach in Fig. 3. A similar counterexample can easily be constructed for Glassner's original method.

Another pseudonormal, proposed by Huang et al. [21], uses the incident face normal with the largest inner product, i.e.,

2. Infimum is the greatest lower bound of a set, denoted  $\inf$ .  
3. It is assumed that the normals have length 1.

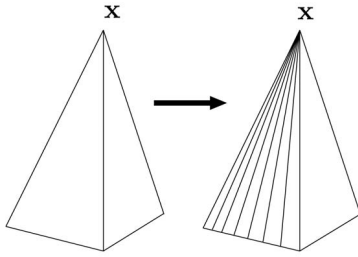


Fig. 2. By subdividing one of the incident faces of  $x$  enough, the unweighted mean normal can come arbitrarily close to the normal of that face. Since just using the normal of an arbitrary incident face clearly does not work, this approach is inapplicable for sign computation in general.

$$\mathbf{n}_m, \quad m = \arg \max_j \|\mathbf{r} \cdot \mathbf{n}_j\|.$$

However, this does not always work. A counterexample is given in Fig. 4.

Max [28] proposes a pseudonormal (for vertices only) consisting of the cross products of all pairs of incident edges (associated with the same face) divided by the squared lengths of these edges, i.e., let  $\mathbf{e}_i$  denote an incident edge, then

$$\mathbf{n}_M = \sum_i \frac{\mathbf{e}_i \times \mathbf{e}_{i+1}}{\|\mathbf{e}_i\|^2 + \|\mathbf{e}_{i+1}\|^2}.$$

It is demonstrated that this pseudonormal produces results that are very close to the analytic normals for a certain class of smooth surfaces. However, this normal is not suited for sign computation. To see this, consider what happens when a surface is retesselated to create a face that is extremely small. In this case, the normal of the small triangle will completely dominate the sum. See Fig. 5.

Another pseudonormal, which we have considered due to its similarity to the angle weighted pseudonormal (and the fact that it is faster to compute), is constructed by summing the cross product of normalized adjacent edges, i.e.,

$$\mathbf{n}_x = \sum_i \frac{\mathbf{e}_i \times \mathbf{e}_{i+1}}{\|\mathbf{e}_i\| \cdot \|\mathbf{e}_{i+1}\|},$$

which corresponds to weighting the face normal with (twice) the area of the triangle. It is seen that this

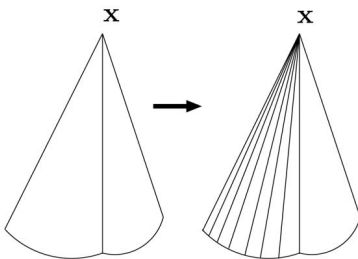


Fig. 3. Glassner's modified approach is based on finding the intersections of an infinitesimal ball and the edges incident on  $x$ . A plane is then fitted to these intersection points. On the left,  $x$  is shown, together with the part of the mesh inside the ball. On the right, we see what happens when a face of the mesh is subdivided. This generates a great number of intersection points lying in the plane of the subdivided face and these points will dominate the least squares computation, thus making the pseudonormal arbitrarily close to an original face normal.

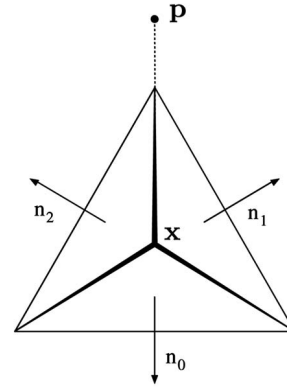


Fig. 4. A counterexample for Huang's normal. In the figure, a pyramid is seen from above. Let  $\mathbf{r} = \mathbf{p} - \mathbf{x}$ , where  $x$  is the apex and  $p$  is a point above the pyramid. It is clear that the inner product  $\mathbf{r} \cdot \mathbf{n}_0$  is greater than both  $\mathbf{r} \cdot \mathbf{n}_1$  and  $\mathbf{r} \cdot \mathbf{n}_2$ . Unfortunately, since  $\mathbf{n}_0$  points away from  $p$ , the point  $p$  is incorrectly classified as being inside the pyramid.

pseudonormal is identical to the angle weighted pseudonormal in the limit as the angles become infinitesimally small, hence the similarity.

The sum of cross products does not give us the desired result. To see this, consider first an almost degenerate scenario: The point  $p$  where we wish to know the sign is located at a point on the positive  $Z$  axis. Our vertex  $x$  (which is the mesh point closest to  $p$ ) is at the origin and there are three adjacent triangles in an almost flat configuration. One of these faces up and the angle at  $x$  is almost  $\pi$ . The two other triangles face down and they make angles close to  $\frac{\pi}{2}$  at  $x$ . The scenario is illustrated in Fig. 6. Since the area of the triangle facing up will be almost 0 and the area of the two triangles facing down will be close to 0.5, the weighted sum of normals will clearly point in the direction of the negative  $Z$  axis. Since  $p$  lies in the direction of the positive  $Z$  axis, the dot product is negative, although we have constructed a case where  $p$  lies outside the mesh.

In the counterexample discussed above, the edges may be unit-length. Therefore, the arguments also constitute a counterexample against the sum of cross products between unnormalized edges.

It is interesting to observe that the counterexamples above are mostly constructed by varying the tessellation

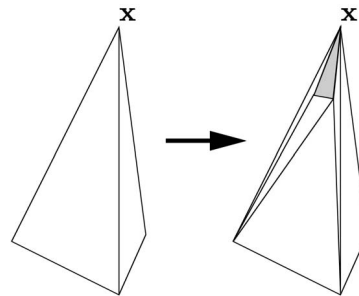


Fig. 5. By a contrived retriangulation, it is possible to change the normal so that it is arbitrarily close to the normal of one of the faces. Here, a very small inset triangle is created (drawn gray). If it is sufficiently small, it will dominate the normal computation, thus making the pseudonormal arbitrarily close to an original face normal.

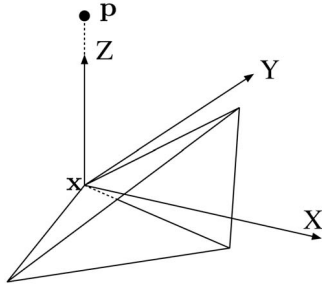


Fig. 6. Again,  $x$  is the point closest to  $p$ , but the three triangles lie almost in the  $XY$  plane and the triangle pointing up has almost collapsed to a line segment.

without changing the geometry. This gives us a clue that the pseudonormal we are looking for should be tessellation invariant, which is a property of the angle weighted pseudonormal.

In the above, we have exclusively concerned ourselves with analytic pseudonormals. If we relax the requirement that the normal must have a closed form expression, it is, indeed, possible to compute a normal with the desired property expressed in (3). Specifically, it is possible to formulate an optimization problem whose solution is a normal which can be used for sign computation. One example is the convex hull normal,  $\mathbf{n}_h$ , which was used in [30], but for a completely different purpose. This procedural normal is computed by considering the *tips* of all the  $\mathbf{n}_i$  (which all lie on the unit sphere) and then computing the convex hull of all these tips. The unnormalized convex hull normal,  $\mathbf{n}_h$ , is then found as the shortest vector from the origin to this convex hull. Finding the closest point to the origin is an optimization problem solved efficiently via Gilbert's algorithm [31]. A proof that  $\mathbf{n}_h$  has the property expressed in (3) is sketched in the Appendix.

### 3 PROOF

Here, it will be proven that the angle weighted pseudonormal can take the role of the face normal in (4), thus generalizing it to all points  $x$  on the mesh,  $M$ . Since we are only interested in the sign, we omit the normalization and only consider

$$\mathbf{N}_\alpha = \sum_i \alpha_i \mathbf{n}_i, \quad (6)$$

easing notation. Also,  $\mathbf{N}_\alpha$  is faster to compute than  $\mathbf{n}_\alpha$ .

**Theorem 1.** *Let there be given a point  $\mathbf{p}$  and assume that  $c$  is a closest point in  $M$  so that*

$$\|\mathbf{c} - \mathbf{p}\| = d, \quad d = \inf_{x \in M} \|\mathbf{p} - \mathbf{x}\|.$$

*Let  $\mathbf{N}_\alpha$  be the sum of normals to faces incident on  $c$ , weighted by the angle of the incident face, i.e.,*

$$\mathbf{N}_\alpha = \sum \mathbf{n}_i \alpha_i. \quad (7)$$

*Finally, consider the vector  $\mathbf{r} = \mathbf{p} - \mathbf{c}$ . It now holds for*

$$D = \mathbf{N}_\alpha \cdot \mathbf{r} \quad (8)$$

*that  $D > 0$  if  $\mathbf{p}$  is outside the mesh.  $D < 0$  if  $\mathbf{p}$  is inside.*

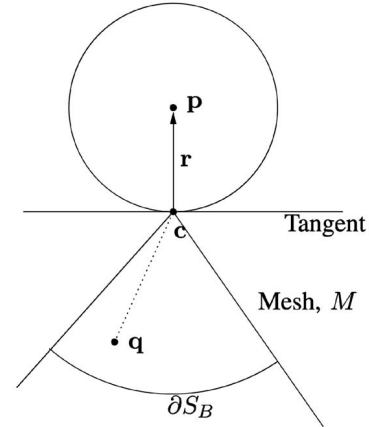


Fig. 7. Illustration of Lemma 1. It is seen that  $\angle(\mathbf{c}\mathbf{q}, \mathbf{c}\mathbf{p}) \geq \pi/2$  since  $c$  is the point in  $M$  closest to  $p$ . Note that  $c$  is not constrained to be a vertex.

To prove this, we first consider the case where  $\mathbf{p}$  is outside the mesh.

Define the volume  $S$  as the intersection of  $M$  and a ball,  $B$ , centered at  $c$ . The radius of  $B$  is chosen arbitrarily to be 1. However,  $B$  may not contain any part of the mesh not incident on  $c$ . If that is the case, we can fix the problem by rescaling the mesh.  $\partial S$  (the boundary of  $S$ ) consists of a part coincident with the mesh,  $\partial S_M$ , and a part coincident with the ball,  $\partial S_B = \partial S - \partial S_M$ . Observe that  $\partial S = \partial S_M \cup \partial S_B$  and  $\partial S_M \cap \partial S_B = \emptyset$ .

Introduce a divergence-free vector field,  $F$ , where, at any point  $\mathbf{q}$ ,  $F(\mathbf{q}) = \mathbf{r}$ . Then, from the theorem of Gauss, we have ( $F$  being divergence free)

$$\begin{aligned} \int_{\partial S} F \cdot \mathbf{n}(\tau) d\tau &= 0 \\ &= \int_{\partial S_M} \mathbf{r} \cdot \mathbf{n}(\tau) d\tau + \int_{\partial S_B} \mathbf{r} \cdot \mathbf{n}(\tau) d\tau. \end{aligned} \quad (9)$$

**Lemma 1.** *For any point  $\mathbf{q} \in S$ , the angle  $\angle(\mathbf{c}\mathbf{q}, \mathbf{c}\mathbf{p})$  is greater than or equal to  $\pi/2$  when  $\mathbf{p} \notin M$ .*

**Proof.** By construction,  $c$  is a star point in  $S$ , i.e., the line segment between  $c$  and any point in  $S$  lies completely in  $S$ . Hence, if there is a  $\mathbf{q}$  such that  $\angle(\mathbf{c}\mathbf{q}, \mathbf{c}\mathbf{p}) < \pi/2$ , there would be a point on the line between  $c$  and  $\mathbf{q}$  which is closer to  $\mathbf{p}$  than  $c$ . This is easily seen because, if

$$\angle(\mathbf{c}\mathbf{q}, \mathbf{c}\mathbf{p}) < \pi/2,$$

the line segment from  $c$  to  $\mathbf{q}$  must pass through the interior of a closed ball of radius  $r$  centered at  $\mathbf{p}$  and any point in the interior of this ball will be closer to  $\mathbf{p}$  than  $c$ . Finally, since  $S \subset M$ , this contradicts our requirement that  $c$  is the point in  $M$  closest to  $\mathbf{p}$ . See Fig. 7.

For all points  $\mathbf{q} \in \partial S_B$ , it is seen that the normal,  $\mathbf{n}(\mathbf{q})$ , is given by  $\mathbf{c}\mathbf{q}$  since  $B$  is the unit sphere centered at  $c$ . So, by Lemma 1,<sup>4</sup>  $\mathbf{n}(\mathbf{q}) \cdot \mathbf{r} \leq 0$  for all the normals,  $\mathbf{n}_B$ , on  $\partial S_B$ . Therefore, we have that

$$\int_{\partial S_B} \mathbf{r} \cdot \mathbf{n}(\tau) d\tau < 0. \quad (10)$$

4. Observe that  $\partial S_B \subset S$ .

The inequality in (10) is strict because the left-hand side is only zero if the area of  $\partial S_B$  is zero and this, in turn, would require the mesh to collapse, breaking our manifold assumption.

From (9) and (10), it now follows that

$$\int_{\partial S_M} \mathbf{r} \cdot \mathbf{n}(\tau) d\tau > 0. \quad (11)$$

It is seen that the intersection of face  $i$  and  $S$  has an area<sup>5</sup> equal to  $\alpha_i$ , implying that the flux of  $F$  through this intersection is given by  $\mathbf{r} \cdot \mathbf{n}_i \alpha_i$ . So,

$$\int_{\partial S_M} \mathbf{r} \cdot \mathbf{n}(\tau) d\tau = \Sigma \mathbf{r} \cdot \mathbf{n}_i \alpha_i = \mathbf{r} \cdot \mathbf{N}_\alpha = D > 0, \quad (12)$$

proving the theorem for  $\mathbf{p}$  outside the mesh. If  $\mathbf{p}$  is inside the mesh, the situation is essentially the same except for the fact that the involved normals point the other way. This means that the integral over  $\partial S_B$  changes sign. Thus,  $D$  becomes negative. which concludes our proof.  $\square$

Note that we do not assume that the closest point is unique. The proof requires only that  $\mathbf{c}$  is  $a$  closest point. This means that Theorem 1 also holds in the case where  $\mathbf{p}$  lies on the medial axis.

## 4 SIGNED DISTANCE ALGORITHM

All algorithms for computing unsigned distances from triangle meshes need to find the closest point on the triangle mesh. Specifically, for a point  $\mathbf{p}$ , the closest point on the mesh,  $\mathbf{c}$ , and the vector,  $\mathbf{r} = \mathbf{p} - \mathbf{c}$ , are computed since  $\|\mathbf{r}\|$  is the distance from  $\mathbf{p}$  to the mesh.<sup>6</sup>

We propose using the fact that  $\mathbf{r}$  is already obtained in conjunction with the result in Section 3 to form an integrated and simple method for computing the *signed* distance to a mesh.

Specifically, we propose augmenting the mesh structure with angle weighted pseudonormals for each face, edge, and vertex. These can be computed in a preprocessing step or computed during the actual voxelization.

Now, it is straightforward to extend any algorithm for computing *unsigned* distances to computing signed distances. Once  $\mathbf{c}$  and  $\mathbf{r}$  are found for a given  $\mathbf{p}$ , the associated distances are

$$d = \|\mathbf{r}\| \text{sign}(\mathbf{r} \cdot \mathbf{N}_\alpha),$$

instead of

$$d = \|\mathbf{r}\|.$$

Here,  $\mathbf{N}_\alpha$  is the angle weighted pseudonormal associated with  $\mathbf{c}$ .

### 4.1 A Concrete Algorithm

In this section, we propose a concrete algorithm for computing the signed distance to a 3D triangle mesh at an

5. Note that the area of a wedge cut out of the unit disc is equal to the angle of that wedge.

6. In fact,  $\mathbf{c}$  and  $\mathbf{p}$  may not be explicitly computed since we may obtain  $\|\mathbf{r}\|$  from the distance to the plane of the triangle combined with the distance in the plane to the triangle. Nevertheless, finding  $\mathbf{r}$  and  $\mathbf{c}$  explicitly will only add a small, constant overhead.

arbitrary point. The method is also applicable to signed distance field generation. In this or other cases where multiple distances are computed, it is possible to cache information from previous invocations.

The fastest proposed methods for computing the distance to a triangle mesh are based on hierarchical data structures such as hierarchies of bounding boxes [22] or a hierarchy of sphere swept primitives [23], [4]. Our method was inspired by [4], but uses oriented bounding boxes since they seem to provide a good trade-off between simplicity and efficiency.

During initialization, the vertex and edge normals are computed for each triangle. Then, the triangles are stored in a hierarchy of bounding boxes.

To query the hierarchy, we maintain a priority queue of bounding boxes whose key is the lower bound (most optimistic) on the shortest distance to the mesh. Initially, only the root node is contained in the queue. The top of the priority queue is extracted and its two children are inserted instead. We pick the top node again and, in this way, we always proceed down the tree along the nodes with the lowest bound on the shortest distance.

If the node picked from the queue contains only a single triangle, its lower and upper distance bounds are both identical to the distance to the triangle. In this case, the triangle distance is the actual shortest distance to the model and the algorithm returns that distance. The sign is computed based on the normal at the closest feature and the vector from the closest point to the query point.

### 4.2 Implementation Details

The type of bounding box used is of some importance. We use oriented bounding boxes (OBB) since they are well-known and produce a good fit compared to some of the alternatives such as axis aligned bounding boxes. AABBs were also implemented, but, while an AABB hierarchy is faster to construct, the queries are invariably much slower. To find the orientation of an OBB, a covariance matrix is computed for the distribution of points on the triangles contained in the bounding box. The eigenvectors of this covariance matrix define the orientation of the box, as discussed in [32].

A very simple but very important optimization is to only use square distances and compute the square root upon returning. Another important optimization used is to record the smallest upper bound on the shortest distance. In other words, some encountered bounding box has the smallest upper bound on the shortest distance. We record this value. When a bounding box is encountered whose lower distance bound is greater than this value, it is not inserted into the priority queue since it cannot contain the shortest distance.

The upper bound on the shortest distance is set to the distance to an arbitrary point on the mesh inside the bounding box. It is clear that any point on the mesh provides an upper bound on the shortest distance since an arbitrary point can never be closer than the closest point as that would entail *it being* the closest point. For this reason, we store in the bounding box the mesh vertex closest to the center of the box. This will almost invariably produce a

TABLE 1  
Tabulation of Our Results

|        |          | Hierarchy construction |        |                 |        |          |      | Distance Computation |       |                 |       |          |      | Ray Casting |       |
|--------|----------|------------------------|--------|-----------------|--------|----------|------|----------------------|-------|-----------------|-------|----------|------|-------------|-------|
|        |          | No sign comp.          |        | With sign comp. |        | Overhead |      | No sign comp.        |       | With sign comp. |       | Overhead |      | OBB         | AABB  |
| Model  | Polygons | OBB                    | AABB   | OBB             | AABB   | OBB      | AABB | OBB                  | AABB  | OBB             | AABB  | OBB      | AABB | OBB         | AABB  |
| Cube   | 12       | 3.5e-4                 | 1.2e-4 | 4.2e-4          | 1.9e-4 | 21%      | 53%  | 13.79                | 13.98 | 14.15           | 14.52 | 3%       | 4%   | 9.96        | 54.76 |
| Man    | 3330     | 0.16                   | 0.07   | 0.18            | 0.09   | 16%      | 35%  | 81.76                | 83.55 | 80.97           | 86.01 | -1%      | 3%   | 10.01       | 50.8  |
| Simple | 4096     | 0.18                   | 0.08   | 0.21            | 0.11   | 17%      | 37%  | 62.43                | 110.5 | 62.73           | 114.2 | 1%       | 3%   | 30.12       | 172.5 |
| Pieta  | 6976     | 0.37                   | 0.16   | 0.43            | 0.22   | 18%      | 39%  | 88.21                | 89.39 | 88.66           | 91.99 | 1%       | 3%   | 24.65       | 114.3 |
| Femur  | 7798     | 0.42                   | 0.18   | 0.5             | 0.25   | 18%      | 36%  | 103.4                | 134.5 | 103.8           | 138.3 | 0%       | 3%   | 11.5        | 54.54 |
| Bunny  | 11316    | 0.59                   | 0.26   | 0.7             | 0.36   | 19%      | 38%  | 81.77                | 117.9 | 81.14           | 121.6 | -1%      | 3%   | 40.74       | 202.3 |
| Horse  | 60680    | 3.69                   | 1.57   | 4.36            | 2.21   | 18%      | 41%  | 138.7                | 226.9 | 137.7           | 235.3 | -1%      | 4%   | 19.2        | 93    |
| Dragon | 276680   | 19.01                  | 8.62   | 22.63           | 12.39  | 19%      | 44%  | 148.3                | 225.1 | 146.9           | 231.8 | -1%      | 3%   | 40.75       | 180.2 |

All timings are given in seconds. The leftmost group of columns indicates the model used and the number of polygons. The center left group shows timings for the hierarchy construction. The timings were performed first without generation of edge and vertex normals for sign computation and then with this information. The final column in this group contains the sign-information generation overhead. The center right group contains the results of the distance field generation ( $128 \times 128 \times 128$  voxels). Again, we have timed the distance field generation without and then with sign computation and computed the overhead. The rightmost group of columns contains timings from the inside-outside test by ray casting. Again,  $128 \times 128 \times 128$  inside-outside tests were performed. (Note that the negative overhead values are due to statistical uncertainty.)

tighter upper bound on the distance than the point in the box furthest from the query point.

When distances are computed at many points (such as the voxels in a distance field), it is possible to exploit coherence. In accordance with the triangle inequality, the distance from a point to the mesh cannot be greater than the distance to an adjacent point plus the distance from that point to the mesh. Thus, we have an initial upper bound that can be used to cull nodes.

### 4.3 Evaluation

As a concrete experiment, we have used the algorithm to generate a discrete distance field of  $128 \times 128 \times 128$  voxels. The experiment has been carried out on a number of meshes and it was done twice. First, we computed a distance grid with signs and then we disabled the sign computation (both in the bounding hierarchy generation phase and the actual computation of distances) in order to discover how much extra overhead was involved in computing the sign.

Fortunately, the bounding box hierarchy used for distance queries is equally useful for ray intersections. Counting ray intersections is one of the most common techniques for the point in polyhedron test. Hence, this is an obvious benchmark and we have compared the time it takes to generate a signed distance field to the time it takes to just compute the sign (i.e., perform the inside outside test) using ray casting.

The results are shown in Table 1. The timings were performed on a 2.2 GHz Intel Xeon processor. The timings were measured in wall-clock time and we picked the best of three runs for all experiments.

From the results, we may conclude that the time it takes to build an OBB or AABB tree hierarchy is affected by whether angle-weighted normals are computed. Typically, it increases the time by just less than 20 percent in the case of OBBs and around 40 percent in the case of AABBs. This is not insignificant, but it is a precomputation that can easily be stored if the model is static.

On the other hand, the actual distance computations are almost completely unaffected by the simple dot product

that is required in order to find out whether the point is inside or outside. This is particularly true when OBBs are used. In this case, the overhead is less than the statistical variation, as witnessed by the fact that the overhead is sometimes negative. In the case of AABBs, the overhead is small but significant.

It is clear from the timings that if only the sign and not the distance is required, it is more efficient to use ray casting. However, it should be noted that this test has not been implemented robustly. In a real application, it would be necessary to, e.g., recast a ray if it hits a vertex or an edge and that would incur some overhead. To sum up, if all that is required is a point in a polyhedron test, ray casting is the best option. However, if distances are also required, the sign can be had *almost for free* using our method.

Finally, we observe that the timings seem to vary greatly, independently of the number of polygons. This is true of both the distance field computations and the inside-outside tests using ray casting. This is not surprising when one considers that the models vary in shape and some fit the volume less well than others. Thus, for some models (e.g., the horse), there is much empty space in the volume, which speeds up the computations compared to more rotund models.

## 5 ROBUSTNESS CONSIDERATIONS

When considering robustness, it is important to distinguish between the case where the mesh is a closed 2-manifold and the case where it is not. The latter case cannot be handled by any signed distance algorithm since only a closed 2-manifold is guaranteed to partition space into an interior, an exterior, and a boundary region. The mesh may fail to be a closed 2-manifold for many reasons. For instance, it may fail to be watertight, it may self-intersect, and it is possible that the immediate neighborhood of a vertex cannot be mapped to a disc.

Unfortunately, degeneracies are also possible in meshes that do form 2-manifolds. In particular, the mesh may include some triangles which have collapsed into line segments or points. In this case, it is best to remove the

degenerate triangles. However, this should be done without introducing changes in the topology of the mesh. A robust procedure to remove such degeneracies was proposed by Botsch and Kobbelt [33].

Provided that the mesh is a 2-manifold, our proposed inside-outside computation is quite robust for reasons that are elaborated below. The same is true of scan-conversion only if certain precautions are taken. In particular, problems arise when mesh vertices or edges coincide with the points where inside outside queries are made. A technique for scan converting polygons is discussed in detail by Foley et al. [19] and Linhart discusses robustness issues pertaining to inside-outside testing by ray casting. If discrete signed distance fields are created using characteristics scan conversion [16], it is important to slightly dilate the characteristics so that they overlap. This is done in order to ensure that all voxels belong to a characteristic. However, if the dilation is too great, artifacts can also result, as reported by Erleben and Dohmann [34].

### 5.1 The Proposed Approach

In our dealings with the angle weighted pseudonormal for inside-outside computation, we have never experienced problems with numerical robustness. In the following, we will argue that this numerical robustness is an inherent property of the angle weighted pseudonormal.

Recall that the basic equation used for determining if a point is inside or out is

$$\sum \mathbf{r} \cdot \mathbf{n}_i \alpha_i > 0, \quad (13)$$

which, when discussing numerical robustness, becomes

$$\sum \mathbf{r} \cdot \mathbf{n}_i \alpha_i + \xi > 0, \quad (14)$$

where  $\xi$  is a numerical noise term. Numerical instability thus occurs when  $\xi$  has the same magnitude as the left side of (13). This can occur in the following situations:

- **The length  $r$  is comparable to the numerical precision.** In this case, the point  $\mathbf{p}$  is on the surface, within numerical precision. Thus, the signed distance is zero within the same bound making the sign irrelevant.
- **All the  $\mathbf{n}_i$  are perpendicular to  $\mathbf{r}$ .** Since  $\mathbf{r}$  connects a point  $\mathbf{p}$  to its closest point on the mesh, this can only happen if the mesh violates the manifold assumption by collapsing or being within numerical precision of doing so.
- **All the  $\alpha_i$  are small.** This implies that all the  $\mathbf{n}_i$  are considerably affected by numerical noise, increasing the noise term  $\xi$ . On the other hand, the normals are weighted by their angle, which diminishes the effect of these less reliable normals. Thus, all  $\alpha_i$  have to be small to give serious problems. If all  $\alpha_i$  approach numerical precision, the manifold assumption is challenged.

A slightly more subtle concern is that we may erroneously consider a point to be closest to the wrong feature of the mesh, i.e., it is computed that  $\mathbf{c}'$  is the closest point instead of  $\mathbf{c}$ . However, if the misclassification is due to

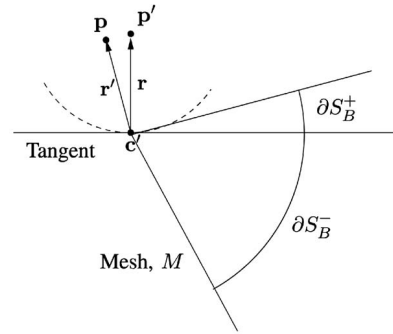


Fig. 8. Considering the same situation as in Fig. 7, apart from  $\mathbf{c}'$  not being the closest point to  $\mathbf{p}$ . In this case, the flux of  $F(\mathbf{q}) = \mathbf{r}$  is positive through  $\partial S_B^+$  and negative through  $\partial S_B^-$ .

numerical noise, we can safely assume that there is a point  $\mathbf{p}'$  nearby which does have  $\mathbf{c}'$  as its closest point. Assuming  $\|\mathbf{r}\| \gg 0$ , the angle  $\angle(\mathbf{r}, \mathbf{r}')$  will be close to zero. This situation is as shown in Fig. 8.

From the proof of Theorem 1, it is seen that the flux of  $F(\mathbf{q}) = \mathbf{r}$  through the entire  $\partial S_B$  must be negative<sup>7</sup> in order to achieve the correct result. But, if  $\mathbf{c}'$  is not the closest point,  $\partial S_B$  can be divided into a region of positive flux,  $\partial S_B^+$ , and one of negative flux,  $\partial S_B^-$ . Fortunately, if  $\angle(\mathbf{r}, \mathbf{r}')$  is very small, the positive region,  $\partial S_B^+$ , will also be very small, as illustrated in Fig. 8. Since  $\partial S_B = \partial S_B^- \cup \partial S_B^+$  and  $\partial S_B$  is only very small if the mesh is close to collapsing, it can be assumed that

$$\left| \int_{\partial S_B^+} F \cdot \mathbf{n}(\tau) d\tau \right| < \left| \int_{\partial S_B^-} F \cdot \mathbf{n}(\tau) d\tau \right| \Rightarrow \int_{\partial S_B} F \cdot \mathbf{n}(\tau) d\tau < 0,$$

unless the manifold assumption is violated or close to being so compared to numerical precision.

The robustness claim is thus, to a large extent, based on the manifold assumption, i.e., the assumption that the mesh does not collapse or come within numerical precision of doing so. The question thus arises of how close can a mesh come to collapsing before numerical issues start to arise. To give a coarse estimate of this, we performed a simple experiment. The mesh used for this experiment was a five-sided pyramid whose base vertices lay randomly on a circle of varying radius, as illustrated in Fig. 9. In this setting, define the margin as the least amount the pseudonormal can turn in *any* direction while maintaining its discriminatory power for *all* points closest to the vertex. More formally, this margin is obtained for a given pyramid and, thus,  $\mathbf{n}_\alpha$  as

$$\inf_{\mathbf{r} \in V_c^+} \arcsin(\mathbf{n}_\alpha \mathbf{r}), \quad (15)$$

where  $V_c^+$  is the set of vectors to all points outside the mesh and closest to the vertex  $\mathbf{c}$ . In principle, this margin will never be zero, as proven here, unless due to numerical

7. Dealing only with point outside the mesh for simplicity. As with the proof, the same arguments can be made for points inside the mesh by simply flipping the signs.

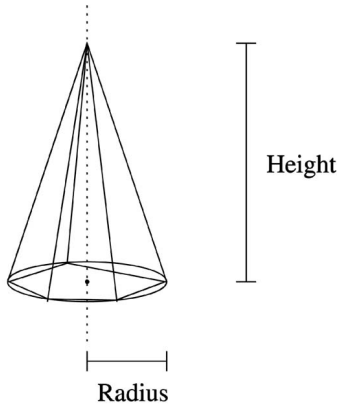


Fig. 9. The five-sided pyramid of unit height and varying radius which was used for the numerical robustness experiments. In all experiments, the apex (which is the vertex under consideration) is the closest mesh point.

noise. The negative margins thus indicate that  $\xi$  has become too large in (14) and the discriminatory ability is lost due to noise. In relation to Fig. 8, this margin can also be seen as how large the angle between  $r$  and  $r'$  can be while maintaining the discriminatory power.

For each radius, we computed the min, mean, and max values of (15) over 1,000 experiments. The results are depicted in Fig. 10, where it is seen that numerical issues start to occur at around a radius of  $1.5e-4$ . Since the pyramid has unit height, this should be sufficient for almost all meshes encountered. Note that this experiment was carried out without paying special attention to numerical issues in the implementation.

In conclusion, the angle weighted pseudonormal is numerically robust *without* the need to handle special cases. Second, if infinite numerical robustness is needed, infinite

precision arithmetic c.f., e.g., [35] is required, as is generally the case within the field of computational geometry.

## 6 DISCUSSION

We have proposed and proven that the angle weighted pseudonormal proposed by Thürmer and Wüthrich [1] and Séquin [2] has the property that it can be used for determining whether a given point is inside or outside a given mesh or polyhedron. It is also demonstrated that a variety of other pseudonormals do *not* possess this property.

This new insight means that the inside outside test for a polyhedron with triangular faces can be performed using only information obtained from the nearest point on the mesh. Put differently, the usual inside outside test for smooth surfaces has been generalized to triangle meshes.

It has been argued that this test is numerically robust, and its applicability has been demonstrated by sketching a simple method for computing signed distances. The method can be used as an extension to existing distance algorithms and we have also provided a concrete example of a specific distance algorithm extended with sign computation. Our findings show that (except for initialization) the overhead due to sign computation is almost immeasurably small when the algorithm is used for signed distance field generation.

The relevance of these results is not limited to signed distance field computation, however, and it strengthens the use of angle weighted pseudonormal as a generalization of face normals.

## APPENDIX

In the following, an outline of a proof that the convex hull normal,  $\mathbf{n}_h$ , has the property expressed in (3) is presented.

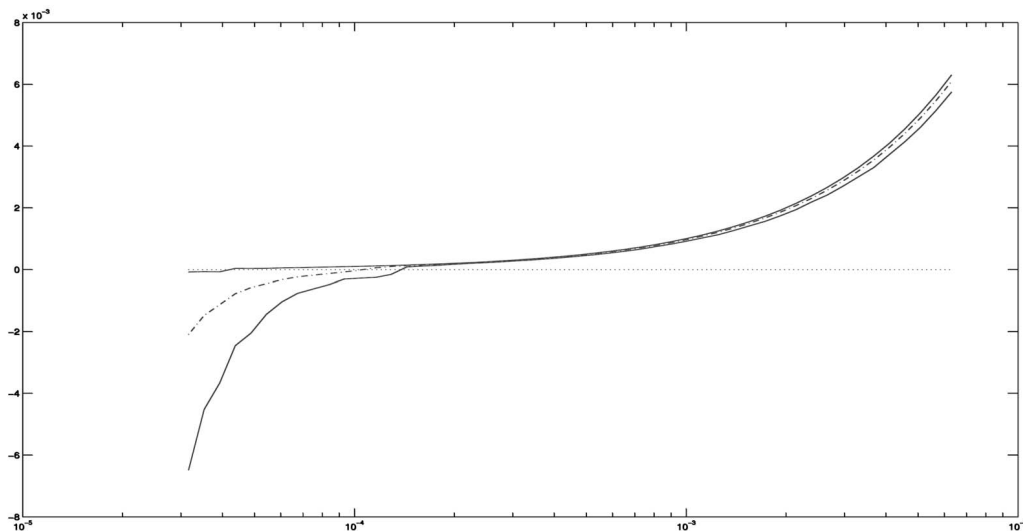


Fig. 10. The margins of the pyramid illustrated in Fig. 9, where the radius is depicted along the abscissa and the margin along the ordinate. The plots show the maximum, minimum, and mean margins from 1,000 runs. It should be noted that the margin calculation itself, given the angle weighted pseudonormal, is just as susceptible to noise as the angle weighted pseudonormal itself. But, for achieving a ball park figure, this will do.

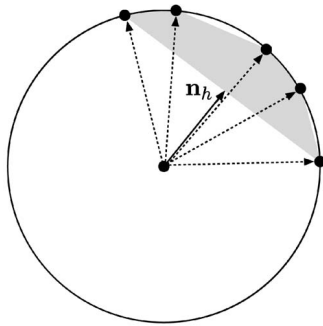


Fig. 11.  $\mathbf{n}_h$  for a vertex and the convex hull of the tips of the normals of faces incident on that vertex.

Allegedly, this is known, and we do not claim novelty for the following outline, which is mainly included for completeness.

We consider only vertices at convex points of the mesh, i.e., the Voronoi region is outside of the mesh.<sup>8</sup> Given a vertex, define the Voronoi cone as the Voronoi region of the vertex if the vertex and its incident elements were the only elements of the mesh. It is seen that any point on the border of this Voronoi cone can be written as a positive linear combination of the normals of the incident faces.

All Voronoi regions are convex, hence any point in the Voronoi cone can be written as a positive linear combination of two of its border points and, hence, as a positive linear combination of the normals of the incident faces ( $\mathbf{n}_i$ ). Since all points in the Voronoi region of the vertex are also in the Voronoi cone, all points,  $\mathbf{p}$ , in the Voronoi region can be written as a positive linear combination of incident face normals,  $\mathbf{n}_i$ , i.e.,

$$\mathbf{p} = [\mathbf{n}_1, \dots, \mathbf{n}_k] \begin{bmatrix} \gamma_1 \\ \vdots \\ \gamma_k \end{bmatrix}, \quad \forall i \gamma_i \geq 0. \quad (16)$$

Let us construct the convex hull of the tips of the normals,  $\mathbf{n}_i$ . Define  $\mathbf{n}_h$  as the vector from the origin to the closest point in this convex hull, as illustrated in Fig. 11. From Theorem 4.1 in [31], it follows that  $\mathbf{n}_h$  is a separating axis of the origin and the convex hull unless the convex hull includes the origin. Since the latter possibility would entail a degenerate mesh, we conclude that  $\mathbf{n}_h$  is a separating axis. Therefore,

$$\forall i \quad \mathbf{n}_h^T \mathbf{n}_i > 0. \quad (17)$$

Combining (16) and (17), we obtain

$$\mathbf{n}_h^T \mathbf{p} > 0, \quad (18)$$

which concludes the proof.  $\square$

## ACKNOWLEDGMENTS

The authors would like to thank Kenny Erleben and Henrik Dohlmann for sharing their experiences regarding the

characteristics scan conversion method. They would also like to thank Miloš Šrámek and an anonymous reviewer for valuable comments.

## REFERENCES

- [1] G. Thürmer and C. Wüthrich, "Computing Vertex Normals from Polygonal Facets," *J. Graphics Tools*, vol. 3, no. 1, pp. 43-46, 1998.
- [2] C.H. Séquin, "Procedural Spline Interpolation in Unicubix," *Proc. Third USENIX Computer Graphics Workshop*, pp. 63-83, 1986.
- [3] B. Payne and A. Toga, "Distance Field Manipulation of Surface Models," *IEEE Computer Graphics and Applications*, vol. 12, no. 1, pp. 65-71, 1992.
- [4] A. Guézic, "Meshsweeper: Dynamic Point-to-Polygonal Mesh Distance and Applications," *IEEE Trans. Visualization and Computer Graphics*, vol. 7, no. 1, pp. 47-60, Jan.-Mar. 2001.
- [5] J. Sethian, *Level Set Methods and Fast Marching Methods Evolving Interfaces in Computational Geometry, Fluid Mechanics, Computer Vision, and Materials Science*. Cambridge Univ. Press, 1999.
- [6] S.J. Osher and R.P. Fedkiw, *Level Set Methods and Dynamic Implicit Surfaces*, first ed. Springer Verlag, Nov. 2002.
- [7] *Geometric Level Set Methods in Imaging, Vision, and Graphics*, S. Osher and N. Paragios, eds. Springer, 2003.
- [8] M. Lin and S. Gottschalk, "Collision Detection between Geometric Models: A Survey," *Proc. IMA Conf. Math. of Surfaces*, 1998.
- [9] M.G. Coutinho, *Dynamic Simulations of Multibody Systems*. Springer, 2001.
- [10] E. Guendelman, R. Bridson, and R.P. Fedkiw, "Nonconvex Rigid Bodies with Stacking," *ACM Trans. Graphics*, vol. 22, no. 3, pp. 871-878, 2003.
- [11] J. Linhart, "A Quick Point-in-Polyhedron Test," *Computers & Graphics*, vol. 14, no. 3, pp. 445-448, 1990.
- [12] F.R. Feito and J.C. Torres, "Inclusion Test for General Polyhedra," *Computers & Graphics*, vol. 21, no. 1, pp. 23-30, 1997.
- [13] P.C.P. Carvalho and P.R. Cavalcanti, "Point in Polyhedron Testing Using Spherical Polygons," *Graphics Gems V*, A.W. Paeth, ed., pp. 42-49, AP Professional, 1995.
- [14] M. Jones, "The Production of Volume Data from Triangular Meshes Using Voxelisation," *Computer Graphics Forum*, vol. 15, no. 5, pp. 311-318, 1996.
- [15] F. Dacheville and A. Kaufman, "Incremental Triangle Voxelization," *Proc. Graphics Interface 2000*, pp. 205-212, 2000.
- [16] S. Mauch, "Efficient Algorithms for Solving Static Hamilton-Jacobi Equations," PhD dissertation, Caltech, 2003.
- [17] C. Sigg, R. Peikert, and M. Gross, "Signed Distance Transform Using Graphics Hardware," *Proc. IEEE Visualization '03*, pp. 83-90, 2003.
- [18] A. Sud, M.A. Otaduy, and D. Manocha, "Difi: Fast 3D Distance Field Computation Using Graphics Hardware," *Proc. Eurographics*, vol. 23, no. 3, 2004.
- [19] J. Foley, A. van Dam, S. Feiner, and J. Hughes, *Computer Graphics: Principles and Practice in C*, second ed. Addison-Wesley, 1995.
- [20] S.F.F. Gibson, R.N. Perry, A.P. Rockwood, and T.R. Jones, "Adaptively Sampled Distance Fields: A General Representation of Shape for Computer Graphics," *Proc. SIGGRAPH 2000*, pp. 249-254, 2000.
- [21] J. Huang, Y. Li, R. Crawfis, S.-C. Lu, and S.-Y. Liou, "A Complete Distance Field Representation," *Proc. Visualization 2001*, pp. 247-254, 2001.
- [22] D.E. Johnson and E. Cohen, "Bound Coherence for Minimum Distance Computations," *Proc. 1999 IEEE Int'l Conf. Robotics and Automation*, pp. 1843-1848, 1999.
- [23] E. Larsen, S. Gottschalk, M.C. Lin, and D. Manocha, "Fast Proximity Queries with Swept Sphere Volumes," technical report, Dept. of Computer Science, Univ. of North Carolina Chapel Hill, 1999.
- [24] J. Wu and L. Kobbelt, "Piecewise Linear Approximation of Signed Distance Fields," *Proc. Vision, Modeling, and Visualization 2003*, 2003.
- [25] H. Fuchs, Z. Kedem, and B. Naylor, "On Visible Surface Generation by A Priori Tree Structures," *Proc. Seventh Ann. Conf. Computer Graphics and Interactive Techniques*, pp. 124-133, 1980.
- [26] H. Gouraud, "Continuous Shading of Curved Surfaces," *IEEE Trans. Computers*, vol. 20, no. 6, pp. 623-629, June 1971.
- [27] A.S. Glassner, *Computing Surface Normals for 3D Models*, pp. 562-566. Academic Press, 1990.

8. The proof extends trivially to the case where the point is concave.

- [28] N. Max, "Weights for Computing Vertex Normals from Facet Normals," *J. Graphics Tools*, vol. 4, no. 2, pp. 1-6, 1999.
- [29] H. Aanaes and J.A. Baerentzen, "Pseudo-Normals for Signed Distance Computation," *Proc. Vision, Modeling, and Visualization 2003*, 2003.
- [30] P. Sander, X. Gu, S. Gortler, H. Hoppe, and J. Snyder, "Silhouette Clipping," *Proc. ACM SIGGRAPH Conf. Computer Graphics*, pp. 327-334, 2000.
- [31] G. vandenBergen, *Collision Detection in Interactive 3D Environments*. Morgan Kaufmann, 2004.
- [32] S. Gottschalk, "Collision Queries Using Oriented Bounding Boxes," PhD dissertation, Univ. of North Carolina at Chapel Hill, 2000.
- [33] M. Botsch and L. Kobbelt, "A Robust Procedure to Eliminate Degenerate Faces from Triangle Meshes," *Vision, Modeling, Visualization 2001 Proc.*, 2001.
- [34] K. Erleben and H. Dohlmann, personal communications.
- [35] J.R. Shewchuk, "Robust Adaptive Floating-Point Geometric Predicates," *Proc. 12th Ann. Symp. Computational Geometry*, pp. 141-150, May 1996.



**J. Andreas Bærentzen** received the MSc and PhD degrees from the Technical University of Denmark (DTU). He is now an assistant professor of informatics and mathematical modeling at DTU. His research is focused on the development of techniques for geometry representation and effective techniques for interactive shape manipulation and visualization.



**Henrik Aanaes** received the master's and PhD degrees from the Technical University of Denmark, where he currently holds a position as assistant professor. His field of research is the reconstruction of 3D objects from images thereof, primarily the structure from motion problem. He is also interested in methods for the handling of 3D objects, which is an integral part of his research activities. He is a member of the IEEE.

▷ For more information on this or any other computing topic, please visit our Digital Library at [www.computer.org/publications/dlib](http://www.computer.org/publications/dlib).