HKU CS Tech Report TR-2006-01

Computing the Minimum Directional Distance between Two Convex Polyhedra

Yi-King Choi, Xueqing Li, Fengguang Rong, Wenping Wang and Stephen Cameron

***Abstract*— Given two convex polyhedra P and Q and a direction $\mathbf{s}$, the minimum directional distance (MDD) is defined to be the shortest translational distance along the direction $\mathbf{s}$ that is required to make P and Q just in contact. In this paper we propose a novel method, called MDD-DUAL , for computing the MDD between two convex polyhedra. The MDD is equivalent to the shortest distance between the origin and the Minkowski difference M of the polyhedra in the direction $\mathbf{s}$. Our idea is to reduce the MDD problem to seeking a vertex on the dual polyhedron of M with the maximum signed distance from a special plane by means of a duality transformation. We show that this is equivalent to locating a face on M with which a ray shooting from the origin in the direction $\mathbf{s}$ first intersects. The MDD can then easily be derived from the signed distance.**

Our algorithm constructs only a subset of the faces on M along the search path. By further breaking down the search into three phases, each on a different type of faces on M, MDD-DUAL reports the MDD between two convex polyhedra efficiently.

***Index Terms*— minimum directional distance, directional separating distance, directional penetration depth, convex polyhedra, duality transformation, signed distance.**

I. INTRODUCTION

It is often important to determine the distance between two geometric objects in order to understand their spatial relationship. When the two objects are separate, we focus in their separating distance, i.e. the minimum translation distance to bring them just in touch; and in the case of two intersecting objects, it is then the penetration depth, i.e. the minimum translation distance to separate them, that is of our interests. The distances between objects are useful in many applications in robotics and computer graphics, or other areas that require physical simulations, where responses are often deduced based on the distance information. Distance computation is also commonly studied in relation to collision detection of objects (see a survey in [1], [2]). Using a commonly used measure by Cameron and Cully [3], a positive distance corresponds to separation, while a negative distance means intersection [3]. In this paper, we will concentrate on distance computation for convex polyhedra.

Numerous work has been conducted in the fields of computational geometry, computer graphics and robotics for computing the shortest distance between convex polyhedra. The Lin-Canny (LC) [4] and V-Clip [5] are feature-based algorithms that find a sequence of pairs of candidate witness points which eventually converge to the closest features between the two polyhedra. The LC algorithm has an almost-constant complexity in exploiting temporal coherence and reporting the witness points upon consecutive invocation of the method. Another commonly used algorithm of Gibert, Johnson and Keerthi (GJK) [6], works on the simplices of the Minkowski difference of two polyhdera and uses convex optimization techniques to compute the closest points. Modified approaches [7], [8] and improved implementations [9] based on GJK were developed. It is also shown in [8] that an enhanced version of GJK has $\mathcal{O}(1)$ time cost under the assumption of strong geometric coherence. Specific to computing the penetration depth of two convex polyhedra, Agarwal et al. [10] presented a randomized algorithm whose expected running time is $\mathcal{O}(m^{\frac{3}{4}+\epsilon}+n^{\frac{3}{4}+\epsilon}+m^{1+\epsilon}+n^{1+\epsilon})$, for any constant $\epsilon > 0$, where m and n are the number of faces of the two polyhedra. Apart from this theoretical result, Kim et al. [11] devised an incremental algorithm with an implementation for estimating the penetration depth.

The separating distance and penetration depth between two convex polyhedra tell how far the two polyhedra can translate towards or away from each other, respectively, so that they just touch each other externally. The translation must always be along the direction between the two features that realize the shortest distance on each of the polyhedra. However, in some applications the directions of object translations are not unrestricted and are often confined in just several specific directions. Computing the shortest distance in this case may not give the desired results, since its corresponding distance may not align with the allowable directions. In these situations, the directional separating distance or directional penetration depth would give a more appropriate indication to solve the problems.

We follow similar notation as in [3] and define the function $\mathrm{MDD}^{+}(A, B, \mathbf{s})$ for two objects A and B in the direction $\mathbf{s} \in \mathbb{R}^{\mathbf{2}}$ as

$$\mathrm{MDD}^{+}(A,B,\mathbf{s}) = \min\{\|t\mathbf{s}\| \mid \mathrm{Int}(A) \cap \mathrm{Int}(B^{t\mathbf{s}}) = \emptyset \wedge \partial A \cap \partial B \neq \emptyset, t \in \mathbb{R}\}$$

where $\mathrm{Int}(A)$ and ∂A denote the interior and boundary of A, respectively, and $B^{t\mathbf{s}} = \{\mathbf{b} + t\mathbf{s} \mid \mathbf{b} \in B\}$ is the result of B translated by $t\mathbf{s}$. MDD^{+} is zero when A and B are in external contact; otherwise, it is positive for both intersecting and separate objects. To distinguish the two cases, we further define

$$\mathrm{MDD}(A,B,\mathbf{s}) = \begin{cases} -\mathrm{MDD}^{+}(A,B,\mathbf{s}) & \text{if } A \text{ intersects } B, \\ +\mathrm{MDD}^{+}(A,B,\mathbf{s}) & \text{otherwise.} \end{cases} \tag{1}$$

to be the minimum directional distance (MDD) of two objects A and B in the direction $\mathbf{s}$. The MDD is positive and gives the directional separating distance if two objects are separate; it is negative and gives the directional penetration depth if they intersect. It can be shown that the MDD of two convex polyhedra is equivalent to the shortest directional distance between the origin and their Minkowski difference polyhedron [12]. Therefore, the MDD can be found by intersecting a ray from the origin with the Minkowski difference polyhedron, whose geometric complexity is known to be of $\mathcal{O}(n^2)$. Hence, a brute-force algorithm in finding the MDD will run in $\mathcal{O}(n^2)$ time. Dobkin et al. [12] presented an $\mathcal{O}(\log^2 n)$ algorithm for computing the directional penetration depth of two convex polyhedra using the hierarchical representation of polyhedra by Dobkin and Kirkpatrick [13], [14], where n is the total number of vertices of the two polyhedra.

A. Major contributions

In this paper, we present an algorithm MDD-DUAL to compute the minimum directional distance between two convex polyhedra, both in cases where the polyhedra are separate or intersect. When the polyhedra are separate (or intersect), the directional separating distance (or penetration depth, respectively) is determined. By means of a duality transformation, we define a convex function (namely, the signed distance of a vertex on the Minkowski difference M of the polyedra to a plane) in the dual space which is shown to be equivalent to finding a face containing the intersection of a ray from the origin with M. The convex nature of the latter process is not intuitively seen, but is confirmed easily by its dual counterpart. Moreover, the geometric complexity of M is known to be $\mathcal{O}(n^2)$, of which most of the faces are of EE-type (see section II-C). By breaking down the search on the Minkowski difference into 3 different phases, we can skip most of the EE-type faces and obtain the optimal result in an efficient way.

B. Paper Organization

In Section II, we first introduce some definitions and notation that will be used in our discussion. We will then explain our algorithm in detail in Section IV, where the correctness of MDD-DUAL will be proved. The algorithm MDD-DUAL to solve the collision detection problem of two convex polyhedra will be demonstrated in Section V. Some implementation details will be given in Section VI. The performance of MDD-DUAL are presented in section VII.

II. PRELIMINARIES

A. Definitions and notations

Let P be a convex polyhedron in $\mathbb{E}^3$. Let $\mathcal{V}_P$, $\mathcal{F}_P$, and $\mathcal{E}_P$ denote the set of vertices, faces, and edges of P, respectively.

Definition 1: Given two polyhedra P and Q as two point sets, P and Q are said to be *overlapping* if $P \cap Q \neq \mathbf{0}$; otherwise, they are *separate*.

Definition 2: A *supporting vertex* $\mathrm{s}_P(\mathbf{n})$ of P in the direction $\mathbf{n} \neq \mathbf{0}$ is a vertex in $\mathcal{V}_P$ satisfying $\mathbf{n} \cdot \mathrm{s}_P(\mathbf{n}) = \max\{\mathbf{n} \cdot \mathbf{v} \mid \mathbf{v} \in \mathcal{V}_P\}$, where $\mathbf{x} \cdot \mathbf{y}$ is the dot-product of the vectors $\mathbf{x}$ and $\mathbf{y}$.

For a supporting vertex $\mathrm{s}_P(\mathbf{n})$, we also have $\mathbf{n} \cdot \mathrm{s}_P(\mathbf{n}) = \max\{\mathbf{n} \cdot \mathbf{p} \mid \mathbf{p} \in P\}$. We may also define the *supporting edge (or face)* of P in $\mathbf{n}$ as the edge (or face) that contains two (or more than two) supporting vertices in $\mathbf{n}$.

Definition 3: Let $\hat{\mathbf{n}}(f)$ denote the unit normal vector of a face f. The supporting vertex of P for f, $\mathrm{s}_P(f)$, is defined as the supporting vertex of P in the normal direction of f, i.e., $\mathrm{s}_P(f) = \mathrm{s}_P\big(\hat{\mathbf{n}}(f)\big)$.

B. Gaussian image of a polyhedron

The Gaussian image $G(P)$ of a convex polyhedron P is a planar graph embedded on the unit sphere $\mathcal{S}^2$ (Figure 1): A face $f \in \mathcal{F}_P$ corresponds to a point $G(f) = \hat{\mathbf{n}}(f) \in \mathcal{S}^2$; an edge $e \in \mathcal{E}_P$ common to two faces $f_0, f_1 \in \mathcal{F}_P$ corresponds to a great arc $G(e)$ connecting two vertices $G(f_0)$ and $G(f_1)$ on $\mathcal{S}^2$; a vertex $\mathbf{v} \in \mathcal{V}_P$ common to the faces $f_0, \ldots, f_m$ corresponds to a convex spherical polygon $G(\mathbf{v})$ whose vertices are $G(f_0), \ldots, G(f_m)$.

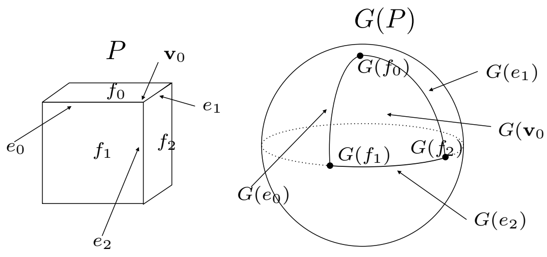


Fig. 1. A polyhedron P and its Gaussian image $G(P)$ on $\mathcal{S}^2$.

For a feature (i.e. vertex, edge or face) ϕ of a polyhedron P, the Gaussian image of ϕ is the set of normal directions of planes that may come into contact with P at ϕ. In other words, ϕ is the supporting feature of P in the directions represented by its Gaussian image.

C. Minkowski Sum of Two Polyhedra

Given two polyhedra P and Q, let $-Q = \{-q \mid q \in Q\}$. We consider the Minkowski sum M of P and $-Q$ (or equivalently, the Minkowski difference P and Q) defined by

$$M \equiv P \oplus -Q = \{\mathbf{p} - \mathbf{q} \mid \mathbf{p} \in P, \mathbf{q} \in Q\}.$$

The origin $\mathbf{0}$ is in M if and only if there are some $\mathbf{p} \in P$ and $\mathbf{q} \in Q$ such that $\mathbf{p} = \mathbf{q}$, i.e. P and Q share a common point (so they overlap).

As P and Q are both convex, M is also a convex polyhedron [15]. The Gaussian image of M (denoted by $G(M)$) can be obtained by superimposing the Gaussian images $G(P)$ and $G(-Q)$. For any face $f_p \in \mathcal{F}_P$, the point $G(f_p)$ must fall within the region $G\big(\mathrm{s}_{-Q}(f_p)\big)$, i.e. the Gaussian image of the supporting vertex of $-Q$ for f_p, on $\mathcal{S}^2$. The same is true regarding any face $f_q \in \mathcal{F}_{-Q}$ by interchanging the roles of P and $-Q$, which are symmetric in $M = P \oplus -Q$. Hence, each point in $G(P)$ and $G(-Q)$ corresponds to a face in M (Fig. 2). Furthermore, each arc-arc intersection on $\mathcal{S}^2$ corresponds to a pair of edges (one from P and one from $-Q$) sharing a common normal direction and amounts to a face in M.

Hence, the faces in $\mathcal{F}_M$ can be classified into the following three subsets (Figure 3):

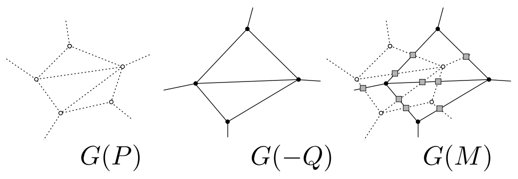


Fig. 2. The planar representation of the Gaussian image $G(M)$ by superimposing $G(P)$ and $G(-Q)$. There are three types of vertices in $G(M)$: (i) (white point) a point of $G(P)$ falling within a region of $G(-Q)$, i.e., a face in $\mathcal{F}_{\mathrm{fv}}$; (ii) (black point) a point of $G(-Q)$ falling within a region of $G(P)$, i.e., a face in $\mathcal{F}_{\mathrm{vf}}$; and (iii) (shaded square) the intersection point of two arcs, each from $G(P)$ and $G(-Q)$, i.e., a face in $\mathcal{F}_{\mathrm{ee}}$.

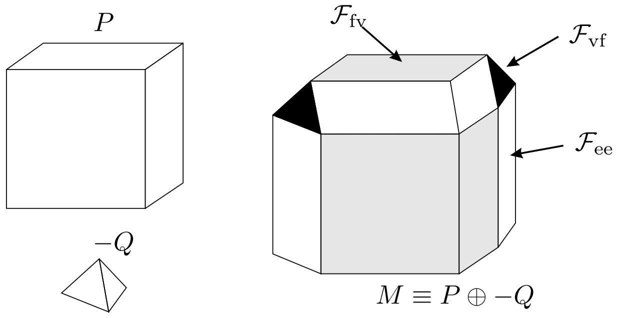

P
$\mathcal{F}_{\mathrm{fv}}$
$\mathcal{F}_{\mathrm{vf}}$
$\mathcal{F}_{\mathrm{ee}}$
$-Q$
$M \equiv P \oplus -Q$

Fig. 3. The Minkowski sum M of P and $-Q$. Faces on M can be classified as of type $\mathcal{F}_{\mathrm{fv}}$, $\mathcal{F}_{\mathrm{vf}}$, or $\mathcal{F}_{\mathrm{ee}}$.

$\mathcal{F}_{\mathrm{fv}}$: Each face $F(f_p, \mathbf{v}_q)$ is a point set $\{\mathbf{x} + \mathbf{v}_q \mid \mathbf{x} \in f_p\}$, where $f_p \in \mathcal{F}_P$ and $\mathbf{v}_q \in \mathcal{V}_{-Q}$. Also, $\mathbf{v}_q = \mathrm{s}_{-Q}(f_p)$.

$\mathcal{F}_{\mathrm{vf}}$: Each face $F(\mathbf{v}_p, f_q)$ is a point set $\{\mathbf{v}_p + \mathbf{x} \mid \mathbf{x} \in f_q\}$, where $f_q \in \mathcal{F}_{-Q}$ and $\mathbf{v}_p \in \mathcal{V}_P$. Also, $\mathbf{v}_p = \mathrm{s}_P(f_q)$.

$\mathcal{F}_{\mathrm{ee}}$: Each face $F(e_p, e_q)$ is a parallelogram with vertices $\mathbf{v}_0 = \mathbf{v}_{p_0} + \mathbf{v}_{q_0}, \mathbf{v}_1 = \mathbf{v}_{p_1} + \mathbf{v}_{q_0}, \mathbf{v}_2 = \mathbf{v}_{p_1} + \mathbf{v}_{q_1}, \mathbf{v}_3 = \mathbf{v}_{p_0} + \mathbf{v}_{q_1}$ where $\mathbf{v}_{p_0}, \mathbf{v}_{p_1} \in \mathcal{V}_P$, $\mathbf{v}_{q_0}, \mathbf{v}_{q_1} \in \mathcal{V}_{-Q}$, and $e_p = (\mathbf{v}_{p_0}, \mathbf{v}_{p_1}) \in \mathcal{E}_P$, $e_q = (\mathbf{v}_{q_0}, \mathbf{v}_{q_1}) \in \mathcal{E}_{-Q}$. Moreover, the Gaussian images of e_p and e_q intersect on $\mathcal{S}^2$.

D. *Duality Transformation*

In the projective 3-space, the concept of duality between points and planes is given by the symmetry between point-coordinates and plane-coordinates in the equation $\sum_{i=0}^{3} u_i x_i = 0$ [16], [17] (*). A more general formulation is to consider the self-dual duality with respect to a given non-singular quadric surface $\mathcal{B} : X^T B X = 0$ where $X = (x, y, z, 1)^T$ is the homogeneous coordinates of a point, and B is a 4×4 real symmetric matrix. The dual of a point Y_0 is a plane $\mathcal{Y} : Y_0^T B X = 0$ (the *polar* of Y_0) and the dual of a plane $\mathcal{V} : V_0^T X = 0$ is a point $U_0 = B^{-1} V_0$ (the *pole* of $\mathcal{V}$ [18]). It is easy to verify that if $\mathcal{Y}$ is the dual of Y_0, then Y_0 is the dual of $\mathcal{Y}$. Also, if Y_0 is a point on $\mathcal{B}$, its dual is the tangent plane to $\mathcal{B}$ at Y_0. The duality expressed in (*) is a special case where the quadrics $\mathcal{B}$ is an imaginary sphere.

In this work, we consider the duality transformation with respect to the unit sphere in $\mathbb{E}^3$. We now describe the dual relationship between a point and a plane in $\mathbb{E}^3$ in terms of affine coordinates $\mathbf{x} = (x, y, z)^T$. Suppose a plane Π, not passing through the origin, be given by $A^T \mathbf{x} = k$ in the *primal space* $\mathbb{E}$, where $A \in \mathbb{R}^3$ and k is a nonzero real number. A *duality transformation* maps Π to a point $\mathbf{w} = A/k$ in the *dual space* $\mathbb{E}^{3*}$. A point $\mathbf{u} \neq \mathbf{0}$ in $\mathbb{E}^3$ is transformed to a plane $\mathcal{U} : \mathbf{u}^T \mathbf{x} = 1$ in $\mathbb{E}^{3*}$ (Fig. 4). If we extend $\mathbb{E}^3$ to include the plane at infinity (i.e., the extended Euclidean space), a plane passing through the origin in $\mathbb{E}^3$ is mapped to a point at infinity in $\mathbb{E}^{3*}$; whereas the origin in $\mathbb{E}^3$ is transformed to the plane at infinity in $\mathbb{E}^{3*}$. Note that $\mathbb{E}^3$ is the dual space of $\mathbb{E}^{3*}$. In the sequel, we will use ψ^* to denote the dual counterpart of an entity ψ in $\mathbb{E}^3$. We may also consider a duality tranformation centred at an arbitrary point $\mathbf{c} \in \mathbb{E}^{\mathbf{3}}$. This can be done by first translating $\mathbb{E}^3$ to centre at $\mathbf{c}$ before applying duality, and we call $\mathbf{c}$ the *centre of duality*.

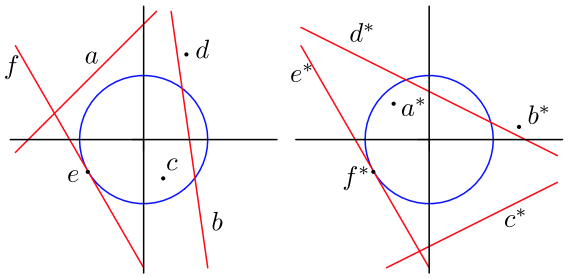


Fig. 4. A 2D illustration of duality transformation between the primal space (left) and the dual space (right).

Suppose that the centre of duality is contained in the interior of a convex polyhedron M, then every faces of M are properly transformed to a vertex not at infinity. The dual M^* is therefore a convex polyhedron; the vertices and faces of M^* are $\mathcal{F}_M^*$ and $\mathcal{V}_M^*$, respectively. The dual of an edge defined by two adjacent vertices $\mathbf{v}_0$, $\mathbf{v}_1$ in M is an edge common to two adjacent faces $\mathbf{v}_0^*$, $\mathbf{v}_1^*$ in M^*.

Property 1: Any point $\mathbf{x} \in \mathrm{Int}(M)$ is transformed by a duality to a plane $\mathbf{x}^*$ not intersecting M^*, where $\mathrm{Int}(M)$ stands for the interior of a polyhedron M; while any point $\mathbf{x} \notin M$ is transformed to a plane $\mathbf{x}^*$ that intersects M^*.

III. The Key Idea

In this section, we explain the fundamental concept of our algorithm, which relates the MDD problem of two polyhedra in the primal space to a search for a vertex on a Minkowski difference polyhdedron in the dual space.

A. *Condition for the separation of two polyhedra*

Let $M \equiv P \oplus -Q$ be the Minkowski difference of two convex polyhedra P and Q. We have,

$$\begin{aligned} P \text{ and } Q \text{ overlap} &\leftrightarrow \text{the origin } \mathbf{o} \in M \\ &\leftrightarrow \text{the plane } \mathbf{o}^* \text{ does not intersect } M^* \end{aligned}$$

where the centre of duality can be any fixed point $\mathbf{c} \neq \mathbf{0}$ in M (Figure 5). By checking the signed distance (which will be discussed in the next section) of the vertices of M^* to the plane $\mathbf{o}^*$, we can deduce whether $\mathbf{o}^*$ intersects M^* or not.

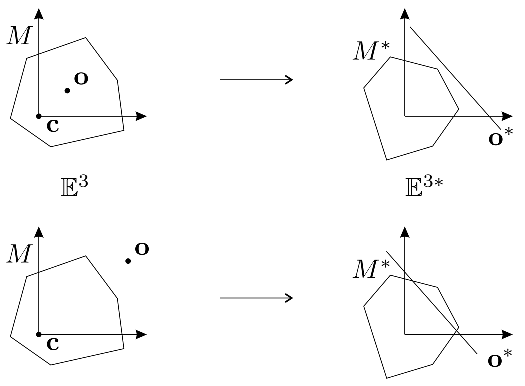


Fig. 5. A 2D illustration demonstrating the geometric relationship of M and $\mathbf{o}$ in the primal space $\mathbb{E}^3$ and the dual space $\mathbb{E}^{3*}$.

B. The Objective Function—Signed Distance Function

Given a plane $\Pi : A^T\mathbf{x} = k$ where $L \in \mathbb{R}^3$, $\mathbf{x} = (x, y, z)^T$, $k \in \mathbb{R}$, we assume that the plane equation is normalized such that $\|A\| = 1$ is the unit normal of Π pointing away from the origin so that $k > 0$ is the shortest distance from the origin $\mathbf{0}$ to the plane. The *signed distance* of a point $\mathbf{x}_0$ to the plane Π is then given by

$$\mathrm{d}_\Pi(\mathbf{x}_0) = \mathbf{A}^T\mathbf{x}_0 - k.$$

If $\mathbf{x}_0$ lies on Π, $\mathrm{d}_\Pi(\mathbf{x}_0) = 0$; if $\mathbf{x}_0$ lies on the same side of Π as $\mathbf{0}$, $\mathrm{d}_\Pi(\mathbf{x}_0) < 0$; and if $\mathbf{x}_0$ lies on the opposite side of Π to $\mathbf{0}$, $\mathrm{d}_\Pi(\mathbf{x}_0) > 0$.

Let $d_{\max} = \max\{\mathrm{d}_{\mathbf{o}^*}(\mathbf{m}) \mid \mathbf{m} \in M^*\}$ be the maximum signed distance of all points in M^* to the plane $\mathbf{o}^*$. Since M^* is convex, the point that attains the maximum signed distance $d_{\max}$ to the plane $\mathbf{o}^*$ must lie on the boundary of M^*, i.e. either on a vertex, a face or an edge. In any case, there is a vertex $f^*_{\max} \in M^*$ such that

$$\mathrm{d}_{\mathbf{o}^*}(f^*_{\max}) = d_{\max}.$$

Hence, to compute $d_{\max}$, we may consider only the vertices of M^* and find their signed distances to the plane $\mathbf{o}^*$.

The sign of $d_{\max}$ indicates whether the two polyhedra P and Q overlap. If $d_{\max} < 0$, all vertices on M^* are on the same side of $\mathbf{o}^*$ as the origin; M^* and $\mathbf{o}^*$ do not intersect and hence P and Q overlap. On the other hand, if $d_{\max} > 0$, at least one vertex is at the opposite side of $\mathbf{o}^*$ to the origin; in this case, $\mathbf{o}^*$ intersects M^* and therefore P and Q are separate. If $d_{\max} = 0$, $\mathbf{o}^*$ touches M^* at some boundary point, and $\mathbf{o}$ lies on the boundary of M. By the construction of M as described in section II-C, $\mathbf{o} = \mathbf{p} - \mathbf{q}$ for some boundary points $\mathbf{p}$ and $\mathbf{q}$ of P and Q, respectively. It implies that $\mathbf{p} = \mathbf{q}$ and P and Q share a common boundary point, i.e., P and Q touch each other.

Let f be a face of M in $\mathbb{E}^3$. We may then define the *signed distance* of f denoted by $\mathrm{d}(f)$, to be the signed distance of f^* to the plane $\mathbf{o}^*$ in $\mathbb{E}^{3*}$, i.e.

$$\mathrm{d}(f) = \mathrm{d}_{\mathbf{o}^*}(f^*).$$

Suppose f is contained in a plane $\mathcal{H}_f : \mathbf{N}^T\mathbf{x} = k$, where $\|\mathbf{N}\| = 1$ and $k > 0$. The plane equation of f is then $\mathbf{N}^T\mathbf{x} = k - \mathbf{N}^T\mathbf{c}$ after a translation of $-\mathbf{c}$, where $\mathbf{c}$ is the centre of duality. Hence, $f^* = \mathbf{N}/(k - \mathbf{N}^T\mathbf{c}) \in \mathcal{V}_{M^*}$ in the dual space. The origin $\mathbf{o}$ is translated to $-\mathbf{c}$ and therefore the plane equation of $\mathbf{o}^*$ is $-\mathbf{c}^T\mathbf{x} = 1$. The signed distance $\mathrm{d}(f)$ can then be expressed explicitly as:

$$\begin{aligned}\mathrm{d}(f) = \mathrm{d}_{\mathbf{o}^*}(f^*) &= \frac{-\mathbf{c}^T}{\|\mathbf{c}\|} \cdot \frac{\mathbf{N}}{k - \mathbf{N}^T\mathbf{c}} - \frac{1}{\|\mathbf{c}\|} \\ &= -\frac{k}{\|\mathbf{c}\|(k - \mathbf{N}^T\mathbf{c})}. \end{aligned} \qquad (2)$$

The signed distance of $\mathbf{c}$ and $\mathbf{o}$ to the plane $\mathcal{H}_{f_{\max}}$ (the containing plane of $f_{\max}$) are given by $\mathrm{d}_{\mathcal{H}_{f_{\max}}}(\mathbf{c}) = -(k - \mathbf{N}^T\mathbf{c})/\|\mathbf{N}\|$ and $\mathrm{d}_{\mathcal{H}_{f_{\max}}}(\mathbf{o}) = -k/\|\mathbf{N}\|$, respectively. If $\mathrm{d}(f_{\max}) < 0$, by Eq. (2), $k - \mathbf{N}^T\mathbf{c} > 0$ and therefore $\mathrm{d}_{\mathcal{H}_{f_{\max}}}(\mathbf{c})$ and $\mathrm{d}_{\mathcal{H}_{f_{\max}}}(\mathbf{o})$ are of the same sign; which means that $\mathbf{c}$ and $\mathbf{o}$ are on the same side of the face f and $\mathbf{o} \in M$. On the other hand, if $\mathrm{d}(f_{\max}) > 0$, $\mathbf{c}$ and $\mathbf{o}$ are on opposite sides of f and we have $\mathbf{o} \notin M$.

The function $\mathrm{d}(f)$ is defined in the dual space $\mathbb{E}^{3*}$ as the signed distance from the point f^* to the plane $\mathbf{o}^*$. We will now derive the geometric meaning of $\mathrm{d}(f)$ in the primal space $\mathbb{E}^3$. The quantity $\mathrm{d}(f) = \mathrm{d}_{\mathbf{o}^*}(f^*)$ uniquely determines a plane $\mathbf{l}^*$ in $\mathbb{E}^{3*}$ parallel to $\mathbf{o}^*$ such that $\mathrm{d}_{\mathbf{o}^*}(\mathbf{x}) = \mathrm{d}_{\mathbf{o}^*}(f^*)$ for all points $\mathbf{x} \in \mathbf{l}^*$ (Figure 6). The plane $\mathbf{l}^*$ therefore passes through f^* and is parallel to the plane $\mathbf{o}^*$. Since $\mathbf{l}^*$ has the same normal direction as $\mathbf{o}^*$, it can be shown easily that its dual point $\mathbf{l}$ must lie on the line $\mathbf{co}$. Moreover, since $\mathbf{l}^*$ passes through f^*, $\mathbf{l}$ must lie on the plane $\mathcal{H}_f$, the containing plane of f. This implies that $\mathbf{l}$ is the intersection of the plane $\mathcal{H}_f$ and the line $\mathbf{co}$.

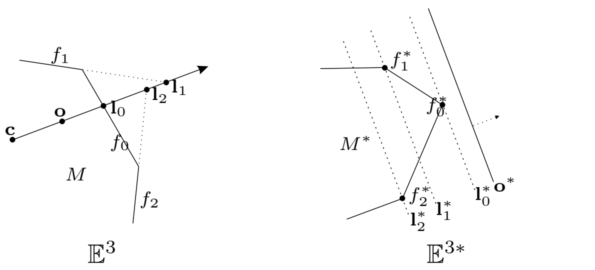


Fig. 6. The vertex f_0^* in $\mathbb{E}^{3*}$ attaining maximum signed distance to $\mathbf{o}^*$ is the dual of a face f_0 in $\mathbb{E}^3$ intersecting the directed line $\mathbf{co}$.

Lemma 1: The ray $\mathbf{co}$ intersects M in the face $f_{\max}$, whose signed distance is the maximum among all faces in $\mathcal{F}_M$, i.e. $\mathrm{d}(f_{\max}) = d_{\max}$.

Proof: Let the line $\mathbf{co}$ be given by $\mathbf{l}(t) = -t\mathbf{c}, t \in \mathbb{R}$. Then, the signed distance for a face f whose containing plane passes through $\mathbf{l}(t)$ is given by

$$\mathrm{d}(f) = \frac{1}{\|c\|}(\frac{1-t}{t}).$$

It means that the face $f_{\max}$, with the maximum signed distance among all faces in $\mathcal{F}_M$, has the closest intersection with the ray $\mathbf{co}$. Since M is convex and $\mathbf{c}$ is in the interior of M, this must be the case where the directed line $\mathbf{co}$ intersects the face $f_{\max}$. ■

Let $d_{\min} = \min\{\mathrm{d}_{\mathbf{o}^*}(\mathbf{m}) \mid \mathbf{m} \in M^*\}$ be the minimum signed distance of all points in M^* to the plane $\mathbf{o}^*$. Also, let $f_{\min}$ be the face having the minimum signed distance among all faces in $\mathcal{F}_M$, i.e. $\mathrm{d}(f_{\min}) = d_{\min}$. We then have the following corollary:

Corollary 2: The ray shooting from $\mathbf{c}$ in the direction $\mathbf{oc}$ intersects M in the face $f_{\min}$, whose signed distance is the minimum among all faces in $\mathcal{F}_M$.

It is now clear that by searching for a face in M with maximum signed distance, we are essentially determining whether $\mathbf{o}$ is in M by firing a ray from an interior point $\mathbf{c}$ of M to $\mathbf{o}$, obtaining a face of M that intersects the ray and deciding whether the intersection point lies within the line segment $\mathbf{co}$. It is hard to perceive that this entire process defines a convex function over all faces in M; but it becomes apparent when the same process is described by its dual equivalent $\mathrm{d}(f)$ which is clearly a convex function defined over all dual vertices f^* of M^*; since M^* is convex, a vertex f^* attaining a local maximum signed distance must also attain the global maximum signed distance among all vertices in M^*. It is also important to notice further that $\mathrm{d}(f)$ can be computed without even explicitly applying duality transformation on M, as can be seen from Eq. (2).

We may then define our objective function as the signed distance $\mathrm{d}(f)$ for all faces f in M. Our objective is to find $d_{\max}$, the maximum signed distance, so as to compute the MDD between two convex polyhedra P and Q. Starting from any face of $\mathcal{F}_M$, we go to the next face with the largest $\mathrm{d}(f)$ among all immediate neighbours of the current face. By this local search, we will visit faces with increasing $\mathrm{d}(f)$ and eventually stop at a face with a locally maximum signed distance. Due to the convexity of the objective function, this local search scheme will lead to the *optimal face*, $f_{\max}$, that attains the maximum signed distance $d_{\max}$ among all faces in M. It is possible to have more than one $f_{\max}$ that attain $d_{\max}$, which happens when the line $\mathbf{co}$ intersects M at an edge or a vertex.

C. Minimum directional distance (MDD)

Let α be the distance between the point $\mathbf{o}$ and the intersection of the directed line $\mathbf{co}$ and $f_{\max}$. If the two polyhedra P and Q are separate, α is their *separating distance* along the direction $\mathbf{oc}$, which is the distance that Q needs to translate in the direction $\mathbf{oc}$ so that it is in external contact with P (Figure 7(a)). If P and Q overlap, α is then their *penetration distance* along $\mathbf{co}$, which is the distance that Q must move in $\mathbf{co}$ until it touches P externally. However, it is possible that a shorter distance can be taken by moving Q in the opposite direction $\mathbf{oc}$ to separate P and Q, i.e. the distance denoted by $\bar{\alpha}$ in Figure 7(b). By Corollary 2, $\bar{\alpha}$ is the distance between the origin and the intersection of the ray from $\mathbf{c}$ to the face $f_{\min}$. According to the definition of MDD in (1), we have

$$\mathrm{MDD}(P, Q, \mathbf{oc}) = \begin{cases} \alpha & \text{if } P \text{ and } Q \text{ are non-intersecting,} \\ \min\{\alpha, \bar{\alpha}\} & \text{otherwise.} \end{cases}$$

We will now establish the relationship between α and $d_{\max} = \mathrm{d}(f_{\max})$. Let $\mathbf{o} = -\mathbf{c}$ so that $\mathbf{c}$ is the centre of duality (Figure 8). Then $\mathbf{o}^*$ is the plane given by $-\mathbf{c}^T\mathbf{x} = 0$. The face $f_{\max} \equiv \mathbf{u}^T\mathbf{x} = 1$ corresponds to the vertex $f^*_{\max} = \mathbf{u}$ on M^* in $\mathbb{E}^{3*}$. We have

$$d_{\max} = -\frac{\mathbf{c}^T\mathbf{u} + 1}{\|\mathbf{c}\|}$$

Let α' be the shortest distance from the point $\mathbf{o}$ to the face $f_{\max}$ and θ be the angle between the line $\mathbf{co}$ and the normal vector of $f_{\max}$. Then

$$\begin{aligned} \alpha' &= -\frac{\mathbf{c}^T\mathbf{u} + 1}{\|\mathbf{u}\|} = \frac{d_{\max}\|\mathbf{c}\|}{\|\mathbf{u}\|}, \text{ and} \\ \cos\theta &= -\frac{\mathbf{c}^T\mathbf{u}}{\|\mathbf{c}\|\|\mathbf{u}\|} = \frac{d_{\max}\|\mathbf{c}\| + 1}{\|\mathbf{c}\|\|\mathbf{u}\|}. \end{aligned}$$

Hence,

$$\alpha = \frac{\alpha'}{\cos\theta} = \frac{d_{\max}\|\mathbf{c}\|^2}{d_{\max}\|\mathbf{c}\| + 1}.$$

Similarly, it can be shown that

$$\bar{\alpha} = \frac{d_{\min}\|\mathbf{c}\|^2}{d_{\min}\|\mathbf{c}\| + 1}.$$

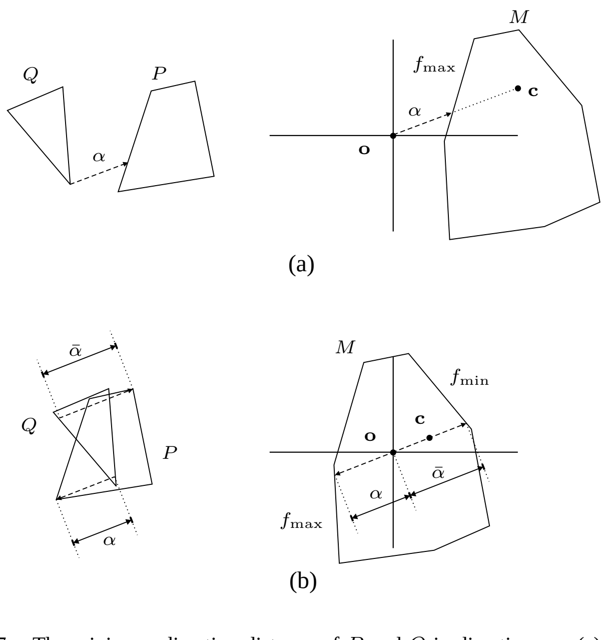
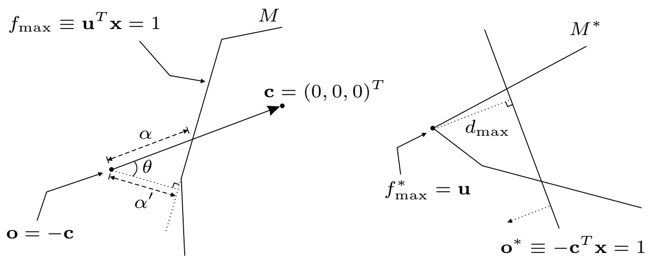


Fig. 7. The minimum direction distance of P and Q in direction $\mathbf{oc}$. (a) P and Q are separate, and Q needs to move along $\mathbf{oc}$ by a distance of α to be in contact with P; (b) P and Q intersect, and Q needs to move either in $\mathbf{co}$ by α or in $\mathbf{oc}$ by $\bar{\alpha}$ to be in external contact with P.

IV. The Algorithm

Given two convex polyhedra P and Q, and a direction $\mathbf{s} \in \mathbb{R}^3$, the steps in MDD-Dual for computing the minimum directional distance of P and Q in $\mathbf{s}$ are as follows:

Step 1: Determine the center of duality $\mathbf{c}$ in the interior of $M = P \oplus -Q$ such that $\mathbf{c} = \alpha\mathbf{s}$ for some nonzero constant $\alpha \in \mathbb{R}$. If $\mathbf{c}$ is not defined, report the $\mathrm{MDD}(P, Q, \mathbf{s})$, else go to Step 2.



$f_{\max} \equiv \mathbf{u}^T\mathbf{x} = 1$
M
$\mathbf{c} = (0,0,0)^T$
α
θ
α'
$\mathbf{o} = -\mathbf{c}$
M^*
$d_{\max}$
$f^*_{\max} = \mathbf{u}$
$\mathbf{o}^* \equiv -\mathbf{c}^T\mathbf{x} = 1$

Fig. 8. The minimum directional distance α in the primal space and the maximum signed distance $d_{\max}$ in the dual space.

Step 2: Obtain $d_{\max}$, the maximum signed distance, by searching for the optimal face $f_{\max}$ among all faces in $\mathcal{F}_{\text{fv}}$, $\mathcal{F}_{\text{vf}}$ and $\mathcal{F}_{\text{ee}}$ individually. If $d_{\max} < 0$, obtain also $d_{\min}$, the minimum signed distance by searching for the face $f_{\min}$ among all faces in $\mathcal{F}_{\text{fv}}$, $\mathcal{F}_{\text{vf}}$ and $\mathcal{F}_{\text{ee}}$ individually.

Step 3: Compute $\text{MDD}(P, Q, \mathbf{s})$ from $d_{\max}$ as described in section III-C.

Steps 1 and 2 will be described in details in the subsequent sections.

A. Determining the center of duality $\mathbf{c}$

In this step, our goal is to determine a point $\mathbf{c}$ in the interior of $M = P \oplus -Q$ that will be used as the centre of duality. As explained in sections II-D and III-C, it is necessary that $\mathbf{c} = k\mathbf{s}$ for some nonzero real constant α in order to obtain the MDD in the direction $\mathbf{s}$; the condition $k \neq 0$ is to ensure that $\mathbf{c}$ is not the origin.

We first obtain two sets of points $\dot{P}$ and $\dot{Q}$ by applying an orthographic projection along $\mathbf{s}$ of all vertices of P and Q to a plane $\mathcal{H}$ normal to $\mathbf{s}$. This projection can be done in $\mathcal{O}(n)$ time (Figure 9(a)) where n is the total number of vertices of P and Q. The next step is to construct the convex hull, $CH(\dot{P})$

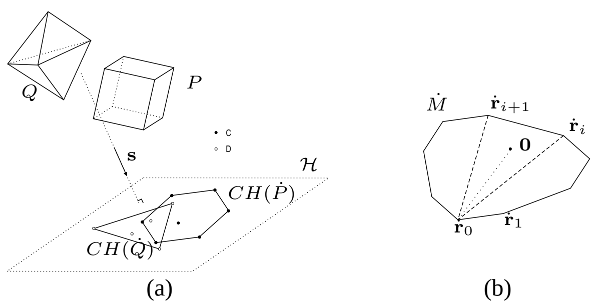


Fig. 9. (a) Orthographic projection of P and Q along $\mathbf{s}$ to a plane $\mathcal{H}$ normal to $\mathbf{s}$; and (b) the Minkowski difference $\dot{M}$ of $CH(\dot{P})$ and $CH(\dot{Q})$ with the triangle $\triangle\dot{\mathbf{r}}_0\dot{\mathbf{r}}_i\dot{\mathbf{r}}_{i+1}$ containing the origin $\mathbf{0}$.

and $CH(\dot{Q})$, of the points $\dot{\mathcal{P}}$ and $\dot{\mathcal{Q}}$, respectively, which can also be done in $\mathcal{O}(n)$ time since the boundary vertices of $CH(\dot{P})$ and $CH(\dot{Q})$ are the silhouette vertices of P and Q as viewed along $\mathbf{s}$. We then build the Minkowski difference $\dot{M} = CH(\dot{P}) \oplus -CH(\dot{Q})$ which takes $\mathcal{O}(n)$ time. Let $\dot{\mathbf{r}}_j = \dot{\mathbf{p}}_j - \dot{\mathbf{q}}_j$, be the vertices of $\dot{M}$ in anticlockwise order (Figure 9(b)). We may then quickly determine in $\mathcal{O}(n)$ time whether $\mathbf{0}$ is in $\dot{M}$ by examining the triangles $\triangle\dot{\mathbf{r}}_0\dot{\mathbf{r}}_j\dot{\mathbf{r}}_{j+1}$, $j = 1, \ldots, l-2$ where l is the number of vertices of $\dot{M}$. We have $\mathbf{0} \in \dot{M}$ if and only if $\mathbf{0} \in \triangle\dot{\mathbf{r}}_0\dot{\mathbf{r}}_i\dot{\mathbf{r}}_{i+1}$, for some i. There are now three cases to consider; case 1 and 2 will report the MDD between P and Q, while case 3 will determine a point $\mathbf{c}$ as the centre of duality and the algorithm MDD-DUAL will proceed with step 2.

Case 1: $\mathbf{0} \notin \dot{M}$. Here, P and Q do not have external contact no matter how far Q is moved along $\mathbf{s}$ and hence the MDD of P and Q is undefined.

Case 2: $\mathbf{0}$ is on the boundary of $\dot{M}$. In this case, P and Q either have no contact or have only external contact no matter how far Q is moved along $\mathbf{s}$; also, the ray shooting from $\mathbf{o}$ in $\mathbf{s}$ touches M at the boundary. Our aim would be to find a face on M that the ray touches. Assume that $\mathbf{0}$ lies on an edge $\dot{\mathbf{r}}_i\dot{\mathbf{r}}_{i+1}$ of $\dot{M}$, with $\dot{\mathbf{r}}_i = \dot{\mathbf{p}}_i - \dot{\mathbf{q}}_i$ and $\dot{\mathbf{r}}_{i+1} = \dot{\mathbf{p}}_{i+1} - \dot{\mathbf{q}}_{i+1}$. Then, $CH(\dot{P})$ and $CH(\dot{Q})$ must contact at the features $(\dot{\mathbf{p}}_i, \dot{\mathbf{p}}_{i+1})$ and $(\dot{\mathbf{q}}_i, \dot{\mathbf{q}}_{i+1})$, which may be either an edge or a vertex. Let ϕ_P and ϕ_Q be the features on P and Q whose projection to $\mathcal{H}$ is a subset of $(\dot{\mathbf{p}}_i, \dot{\mathbf{p}}_{i+1})$ and $(\dot{\mathbf{q}}_i, \dot{\mathbf{q}}_{i+1})$, respectively. The features ϕ_P and ϕ_Q must contain the vertices $\mathbf{p}_i, \mathbf{p}_{i+1}$ and $\mathbf{q}_i, \mathbf{q}_{i+1}$, respectively, and they must be where P and Q have contacts when Q moves along $\mathbf{s}$. Now, depending on the nature (i.e. vertex, edge or face) of ϕ_P and ϕ_Q, we construct either an $\mathcal{F}_{\text{fv}}$, $\mathcal{F}_{\text{vf}}$ or an $\mathcal{F}_{\text{ee}}$ face F on M. The distance from $\mathbf{o}$ to F along $\mathbf{s}$ is the MDD of P and Q.

Case 3: $\mathbf{0}$ is in the interior of some triangle $\triangle\dot{\mathbf{r}}_0\dot{\mathbf{r}}_i\dot{\mathbf{r}}_{i+1}$. Then we have two sets of vertices $S_P = \{\dot{\mathbf{p}}_0, \dot{\mathbf{p}}_i, \dot{\mathbf{p}}_{i+1}\}$ and $S_Q = \{\dot{\mathbf{q}}_0, \dot{\mathbf{q}}_i, \dot{\mathbf{q}}_{i+1}\}$ forming the triangle. We want to make sure that at least one of the two sets contain distinct vertices. If this is not the case, we choose $\dot{\mathbf{r}}_k = \dot{\mathbf{p}}_k - \dot{\mathbf{q}}_k$ so that $\dot{\mathbf{p}}_k \notin S_P$ and $\dot{\mathbf{q}}_k \notin S_Q$. The origin $\mathbf{0}$ must be in either $\triangle\dot{\mathbf{r}}_0\dot{\mathbf{r}}_i\dot{\mathbf{r}}_k$ or $\triangle\dot{\mathbf{r}}_i\dot{\mathbf{r}}_{i+1}\dot{\mathbf{r}}_k$, if $k > i+1$; or in either $\triangle\dot{\mathbf{r}}_0\dot{\mathbf{r}}_k\dot{\mathbf{r}}_{i+1}$ or $\triangle\dot{\mathbf{r}}_i\dot{\mathbf{r}}_{i+1}\dot{\mathbf{r}}_k$, if $k < i$. It can be shown that in any case, we could obtain a triangle $\triangle\dot{\mathbf{r}}_{k_0}\dot{\mathbf{r}}_{k_1}\dot{\mathbf{r}}_{k_2}$ such that at least one of the corresponding two sets S_P and S_Q must contain distinct vertices.

Let $w_j, j = 0, 1, 2$ be the barycentric coordinates of $\mathbf{0}$ with respect to $\dot{\mathbf{r}}_{k_0}, \dot{\mathbf{r}}_{k_1}, \dot{\mathbf{r}}_{k_2}$. Hence, $w_i > 0$ and $\sum w_i = 1$. Then, we have

$$\mathbf{0} = \sum w_j \dot{\mathbf{r}}_{k_j} = \sum w_j \dot{\mathbf{p}}_{k_j} - \sum w_j \dot{\mathbf{q}}_{k_j} = \dot{\mathbf{p}} - \dot{\mathbf{q}},$$

where $\dot{\mathbf{p}} = \sum w_j \dot{\mathbf{p}}_{k_j}$ and $\dot{\mathbf{q}} = \sum w_j \dot{\mathbf{q}}_{k_j}$. Since the orthographic projection of P and Q to the plane $\mathcal{H}$ is an affine transformation that preserves the ratio of area and therefore barycentric coordinates, $\dot{\mathbf{p}}$ and $\dot{\mathbf{q}}$ are the projected images of some $\mathbf{p}$ and $\mathbf{q}$, respectively, where $\mathbf{p} = \sum w_j \mathbf{p}_{k_j}$ and $\mathbf{q} = \sum w_j \mathbf{q}_{k_j}$, $\mathbf{p}_{k_j} \in \mathcal{V}_P$ and $\mathbf{q}_{k_j} \in \mathcal{V}_Q$ are the vertices projected to $\dot{\mathbf{p}}_{k_j}$ and $\dot{\mathbf{q}}_{k_j}$ on $\mathcal{H}$, respectively. Since $w_j > 0$ and $\sum w_j = 1$, we have $\mathbf{p} \in P$ and $\mathbf{q} \in Q$. Also as $\dot{\mathbf{p}} = \dot{\mathbf{q}}$, we obtain $\mathbf{c} = \mathbf{p} - \mathbf{q} = \alpha\mathbf{s}$, for some constant α. The following procedures ensure that $\mathbf{c}$ is not the origin and $\mathbf{c}$ is in the interior of M, which make use of the fact that at least one of S_P or S_Q contains distinct vertices. Without loss of generality, suppose that $S_P = \{\dot{\mathbf{p}}_{k_j}\}$ contains distinct vertices. We form a tetrahedron $\mathcal{T}$ with vertices $\mathbf{p}_{k_0}, \mathbf{p}_{k_1}, \mathbf{p}_{k_2}$ and $\mathbf{p}_t$, where $\mathbf{p}_t$ is any vertex in P not coplanar to the three vertices $\mathbf{p}_{k_j}$. It is then easy to choose a new point $\mathbf{p}$ in the interior of $\mathcal{T}$ (and hence the interior of P) that has the same projection $\dot{\mathbf{p}}$ on $\mathcal{H}$, and that $\mathbf{p} \neq \mathbf{q}$. Finally, $\mathbf{c} = \mathbf{p} - \mathbf{q} = \alpha \mathbf{s}$, for some constant $\alpha \neq 0$ and $\mathbf{c}$ is in the interior of M.

B. Computing the maximum signed distance $d_{\max}$

A brute-force search for the face $f_{\max}$ that attains the maximum signed distance $d_{\max}$ is to first construct the Minkowski difference $M \equiv P \oplus -Q$ and find the face with $d_{\max}$. However, the time complexity of constructing M is known to be $\mathcal{O}(mn)$ in the worst case, where m and n are the number of vertices of P and Q, respectively. Moreover, it is inefficient to traverse the faces on M by advancing to an immediate neighbour at each step. We therefore breaks down the search for $f_{\max}$ in three successive phases, each within the subsets $\mathcal{F}_{\text{fv}}$, $\mathcal{F}_{\text{vf}}$ and $\mathcal{F}_{\text{ee}}$ of M. This allows a quicker leap over the faces on M and therefore the search reaches $f_{\max}$ more rapidly. Also, the number of faces on M that needs to be constructed on M can be greatly reduced. The procedure for each search phases will be described in the following subsections. The search for $d_{\min}$ is the same except that at every step, we look for a face with smaller signed distance; therefore, we do not repeat the corresponding procedures here. We will also leave some implementation details to section VI, so that the core parts of the procedures can be distinctly followed.

The following pseudocode gives an overview of how $d_{\max}$ is obtained:

```
MaxSignedDistance($P$, $Q$)
    $d_{\text{fv}} \leftarrow$ Search-FV
    $d_{\text{vf}} \leftarrow$ Search-VF
    $d_{\max} \leftarrow$ Search-EE($d_{\text{fv}}, d_{\text{vf}}$)
    return $d_{\max}$
```

1) Search-FV*:* This procedure is to search for a face with the maximum signed distance among all faces in $\mathcal{F}_{\text{fv}}$. Let f_0 denote the initial face with normal vector $\hat{\mathbf{n}}(f_0)$ in the search. The initial face f_0 may be chosen from $\mathcal{F}_P$ at random or to give a better performance, we may use a heuristic selection in a preprocessing step so that the face $F(f_0, \mathrm{s}_{-Q}(f_0)) \in \mathcal{F}_{\text{fv}}$ is close to $f_{\max}$ on M. The selection for f_0 will be discussed in detailed in section VI-A.

Starting from f_0, the search in Search-FV considers the neighbouring faces of the current face and advances to one which has the local maximum signed distance. The neighbouring (or adjacent) faces are those faces incident to the vertices of the current face in P. Two faces adjacent in P may not constitute adjacent faces in M. In this way, a gain could be obtained in advancing faces in the search based on their adjacency in P.

The procedure is described in the following pseudocode. The function SignedDistance-FV(f) constructs a face $F\big(f, \mathrm{s}_{-Q}(f)\big) \in \mathcal{F}_{\text{fv}}$ and computes its signed distance using 2. The determination of the supporting vertex of $-Q$ for a face f is accelerated using the hierarchical representation of a polyhedron presented in [13].

```
Search-FV
    $d_{\text{fv}}$ = SignedDistance-FV($f_0$)
    For each iteration $i$
        For each of the $n$ faces $f_i^j$, $j = 0, \dots, n-1$,
        that are adjacent to $f_i$ in $P$
            $d_i^j \leftarrow$ SignedDistance-FV($f_i^j$).
        If $d_{\text{fv}} < d_i^k$, where $d_i^k = \max\{d_i^j\}$
            $d_{\text{fv}} \leftarrow d_i,\ f_{i+1} \leftarrow f_i^k$.
        Otherwise,
            Return $d_{\text{fv}}$.
```

We now prove the correctness of Search-FV.

Theorem 3: The signed distance d_{fv} computed by Search-FV, is the maximum signed distance among all faces in $\mathcal{F}_{\text{fv}}$, i.e. $d_{\text{fv}} = \max\{\mathrm{d}(f) \mid f \in \mathcal{F}_{\text{fv}}\}$.

Proof: Consider the set of faces $\mathcal{F}_{\text{fv}}$ and its corresponding dual $\mathcal{F}^*_{\text{fv}}$. If for every two faces $f_0, f_1 \in \mathcal{F}_P$ that share an edge, we connect $F^*(f_0, \mathrm{s}_{-Q}(f_0))$ and $F^*(f_1, \mathrm{s}_{-Q}(f_1))$ by an edge, then by the construction of M and the properties of duality, we know that the point set $\mathcal{F}^*_{\text{fv}}$ and the augmented edges form a polyhedron W^*. Since $\mathcal{F}^*_{\text{fv}} \subset \mathcal{V}_{M^*}$ and M^* is convex, W^* must be convex too. Now, Search-FV searches locally for a vertex in W^* that attains the largest signed distance to the plane $\mathbf{o}^*$. The search path also follows the adjacency of the faces in P and therefore is along the edges of W^*. As W^* is convex, the search will eventually stop at a dual vertex f^* of a face $f \in \mathcal{F}_{\text{fv}}$ attaining the local maximum signed distance, which is also the global maximum signed distance among all dual vertices in $\mathcal{F}^*_{\text{fv}}$ (corresponding to the face set $\mathcal{F}_{\text{fv}}$). ∎

2) Search-VF*:* This procedure computes the maximum signed distance among all faces in $\mathcal{F}_{\text{vf}}$. Upon completion of Search-FV, we obtain a face $f = F\big(f_p, \mathrm{s}_{-Q}(f_p)\big) \in \mathcal{F}_{\text{fv}}$ such that $d_{\text{fv}} = \mathrm{d}(f)$. The face f is supposed to be closest to the optimal face $f_{\max}$ among all faces $\mathcal{F}_{\text{fv}}$, and it should give a good starting point for subsequent search. Hence, the initial face for Search-VF can be chosen as a face f_0 that is incident at $\mathrm{s}_{-Q}(f_p)$ in $-Q$. The search then proceeds in a similar way as Search-FV by interchanging the role of P and $-Q$; the pseudocode is hence omitted for brevity. We also have the following theorem for the correctness of Search-VF.

Theorem 4: The signed distance d_{vf} computed by Search-VF, is the maximum signed distance among all faces in $\mathcal{F}_{\text{vf}}$, i.e. $d_{\text{vf}} = \max\{\mathrm{d}(f) \mid f \in \mathcal{F}_{\text{vf}}\}$.

Proof: Similar to the proof of Theorem 3, by considering the symmetry of P and $-Q$ in the two procedures Search-FV and Search-VF. ∎

3) Search-EE*:* The previous two procedures Search-FV and Search-VF determine the maximum signed distance d_{fv} and d_{vf} among all faces in the set $\mathcal{F}_{\text{fv}}$ and $\mathcal{F}_{\text{vf}}$, respectively. The next step is to search for the remaining faces in $\mathcal{F}_{\text{ee}}$, starting from the face $f \in \mathcal{F}_{\text{fv}} \cup \mathcal{F}_{\text{vf}}$ that attains the signed distance $\max\{d_{\text{fv}}, d_{\text{vf}}\}$.

Let e_p and e_q be edges in $\mathcal{E}_P$ and $\mathcal{E}_{-Q}$, respectively. As mentioned in section II-C, if the Gaussian images of e_p and e_q intersect on S^2, a face $F(e_p, e_q) \in \mathcal{F}_{\text{ee}}$ will be formed. We shall describe in details how to determine whether two arcs intersect in section VI-B.

```
SEARCH-EE
    $\mathrm{d_{ee}} \leftarrow \max\{d_{\mathrm{fv}}, d_{\mathrm{vf}}\}$.
    $f_m \leftarrow$ the face in $\mathcal{F}_{\mathrm{fv}} \cup \mathcal{F}_{\mathrm{vf}}$ that attains $\mathrm{d_{ee}}$.
    $\mathcal{FS}_0 \leftarrow$ all possible face $F(e_p, e_q)$, where
        $e_p$ is an edge incident to a vertex of $f_p$
        and $e_q$ is an edge incident to $\mathrm{s}_{-Q}(f_p)$,
        if $f_m = F\big(f_p, \mathrm{s}_{-Q}(f_p)\big)$, or
        $e_p$ is an edge incident to $\mathrm{s}_P(f_q)$
        and $e_q$ is an edge incident to a vertex of $f_q$,
        if $f_m = F\big(\mathrm{s}_P(f_q), f_q\big)$.
    For each iteration $i = 0, 1, 2, \ldots$
        Let $\hat{f}_i = F(\hat{e}_p, \hat{e}_q) \in \mathcal{FS}_i$ be the face such that
        $\mathrm{d}(\hat{f}_i) = \max\{\mathrm{d}(f) \mid f \in \mathcal{FS}_i\}$
        If $d_{\mathrm{ee}} < \mathrm{d}(\hat{f}_i)$
            $d_{\mathrm{ee}} \leftarrow \mathrm{d}(\hat{f}_i)$
            $\mathcal{FS}_{i+1} \leftarrow$ all possible face $F(e_p, e_q)$, where
                $e_p$ is an edge incident to an end vertex of $\hat{e}_p$,
                $e_q$ is an edge incident to an end vertex of $\hat{e}_q$
        Otherwise,
            Return $d_{\mathrm{ee}}$.
```

Lemma 5: If the optimal face, $f_{\max}$, is in $\mathcal{F}_{\mathrm{fv}} \cup \mathcal{F}_{\mathrm{vf}}$, then SEARCH-EE returns $\mathrm{d}(f_{\max}) = d_{\max}$.

Proof: Since $f_{\max} \in \mathcal{F}_{\mathrm{fv}} \cup \mathcal{F}_{\mathrm{vf}}$, $f_{\max}$ must have the maximum signed distance among all faces in $\mathcal{F}_{\mathrm{fv}} \cup \mathcal{F}_{\mathrm{vf}}$. Hence, either d_{fv} or d_{vf} returned by SEARCH-FV or SEARCH-VF equals $d_{\max}$, i.e. $d_{\max} = \max\{d_{\mathrm{fv}}, d_{\mathrm{vf}}\}$. Also, no face in $\mathcal{F}_{\mathrm{ee}}$ will have a larger signed distance than $\mathrm{d}_{\max}$. Therefore, by the flow of SEARCH-EE, the maximum signed distance $d_{\max}$ is returned. ■

Lemma 6: If the optimal face, $f_{\max}$, is in $\mathcal{F}_{\mathrm{ee}}$, then the initial face set $\mathcal{FS}_0$ in SEARCH-EE must contain at least one face $f_e \in \mathcal{F}_{\mathrm{ee}}$ such that $\mathrm{d}(f_e) > \max\{d_{\mathrm{fv}}, d_{\mathrm{vf}}\}$.

Proof: Without loss of generality, let us assume that $f_m = F\big(f_p, \mathrm{s}_{-Q}(f_p)\big)$ is the starting face in SEARCH-EE attaining the signed distance $\max\{d_{\mathrm{fv}}, d_{\mathrm{vf}}\}$, where $f_p \in \mathcal{F}_P$, $\mathrm{s}_{-Q}(f_p) \in \mathcal{V}_{-Q}$. The neighbouring faces of f_m on M are those faces that are incident to the vertices of f_m. Consider the Gaussian images $G(M)$, $G(P)$ and $G(-Q)$. Let R_M^i be the neighbouring regions of $G(f_m)$ in $G(M)$, R_P^j be the neighbouring regions of $G(f_p)$ in $G(P)$ and R_Q be the region $G\big(\mathrm{s}_{-Q}(f_p)\big)$ in $G(-Q)$. Hence, the neighbouring faces of f_m correspond to those points defining the regions R_M^i (Fig. 10).

Note that $G(f_m)$ and $G(f_p)$ are the same point on the Gaussian sphere $\mathcal{S}^2$. Since $G(f_p)$ lies inside the region R_Q, R_M^i must be the intersection of R_P^j and R_Q. Therefore, the points of R_M^i must be either (A) the points of R_P^j or the points of R_Q, or (B) the intersections of an arc of R_P^j with an arc of R_Q (Fig. 10). The latter set of points (B) correspond to the face set $\mathcal{FS}_0$ in SEARCH-EE. If $\mathcal{FS}_0$ is empty, the neighbouring faces of f_m can only be faces corresponding to points in set (A), i.e. the faces in $\mathcal{F}_{\mathrm{fv}} \cup \mathcal{F}_{\mathrm{vf}}$. Then f_m has the maximum signed distance among all its neighbours, since $\mathrm{d}(f_m) = \max\{d_{\mathrm{fv}}, d_{\mathrm{vf}}\}$. On the other hand, if $\mathcal{FS}_0$ is non-

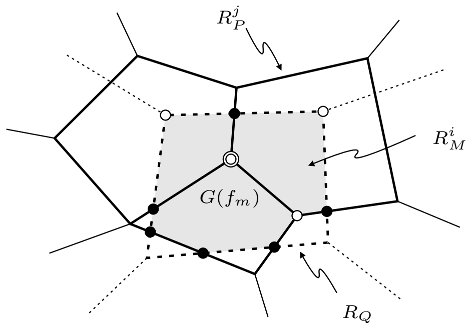


Fig. 10. The Gaussian map of M showing the neighbouring faces of $f_m = F\big(f_p, \mathrm{s}_{-Q}(f_p)\big)$. Solid lines and dotted lines are the arcs of $G(P)$ and $G(-Q)$, respectively. The boundaries of neighbouring regions of $G(f_p)$ in $G(P)$ (R_P^j) are in thick solid lines; the boundaries of $G\big(\mathrm{s}_{-Q}(f_p)\big)$ in $G(-Q)$ (R_Q) are in thick dotted lines; The neighbouring regions of $G(f_m)$ in $G(M)$ (R_M^i) are in grey. The neighbouring faces of f_m correspond to the black (FV- or VF-types) and white (EE-type) points.

empty, and all faces $f_e \in \mathcal{FS}_0$ are such that $\mathrm{d}(f_e) < \mathrm{d}(f_m)$. Again, f_m has the maximum signed distance among all its neighbours. In both cases, it implies that f_m is the optimal face attaining the global maximum signed distance, i.e. $\mathrm{d}(f_m) = d_{\max}$. However, this contradicts that $d_{\max}$ is attained by a face in $\mathcal{F}_{\mathrm{ee}}$. Hence, there must be at least a face $f_e \in \mathcal{FS}_0$ such that $\mathrm{d}(f_e) > \max\{d_{\mathrm{fv}}, d_{\mathrm{vf}}\}$. ■

Lemma 7: The face set $\mathcal{FS}_{i+1}$ in SEARCH-EE contains all EE-type faces that are adjacent to the face $\hat{f}_i = F(\hat{e}_p, \hat{e}_q)$ where $\hat{e}_p \in \mathcal{E}_P$ and $\hat{e}_q \in \mathcal{E}_{-Q}$.

Proof: The neighbouring faces of $\hat{f}_i$ on M are those faces incident to the vertices of $\hat{f}_i$. Consider the Gaussian images $G(M)$, $G(P)$ and $G(-Q)$. The point $G(\hat{f}_i)$ is the intersection of the two arcs $G(\hat{e}_p)$ and $G(\hat{e}_q)$ (Fig. 11). Let R_M^i be the

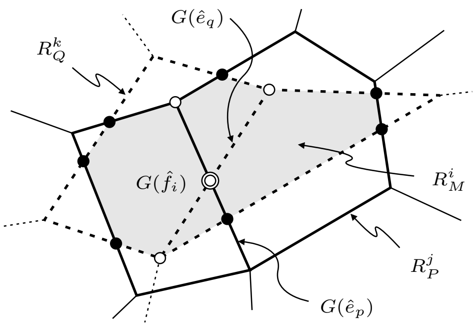


Fig. 11. The Gaussian map of M showing the neighbouring faces of $\hat{f}_i = F(\hat{e}_p, \hat{e}_q)$. Solid lines and dotted lines are the arcs of $G(P)$ and $G(-Q)$, respectively. The boundaries of the neighbouring regions of $G(\hat{e}_p)$ in $G(P)$ (R_P^j) are in thick solid lines; the boundaries of the neighbouring regions of $G(\hat{e}_q)$ in $G(-Q)$ (R_Q^k) are in thick dotted lines; The neighbouring regions of $G(\hat{f}_i)$ in $G(M)$ (R_M^i) are in grey. The neighbouring faces of $\hat{f}_i$ correspond to the black (FV- or VF-types) and white (EE-type) points.

neighbouring regions of $G(\hat{f}_i)$, R_P^j be the two neighbouring regions of $G(\hat{e}_p)$ and R_Q^k be the two neighbouring regions of $G(\hat{e}_q)$. The regions R_M^i must be the intersection of R_P^j and R_Q^k; hence, the points defining R_M^i must be the intersections of the arcs of R_P^j and R_Q^k, and also some points from R_P^j, R_Q^k. The faces in $\mathcal{FS}_{i+1}$ in SEARCH-EE corresponds to the intersections of the arcs of R_P^j and R_Q^k, which are all the EE-type neighbours of $\hat{f}_i$. ■

Theorem 8: The signed distance $d_{\rm ee}$ computed by SEARCH-EE is the maximum signed distance among all faces in $\mathcal{F}_M$, i.e. $d_{\rm ee} = d_{\max} = \max\{{\rm d}(f) \mid f \in \mathcal{F}_M\}$.

Proof: The optimal face $f_{\max}$ attaining the maximum signed distance $d_{\max}$ must be in either $\mathcal{F}_{\rm fv} \cup \mathcal{F}_{\rm vf}$ or $\mathcal{F}_{\rm ee}$. If $f_{\max} \in \mathcal{F}_{\rm fv} \cup \mathcal{F}_{\rm vf}$, by Lemma 5, SEARCH-EE computes $d_{\max}$.

Suppose $f_{\max} \in \mathcal{F}_{\rm ee}$. Lemma 6 guarantees that ${\rm d}(\hat{f}_0) = \max\{{\rm d}(f) \mid f \in \mathcal{FS}_0 > \max\{d_{\rm fv}, d_{\rm vf}\}$. Hence, the iteration in SEARCH-EE will proceed. For each iteration $i > 0$, $d_{\rm ee}$ is the signed distance of the current face, and $\mathcal{FS}_i$ is the set of all EE-type faces neighbouring to the current face (by Lemma 7). The iteration stops when the signed distance of the current face is the maximum among all its neighbouring EE-type faces. Since $d_{\rm ee}$ is increasing for each iteration, $d_{\rm ee} > \max\{d_{\rm fv}, d_{\rm vf}\}$ which means that the signed distance of the current face is also the maximum among all its neighbouring FV- and VF-type faces. Hence, SEARCH-EE stops at a face in $\mathcal{F}_{\rm ee}$ with a local maximum signed distance $d_{\rm ee}$ among all its neighbouring faces in M, and therefore, $d_{\rm ee} = d_{\max}$. ■

V. Collision Detection of Two Convex Polyhedra

To solve the collision detection problem of two convex polyhedra P and Q, we only need to tell whether P and Q separate or not. The choice for the centre of duality $\mathbf{c}$ is much relaxed and it only requires that $\mathbf{c} \in M = P \oplus -Q$ and that $\mathbf{c}$ is not the origin. In this case, we store two distinct interior points, $\mathbf{p}_0, \mathbf{p}_1 \in P$ and $\mathbf{q}_0, \mathbf{q}_1 \in Q$. The vector differences, $\mathbf{p}_i - \mathbf{q}_j$, of these four interior points give rise to four distinct interior points in M, from which it is always possible to obtain an interior point of M which is not the origin.

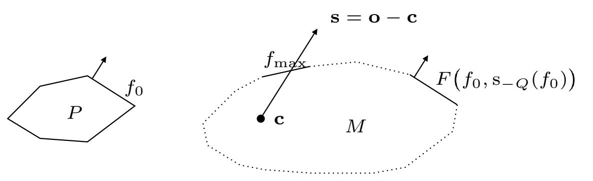
By the condition for the separation of two convex polyhedra in section III-A, we need only to determine whether the plane o^* intersects M^* in the dual space, or equivalently, whether there is a face f with signed distance ${\rm d}(f) > 0$. Hence, an early escape from the search in the procedures SEARCH-FV, SEARCH-VF and SEARCH-EE can be enabled by stopping the search of $f_{\max}$ whenever the signed distance of the current face is positive, in which case the two convex polyhedra P and Q are separate. The remaining situations are $d(f_{\max}) = 0$, which corresponds to P and Q touching each other; or $d(f_{\max}) < 0$, which means that P and Q intersect.

VI. Implementation Issues

In this section, we shall highlight several important issues in implementing MDD-DUAL .

A. To obtain the initial face in SEARCH-FV

In section IV-B.1, an initial face $f_0 \in \mathcal{F}_P$ is to be selected as the starting point for the search of $f_{\max}$ in SEARCH-FV. The face f_0 should be such that $F\big(f_0, {\rm s}_{-Q}(f_0)\big) \in \mathcal{F}_{\rm fv}$ is as close to $f_{\max}$ as possible. Now, $f_{\max}$ is the face on M where the ray $\mathbf{co}$ intersects and therefore $f_{\max}$ must be front-facing with respect to $\mathbf{s} = \mathbf{o} - \mathbf{c}$ such that $\hat{\mathbf{n}}(f_0) \cdot \mathbf{s} > 0$. We may then take the face $f_0 \in \mathcal{F}_P$ such that $\hat{\mathbf{n}}(f_0) \cdot \mathbf{s}$ is the greatest among all faces in $\mathcal{F}_P$. It is worth noting that when M is flat and elongated, $F\big(f_0, {\rm s}_{-Q}(f_0)\big)$ may not be as close to $f_{\max}$ as shown in (Figure 12). Nevertheless, this heuristic scheme in selecting f_0 can still efficiently eliminate most back-facing faces f in M with respect to $\mathbf{s}$ where $\hat{\mathbf{n}}(f) \cdot \mathbf{s} < 0$.



$\mathbf{s} = \mathbf{o} - \mathbf{c}$
$f_{\max}$
$F(f_0, {\rm s}_{-Q}(f_0))$
f_0
P
$\mathbf{c}$
M

Fig. 12. If the shape of M is elongated, the face formed by the initial face f_0 chosen may not be close to $f_{\max}$.

B. To decide whether two arcs on S^2 intersect

In procedure SEARCH-EE, one operation is to decide whether two edges $e_p \in \mathcal{E}_P$ and $e_q \in \mathcal{E}_{-Q}$ form a face $F(e_p, e_q) \in \mathcal{F}_{\rm ee}$. This is done by checking whether the two arcs $G(e_p)$ and $G(e_q)$ intersect on the Gaussian sphere $\mathcal{S}^2$. Let $\mathbf{a}, \mathbf{b}$ be the end points of $G(e_p)$, $\mathbf{c}, \mathbf{d}$ be the end points of $G(e_q)$ and $\mathbf{o}$ be the centre of $\mathcal{S}^2$ (Figure 13). The arcs $G(e_p)$ and $G(e_q)$ intersect if and only if (1) $\mathbf{c}, \mathbf{d}$ are on different sides of the plane $\mathbf{oba}$; (2) $\mathbf{a}, \mathbf{b}$ are on different sides of the plane $\mathbf{ocd}$; and (3) $\mathbf{a}, \mathbf{b}, \mathbf{c}, \mathbf{d}$ are on the same hemisphere.

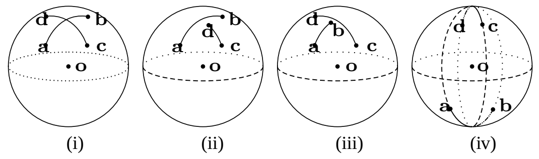


Fig. 13. Determining whether two arcs intersect on $\mathcal{S}^2$. Arcs intersect in (i). No intersection between arcs where (ii) only condition (1); (iii) only condition (2) and (iv) only condition (3) is violated.

Consider the *signed volume*, $|\mathbf{cba}| = \det[\ \mathbf{c}\quad \mathbf{b}\quad \mathbf{a}\ ]$, of a parallelepiped spanned by three vectors $\mathbf{a}, \mathbf{b}, \mathbf{c}$. The quantities $|\mathbf{cba}|$ and $|\mathbf{dba}|$ are of different signs if $\mathbf{c}$ and $\mathbf{d}$ are at opposite sides of the plane $\mathbf{oba}$. Now, the above three conditions can be formulated as (1) $|\mathbf{cba}| \times |\mathbf{dba}| < 0$; (2) $|\mathbf{adc}| \times |\mathbf{bdc}| < 0$; and (3) $|\mathbf{acb}| \times |\mathbf{dcb}| > 0$. The last inequality holds when $\mathbf{a}$ and $\mathbf{d}$ lie on the same side of the plane $\mathbf{ocb}$ which means that all four points will be on the same hemisphere defined by $\mathbf{ocb}$. Note that only the quantities $|\mathbf{cba}|$, $|\mathbf{dba}|$, $|\mathbf{adc}|$ and $|\mathbf{bdc}|$ are to be computed, since $|\mathbf{acb}| = |\mathbf{cba}|$ and $|\mathbf{dcb}| = |\mathbf{bdc}|$.

C. Span of Faces with the Same Normal Direction

Throughout our discussion of the algorithm MDD-DUAL , we made an assumption that all faces on M have distinct normal directions. However, this is not always true for convex polyhedra with arbitrary mesh structures. Therefore, whenever it is required to perform any operations on the current face in the searching procedures (e.g. locating the adjacent faces, etc.), we will have to augment the current face to include also its neighbouring span of faces with the same normal direction.

D. Avoiding Repetitive Visits to a Face

It is important to avoid unnecessary computations for a face which is visited previously in the searching procedures and is known to be non-optimal. We use a hash table to record the visited faces in each procedure so that these faces can be skipped efficiently in MDD-DUAL .

E. Frame Coherence

When the two polyhedra P and Q assume continuous motion from frame to frame, MDD-DUAL may also exploit the temporal or frame coherence. At each time frame, we compute $f_{\max}$ which is either in $\mathcal{F}_{\text{fv}}, \mathcal{F}_{\text{vf}}$ or $\mathcal{F}_{\text{ee}}$. In any case, we can determine quickly a face $f = F\big(f_0, \mathrm{s}_{-Q}(f_0)\big) \in \mathcal{F}_{\text{fv}}$ that is as close to $f_{\max}$ as possible. Since the position and orientation of P and Q will have little changes for consecutive frames, the new optimal face should also be close to $f_{\max}$ and the use of f_0 as the initial face for SEARCH-FV will lead us to the new optimal face more rapidly.

VII. PERFORMANCE

We have implemented MDD-DUAL in C++ and the experiments described in this section are carried out on a PC equipped with a Pentium III 3 GHz CPU and 1GB memory. A set of 6 convex polyhedra are used (the name and the number of vertices of the polyhedra are given in the brackets): a truncated elliptic cone (P_1 – 20), a truncated elliptic cylinder (P_2 – 50), two ellipsoids (P_3 – 200, P_4 – 500), the convex hull of a random point set in a cube (P_5 – 100), and the volume of revolution of a convex profile curve (P_6 – 200). The sizes of the polyhedra are all within a sphere of radius 5. The cone and the cylinder are in the aspect $a : b : h = 1 : 2 : 4$, where a, b are the sizes of the base ellipse and h is the height. The size of the ellipsoids are in $a : b : c = 2 : 2 : 5$, where a, b and c are the length of the three major axes.

A total of 10 pairwise MDD calculations are carried out: (P_1, P_2), (P_1, P_3), (P_4, P_5), (P_4, P_6), and $(P_i, P_i), i = 1, \dots, 6$. For each pair of objects P_i and P_j, P_j assumes 40 random orientations and for each orientation, we move P_j so that the shortest distance P_i and P_j ranges from -1.5 to 1.5 in 21 samples, in which 10 samples correspond to where P_i and P_j intersect, 1 sample corresponds to touching, and 10 samples correspond to separation. Also, for each fixed shortest distance between P_i and P_j with a random orientation, we compute their MDD along 40 random directions using MDD-DUAL . It means that, for each pair of convex polyhedra, we perform a total of $21 \times 40 \times 40$ different MDD computation and the average CPU time taken for each MDD computation is taken. For each reported MDD α along a specified direction s, the reported MDD is verified by moving P_j along s by $-\alpha$ (which should make P_i and P_j in external contact), and use the GJK algorithm to compute the shortest distance of P_i and P_j. We note that the average shortest distance is 1.9×10^{-6} with a standard deviation of 10^{-5}; the maximum of the absolute shortest distance is found to be 10^{-4}.

The performance of MDD-DUAL is shown in Fig. 14. It takes more time for determining the penetration depth for two intersecting polyhedra, as the algorithm needs to repeat itself to find $d_{\min}$, and choose the minimum penetration depth among the values $\alpha, \bar{\alpha}$ as described in Section III-C. Nevertheless, for our test cases, MDD-DUAL takes less than 350 microseconds to compute the MDD of two convex polyhedra.

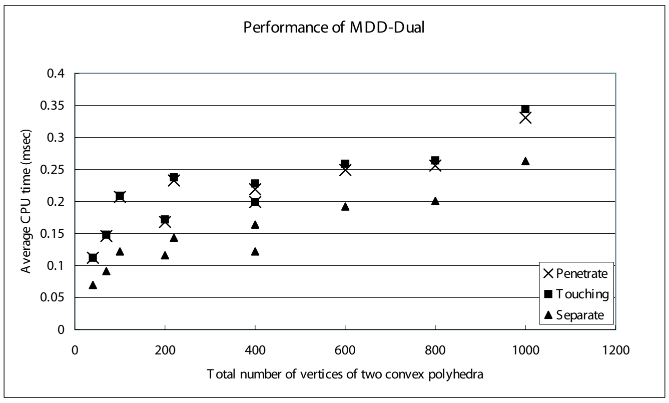


Fig. 14. The average CPU time for a MDD computation for 10 pairs of convex polyhedra.

Not all faces on the Minkowski sum $M = P \oplus -Q$ are being constructed and visited. A typical search path on M is shown in Fig. 15. From the above experiments, it is found that on average 13.7% of the faces on M is visited. In particular, only 2.5% of the EE-type faces is visited on average, which means that most of the EE-type faces (whose worst case complexity is $\mathcal{O}(n^2)$) can be skipped in the computations of MDD-DUAL .

VIII. CONCLUSION

We have presented a novel method, called MDD-DUAL , for computing the minimum directional distance (MDD) between two convex polyhedra along a given direction. The MDD of two convex polyhedra can be computed by finding the shortest directional distance from the origin to the Minkowski difference M of the polyhedra. This is done by finding a face which contains the intersection of a ray from the origin with M. We consider the problem in the dual space where a face on M corresponds to a vertex on the dual polyhedron M^*, and formulate the MDD computation in forms of searching a vertex which attains the maximum signed distance from a plane. The search problem in the dual space is easily shown to be convex, and a search scheme is devised accordingly that can locate the optimal face on M efficiently. The search scheme is divided into 3 stages, each working only on a subset of the faces on M. This division allows the elimination of most EE-type faces whose worst case complexity is $\mathcal{O}(n^2)$. Our experimental results show that MDD-DUAL exhibits both efficient and robust performance.

REFERENCES

[1] P. Jiménez, F. Thomas, and C. Torras, “3D collision detection: a survey.” *Computers & Graphics*, vol. 25, no. 2, pp. 269–285, 2001.
[2] M. C. Lin and D. Manocha, “Collision and proximity queries,” in *Handbook. of Discrete and Computational Geometry*, 2003.
[3] S. Cameron and R. Culley, “Determining the minimum translational distance between two convex polyhedra.” in *Proc. IEEE Int. Conf. on Robotics and Automation*, 1986, pp. 591–596.

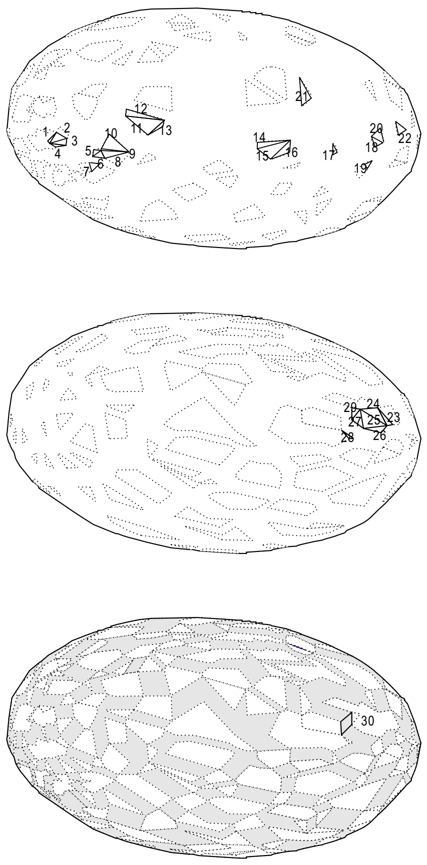


Fig. 15. Search path on M. Top: In SEARCH-FV (dotted regions contain faces in $\mathcal{F}_{\mathrm{fv}}$); Middle: In SEARCH-VF (dotted regions contain faces in $\mathcal{F}_{\mathrm{vf}}$); and Bottom: In SEARCH-EE (gray regions contain faces in $\mathcal{F}_{\mathrm{ee}}$).

[4] M. C. Lin and J. Canny, "A fast algorithm for incremental distance calculation." in *Proc. IEEE Int. Conf. on Robotics and Automation*, Sacremento, April, 1991, pp. 1008–1014.

[5] B. Mirtich, "V-Clip: Fast and robust polyhedral collision detection." *ACM Trans. Graph.*, vol. 17, no. 3, pp. 177–208, 1998.

[6] D. W. J. Elmer G. Gilbert and S. S. Keerthi, "A fast procedure for computing the distance between objects in three-dimensional space," *IEEE J. Robot. Automat.*, no. 4, pp. 193–203, 1988.

[7] S. Cameron, "A comparison of two fast algorithms for computing the distance between convex polyhedra," *IEEE Trans. Robot. Automat.*, vol. 13, no. 6, pp. 915–920, 1997.

[8] ——, "Enhancing GJK: Computing minimum and penetration distances between convex polyhedra." in *Proc. IEEE Int. Conf. on Robotics and Automation*, 1997, pp. 3112–3117.

[9] G. V. den Bergen, "A fast and robust GJK implementation for collision detection of convex objects," *J. Graph. Tools*, vol. 4, no. 2, pp. 7–25, 1999.

[10] P. K. Agarwal, L. J. Guibas, S. Har-Peled, A. Rabinovitch, and M. Sharir, "Penetration depth of two convex polytopes in 3d." *Nordic Journal of Computing*, vol. 7, no. 3, pp. 227–240, 2000.

[11] Y. J. Kim, M. C. Lin, and D. Manocha, "Incremental penetration depth estimation between convex polytopes using dual-space expansion." *IEEE Trans. Visual. Comput. Graphics*, vol. 10, no. 2, pp. 152–163, 2004.

[12] D. P. Dobkin, J. Hershberger, D. G. Kirkpatrick, and S. Suri, "Computing the intersection-depth of polyhedra." *Algorithmica*, vol. 9, no. 6, pp. 518–533, 1993.

[13] D. P. Dobkin and D. G. Kirkpatrick, "A linear algorithm for determining the separation of convex polyhedra." *J. Algorithms*, vol. 6, no. 3, pp. 381–392, 1985.

[14] ——, "Determining the separation of preprocessed polyhedra—a unified approach." in *Proc. 17th Int. Colloq. Automata Lang. Program*, ser. Lecture Notes in Computer Science, 1990, pp. 400–413.

[15] B. Grünbaum, *Convex Polytopes*. Wiley, 1967.

[16] H. G. Eggleston, *Convexity*. New York and London: Cambridge University Press, 1958.

[17] J. G. Semple and G. Kneebone, *Algebraic projective geometry*. Oxford University Press, 1952.

[18] H. Levy, *Projective and Related Geometry*. Macmillan, 1964.