HKU CS Tech Report TR-2005-01

Collision Detection of Convex Polyhedra Based on Duality Transformation

Yi-King Choi [§], Xueqing Li [†], Wenping Wang [§], Stephen Cameron [‡]

[§] Department of Computer Science, The University of Hong Kong, Hong Kong

[†] College of Computer Science and Technology of Shandong University, P.R. China

[‡] Computer Laboratory, Oxford University, United Kingdom

Abstract

Collision detection is an essential problem in many applications in computer graphics, CAD/CAM, and robotics. In this paper, a new method, called *CD-Dual*, for detecting collision between two convex polyhedra is proposed. The idea is based on a local search among the faces on the Minkowski difference (M) of the polyhedra. The local search is guided by a simple signed distance function defined on the dual polyhedron of M. Due to the convexity of the dual polyhedron, it is guaranteed that the result of the local search will lead to a vertex on the dual polyhedron that attains the global maximum signed distance, and this distance tells whether the two polyhedra overlap.

1 Introduction

Collision detection is one of the major subjects in robotics and computer graphics, or other areas that simulate physical environments [10]. Collision detection is to determine whether two moving objects collide or not at any moment. The ability to efficiently detect collision is critical in these applications since system responses are often based on the collision status of the objects. A typical approach is to determine whether two objects intersect at sampled time steps. Although collision occurring between two consecutive steps may be missed, this kind of errors can be reduced by decreasing the sampling time interval. Since objects' orientations and positions differ only slightly between successive time steps for objects with continuous motion, geometrical and temporal coherences can be exploited by making use of witness information, such as a separating plane [1], computed at a previous step.

Various approaches to detecting collision between convex polyhedra have been studied in the literature. One of these is the use of techniques from computational geometry. Voronoi diagrams are used to keep track of the closest features between pairs of objects in [5, 11]. I-COLLIDE [5] employs the "sweep-and-prune" technique to reduce the number of pairs of objects whose collision status is to be considered. While most algorithms are designed for convex objects, some work (e.g. [12]) has also been done for non-convex objects. Computing the intersection or the minimum distance between objects also serve as an alternative approach to collision detection. An $\mathcal{O}(\log^2 n)$ algorithm for polytope-polytope collision detection is given in [7], with the use of the hierarchical representation of a polytope. A commonly used method, GJK [9], that computes the distance between convex polyhedra makes use of the Minkowski difference and convex optimization techniques to compute the closest points. Modified approaches [3, 4] and improved implementations [2] based on GJK have been developed. It is also shown in [4] that an enhanced version of GJK has $\mathcal{O}(1)$ time cost under the assumption of strong geometric coherence.

Most existing methods are efficient when the distance between two objects is large or when two objects penetrate each other deeply. However, they become less efficient when the objects are very close to each other or just penetrate slightly. In this paper, we present a new algorithm, *CD-Dual*, to detect collision between two convex polyhedra based on duality transformation. The algorithm is shown to be fast even in the case when the separating distance or penetration distance is small; thus it exhibits a more balanced running time for all collision configurations. Detailed theoretical analysis and experimental results are also discussed.

2 Preliminaries

In this section, we shall give the definitions that are used in this paper. Also, the concept of duality will be described.

A convex polyhedron P in $\mathbb{E}^3$ can be defined as the intersection of half-spaces defined by some planes 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 *By considering 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 *Let $\mathcal{V}(P)$ denote the set of the vertices of a polyhedron P in $\mathbb{E}^3$. A* supporting vertex $\mathbf{p}$ *of P in the direction $\mathbf{s} \neq \mathbf{0}$ is a vertex in $\mathcal{V}(P)$ such that $\mathbf{s} \cdot \mathbf{p} = \max\{\mathbf{s} \cdot \mathbf{p}' | \mathbf{p}' \in \mathcal{V}(P)\}$, where $\mathbf{x} \cdot \mathbf{y}$ is the dot-product of the vectors $\mathbf{x}$ and $\mathbf{y}$.*

Note that for a supporting vertex $\mathbf{p}$ of P in the direction $\mathbf{s}$, we have $\mathbf{s} \cdot \mathbf{p} = \max\{\mathbf{s} \cdot \mathbf{p}' | \mathbf{p}' \in P\}$. We may also define the *supporting edge* or *supporting face* of P in $\mathbf{s}$ as the edge and face that contains two or more supporting vertices in $\mathbf{s}$.

2.1 Minkowski Sum of Two Polyhedra

Given two polyhedra P and Q, let $\bar{Q} = \{\bar{q} | -\bar{q} \in Q\}$. We consider the Minkowski sum M of P and $\bar{Q}$ defined by $M \equiv P \oplus \bar{Q} = \{\mathbf{p} + (-\mathbf{q}) | \mathbf{p} \in P, \mathbf{q} \in Q\}$. Since P and Q are convex polyhedra, M is a convex polyhedron. It can be shown [9] that P and Q overlap if and only if M contains the origin $\mathbf{0}$, i.e., $\mathbf{0} \in M$.

The Minkowski sum of two polyhedra P and $\bar{Q}$ is in fact the Minkowski difference of P and Q. Intuitively, we may think of P as continually expanding until it reaches $M \equiv P \oplus \bar{Q}$ while Q continually shrinking until it becomes the origin $\mathbf{0}$. If M contains the origin $\mathbf{0}$, we have $\mathbf{p} - \mathbf{q} = \mathbf{0}$ and therefore $\mathbf{p} = \mathbf{q}$ for some $\mathbf{p} \in P$ and $\mathbf{q} \in Q$. It means that P and Q share a common point, i.e., they overlap. Otherwise, if $\mathbf{0} \notin M$, we have $\mathbf{p} - \mathbf{q} \neq \mathbf{0}$ for all $\mathbf{p} \in P$ and $\mathbf{q} \in Q$. Therefore, P and Q are separate.

The faces in $F(M)$ may be classified into the following three subsets (Figure 1):

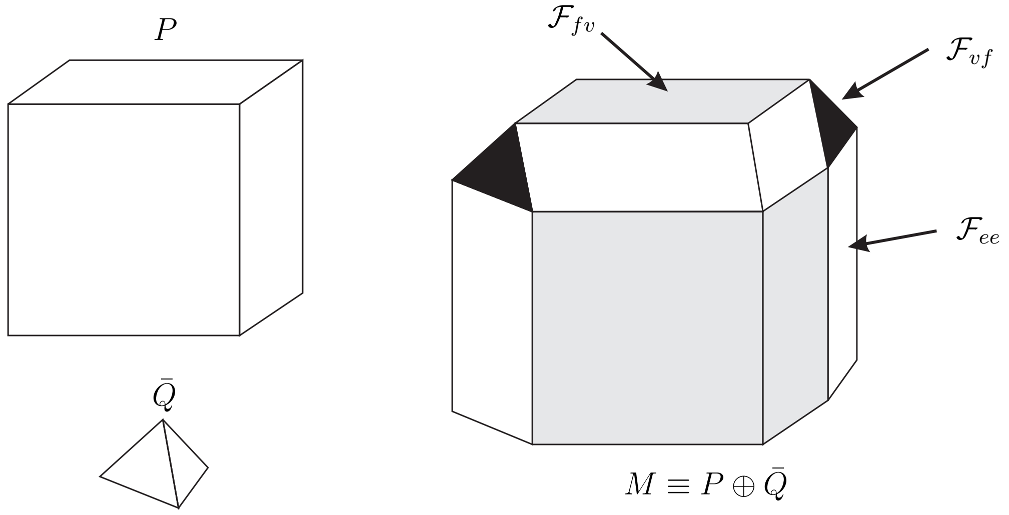


Figure 1: The Minkowski sum M of P and $\bar{Q}$. Faces on M can be classified as of type $\mathcal{F}_{fv}$, $\mathcal{F}_{vf}$, or $\mathcal{F}_{ee}$.

$\mathcal{F}_{fv}$: Each face $F(f_p, \mathbf{v}_{\bar{q}})$ is a point set $\{\mathbf{x}+\mathbf{v}_{\bar{q}} | \mathbf{x} \in f_p\}$, where $f_p \in \mathcal{F}(P)$ and $\mathbf{v}_{\bar{q}} \in \mathcal{V}(\bar{Q})$ is the supporting vertex of $\bar{Q}$ in the direction $\hat{\mathbf{n}}_{f_p}$, the normal vector of f_p.

$\mathcal{F}_{vf}$: Each face $F(\mathbf{v}_p, f_{\bar{q}})$ is a point set $\{\mathbf{v}_p+\mathbf{x} | \mathbf{x} \in f_{\bar{q}}\}$, where $f_{\bar{q}} \in \mathcal{F}(\bar{Q})$ and $\mathbf{v}_p \in \mathcal{V}(P)$ is the supporting vertex of P in the direction $\hat{\mathbf{n}}_{f_{\bar{q}}}$, the normal vector of $f_{\bar{q}}$.

$\mathcal{F}_{ee}$: Each face $F(e_p, e_{\bar{q}})$ is a parallelogram with vertices $(\mathbf{v}_0, \mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3)$ where $\mathbf{v}_0 = \mathbf{v}_{p0} + \mathbf{v}_{\bar{q}0}, \mathbf{v}_1 = \mathbf{v}_{p1} + \mathbf{v}_{\bar{q}0}, \mathbf{v}_2 = \mathbf{v}_{p1} + \mathbf{v}_{\bar{q}1}, \mathbf{v}_3 = \mathbf{v}_{p0} + \mathbf{v}_{\bar{q}1}$ with $\mathbf{v}_{p0}, \mathbf{v}_{p1} \in \mathcal{V}(P)$ forming an edge $e_p = (\mathbf{v}_{p0}, \mathbf{v}_{p1}) \in \mathcal{E}(P)$ and $\mathbf{v}_{\bar{q}0}, \mathbf{v}_{\bar{q}1} \in \mathcal{V}(\bar{Q})$ forming an edge $e_{\bar{q}} = (\mathbf{v}_{\bar{q}0}, \mathbf{v}_{\bar{q}1}) \in \mathcal{E}(\bar{Q})$. Moreover, the Gaussian images of e_p and $e_{\bar{q}}$ intersect on the unit sphere S^2.

Remark 1 *The Gaussian image $G(P)$ of a polyhedron P is a graph embedded on the unit sphere S^2 that is obtained as follows (Figure 2): A face $f \in \mathcal{F}(P)$ is transformed to a vertex $S(f) = \hat{\mathbf{n}} \in S^2$ where $\hat{\mathbf{n}}$ is the unit normal vector of f. An edge $e \in \mathcal{E}(P)$ adjacent to two faces $f_0, f_1 \in \mathcal{F}(P)$ is transformed to a great arc $A(e)$ connecting two vertices $S(f_0)$*

and $S(f_1)$ on S^2. In fact, $G(P)$ is a connected network on S^2 and each region of $G(P)$ corresponds to a vertex $\mathbf{v}_i \in \mathcal{V}(P)$ denoted by $R_P(\mathbf{v}_i)$.

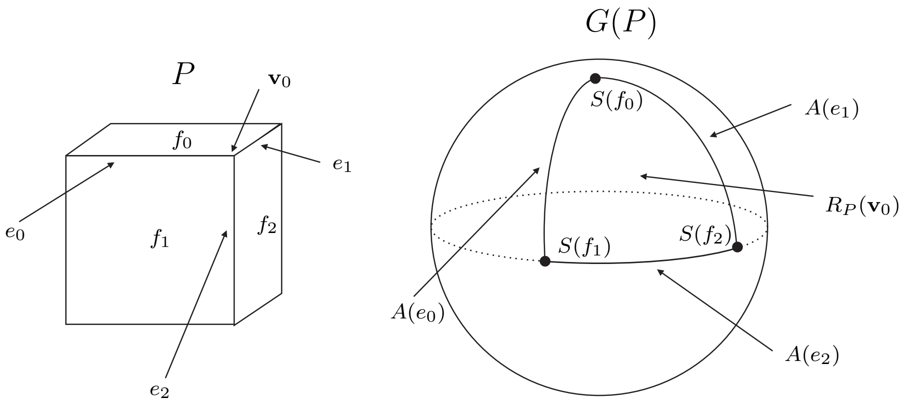


Figure 2: A polyhedron P and its Gaussian image $G(P)$ on S^2.

Remark 2 *The Gaussian image of a feature (vertex, edge or face) ϕ of a polyhedron P is the set of normal directions of planes that 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.*

Remark 3 *The Gaussian image of $M = P \oplus \bar{Q}$ (denoted by $G(M)$) can be obtained by superimposing the Gaussian images $G(P)$ and $G(\bar{Q})$. Each vertex in $G(M)$ corresponds to a face of M. Figure 3 shows how three kinds of vertices in $G(M)$ are related to the three types of faces in $\mathcal{F}(M)$ described above.*

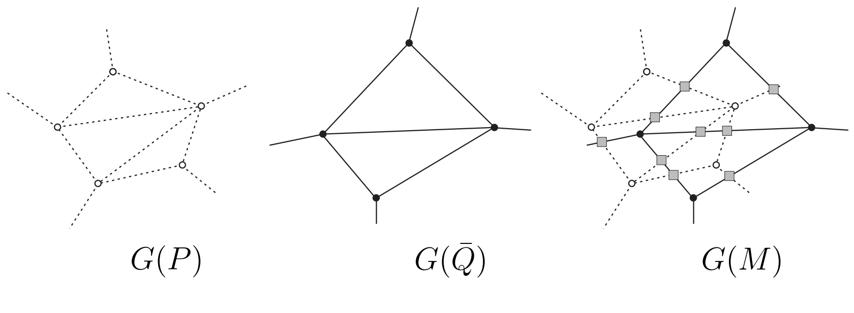


Figure 3: The planar representation of the Gaussian image $G(M)$ by superimposing $G(P)$ and $G(\bar{Q})$. There are three types of vertices in $G(M)$: (i) (white point) a vertex of $G(P)$ falling within a region of $G(\bar{Q})$, i.e., a face in $\mathcal{F}_{fv}$; (ii) (black point) a vertex of $G(\bar{Q})$ falling within a region of $G(P)$, i.e., a face in $\mathcal{F}_{vf}$; and (iii) (shaded square) a vertex formed by two intersecting arcs from $G(P)$ and $G(\bar{Q})$, i.e., a face in $\mathcal{F}_{ee}$.

2.2 Duality

Let a plane L in $\mathbb{E}^3$ be given by $ax+by+cz=d, d\neq 0$. The *duality transformation* denoted by D maps L to a point $\mathbf{l}=(a/d, b/d, c/d)^T$ in the dual space $\hat{\mathbb{E}}^3$. A point $\mathbf{u}=(r,s,t)^T$ in $\mathbb{E}^3$ is transformed by D to a plane $U: rx+sy+tz=1$ in $\hat{\mathbb{E}}^3$. A plane passing through the origin in $\mathbb{E}^3$ is mapped to a point at infinity in $\hat{\mathbb{E}}^3$. The origin in $\mathbb{E}^3$ is transformed to the plane at infinity in $\hat{\mathbb{E}}^3$. In fact $\mathbb{E}^3$ is also the dual space of $\hat{\mathbb{E}}^3$. See [8, 14] for more about duality transformation. There is thus a one-to-one correspondence between a plane (point) in $\mathbb{E}^3$ and a point (plane) in $\hat{\mathbb{E}}^3$. Table 1 summarizes some important properties of a polyhedron P in $\mathbb{E}^3$ and its dual $\hat{P}$ in $\hat{\mathbb{E}}^3$:

Table 1: Properties of a polyhedron P under a duality transformation.

In $\mathbb{E}^3$	In $\hat{\mathbb{E}}^3$
P is convex	$\hat{P}$ is convex
a face in $\mathcal{F}(P)$	a vertex in $\mathcal{V}(\hat{P})$
a vertex in $\mathcal{V}(P)$	a face in $\mathcal{F}(\hat{P})$
P contains the origin	$\hat{P}$ contains the origin
a point $\mathbf{x}$ inside P	the plane $D(\mathbf{x})$ does not intersect $\hat{P}$
a point $\mathbf{x}$ outside P	the plane $D(\mathbf{x})$ intersects $\hat{P}$

3 The Algorithm

In this section, we shall present our algorithm to detect collision between two convex polyhedra. We first give the basic idea of the algorithm.

3.1 Basic Idea

Let P and Q be two convex polyhedra in $\mathbb{E}^3$ and $M \equiv P \oplus \bar{Q}$ be their Minkowski difference. Let $\mathbf{o}$ be the origin of $\mathbb{E}^3$ and $\mathbf{c}$ any fixed point inside M, with $\mathbf{c}\neq\mathbf{o}$. Consider a translation $T:\mathbb{E}^3 \mapsto \mathbb{E}^3_T$ such that $\mathbf{c}'=T(\mathbf{c})$ is the origin of $\mathbb{E}^3_T$ (Figure 4). Let $\mathbf{o}'=T(\mathbf{o})=-\mathbf{c}$ and $M'=T(M)$. Since $\mathbf{c}\in M$, M' must contain the origin. Next, we apply a dual transformation $D:\mathbb{E}^3_T \mapsto \hat{\mathbb{E}}^3_T$ that maps $\mathbf{o}'$ to a plane $\hat{\mathbf{o}}$ and the polyhedron M' to a polyhedron $\hat{M}$. As $\mathbf{o}'$ is not the origin, $\hat{\mathbf{o}}$ is not the plane at infinity. Also, the plane $\hat{\mathbf{o}}$ intersects $\hat{M}$ if and only if $\mathbf{o}'\notin M'$. Hence, we have

$$\begin{aligned} P \text{ and } Q \text{ overlap} &\Leftrightarrow \mathbf{o}\in M \\ &\Leftrightarrow \mathbf{o}'\in M' \\ &\Leftrightarrow \hat{\mathbf{o}} \text{ does not intersect } \hat{M}. \end{aligned}$$

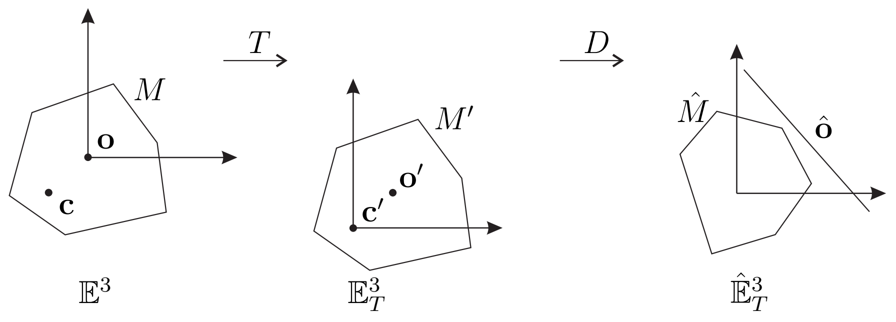


Figure 4: A 2D illustration of how $\hat{M}$ and $\hat{\mathbf{o}}$ are obtained. Note that if $\mathbf{o} \notin M$, $\hat{\mathbf{o}}$ intersects $\hat{M}$.

Therefore, we just need to test if the plane $\hat{\mathbf{o}}$ intersects the polyhedron $\hat{M}$ in order to detect if P and Q overlap.

3.2 Signed Distance Function

Recall the *signed distance* of a point $\mathbf{p} = [x_0, y_0, z_0]^T$ to a plane $L : \mathbf{A}^T\mathbf{x} = k$ which is given by $\mathrm{d}_L(\mathbf{p}) = \mathbf{A}^T\mathbf{p} - k$, where $\mathbf{A} = [l, m, n]^T$ is the unit normal of L, $\mathbf{x} = [x, y, z]^T$, and $k > 0$ is the perpendicular distance of the origin $\mathbf{0}$ to the plane. Note that, if $\mathbf{p}$ lies on L, $\mathrm{d}_L(\mathbf{p}) = 0$; if $\mathbf{p}$ lies on the same side of L as $\mathbf{0}$, $\mathrm{d}_L(\mathbf{p}) < 0$; and if $\mathbf{p}$ lies on the opposite side of L to $\mathbf{0}$, $\mathrm{d}_L(\mathbf{p}) > 0$.

Let d' be the maximum signed distance of all points in $\hat{M}$ to the plane $\hat{\mathbf{o}}$. Since $\hat{M}$ is convex, the point that attains the maximum signed distance d' on $\hat{M}$ to the plane $\hat{\mathbf{o}}$ must lie on the boundary of $\hat{M}$, i.e., either on a vertex, a face or an edge. In any case, there is a vertex $\mathbf{v} \in \mathcal{V}(\hat{M})$ such that its signed distance to the plane $\hat{\mathbf{o}}$, $\mathrm{d}_{\hat{\mathbf{o}}}(\mathbf{v})$, equals d'. Hence, we may consider only the signed distance of all vertices of $\hat{M}$ to the plane $\hat{\mathbf{o}}$.

If $d' < 0$, all vertices on $\hat{M}$ are on the same side of $\hat{\mathbf{o}}$ as the origin and therefore $\hat{M}$ and $\hat{\mathbf{o}}$ do not intersect. Otherwise if $d' > 0$, at least one vertex is at the opposite side of $\hat{\mathbf{o}}$ to the origin, which implies that $\hat{\mathbf{o}}$ intersects $\hat{M}$ and we conclude that P and Q do not overlap. When $d' = 0$, $\hat{\mathbf{o}}$ touches $\hat{M}$ at some boundary point. So $\mathbf{o}$ lies on the boundary of M in $\mathbb{E}^3$. By the construction of M as described in section 2.1, $\mathbf{o} = \mathbf{p} - \mathbf{q}$ 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. In this case, P and Q touch each other.

Let f be a face in $\mathcal{F}(M)$ and $\hat{\mathbf{v}} \in \mathcal{V}(\hat{M})$ be its corresponding dual vertex under $D \circ T$. We may then define the *signed distance* of f denoted by $\mathrm{d}(f)$, to be the signed distance of $\hat{\mathbf{v}}$ to the plane $\hat{\mathbf{o}}$, i.e., $\mathrm{d}_{\hat{\mathbf{o}}}(\hat{\mathbf{v}})$. Let H_f be the plane containing f with plane equation $\mathbf{N}^T\mathbf{x} = k$ with $k > 0$. Then f is translated by T to a face in $\mathcal{F}(M')$ with plane equation $\mathbf{N}^T\mathbf{x} = k - \mathbf{N}^T\mathbf{c}$. This corresponds to $\hat{\mathbf{v}} = \mathbf{N}/(k - \mathbf{N}^T\mathbf{c})$ in $\mathcal{V}(\hat{M})$ in the dual space. The origin $\mathbf{o}$ in $\mathbb{E}^3$ is mapped to $-\mathbf{c}$ in $\mathbb{E}^3_T$ and therefore the plane equation of $\hat{\mathbf{o}}$ in $\hat{\mathbb{E}}^3_T$ is

$-\mathbf{c}^T\mathbf{x} = 1$. The signed distance $\mathrm{d}(f)$ can then be expressed explicitly as follows:

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

Let us now take a closer look at the geometrical meaning of $\mathrm{d}(f)$. The quantity $\mathrm{d}(f) = \mathrm{d}_{\hat{\mathbf{o}}}(\hat{\mathbf{v}})$ uniquely determines a plane $\hat{L}$ in $\hat{\mathbb{E}}^3_T$ such that $\mathrm{d}_{\hat{\mathbf{o}}}(\mathbf{x}) = \mathrm{d}_{\hat{\mathbf{o}}}(\hat{\mathbf{v}})$ for all points $\mathbf{x} \in \hat{L}$ (Figure 5). It is easy to see that the plane $\hat{L}$ passes through $\hat{\mathbf{v}}$ and is parallel to the plane $\hat{\mathbf{o}}$. Let $\mathbf{l}$ be a point in $\mathbb{E}^3$ whose image under $D \circ T$ is $\hat{L} \in \hat{\mathbb{E}}^3_T$. Since $\hat{L}$ has the same

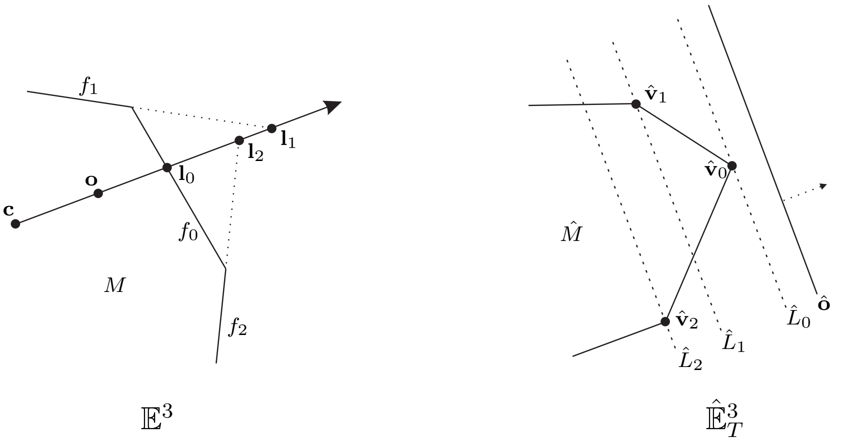


Figure 5: The vertex $\hat{\mathbf{v}}_0$ in $\hat{\mathbb{E}}^3_T$ attaining maximum signed distance to $\hat{\mathbf{o}}$ corresponds to a face f_0 in $\mathbb{E}^3$ intersecting the directed line $\mathbf{co}$. The dual image of $\mathbf{o}$, f_i and $\mathbf{l}_i$ under $D \circ T$ are $\hat{\mathbf{o}}$, $\hat{\mathbf{v}}_i$ and $\hat{L}_i$, $i = 0, 1, 2$, respectively.

normal direction as $\hat{\mathbf{o}}$, $\mathbf{l}$ must lie on the directed line $\mathbf{co}$. Moreover, $\hat{L}$ passes through $\hat{\mathbf{v}}$ and therefore $\mathbf{l}$ lies on the plane H_f containing the face f. This implies that $\mathbf{l}$ is the intersection of the plane H_f and the line $\mathbf{co}$. Note that a larger $\mathrm{d}(f)$ corresponds to a point $\mathbf{l}$ closer to $\mathbf{c}$. If $\mathrm{d}(f)$ is the largest among the signed distances of all faces in $\mathcal{F}(M)$, the intersection point of the plane H_f and the line $\mathbf{co}$ must be the closest to the point $\mathbf{c}$ among all intersections between the directed line $\mathbf{co}$ and the planes containing the faces of M. This can happen only when the line $\mathbf{co}$ intersects the face f, since $\mathbf{c} \in M$ and M is convex. Now, the signed distance of $\mathbf{c}$ and $\mathbf{o}$ to the plane H_f are given by $\mathrm{d}_{H_f}(\mathbf{c}) = -(k - \mathbf{N}^T\mathbf{c})/\|\mathbf{N}\|$ and $\mathrm{d}_{H_f}(\mathbf{o}) = -k/\|\mathbf{N}\|$, respectively. If $\mathrm{d}(f) < 0$, since $k > 0$, by Eq. (1), $k - \mathbf{N}^T\mathbf{c} > 0$ and therefore $\mathrm{d}_{H_f}(\mathbf{c})$ and $\mathrm{d}_{H_f}(\mathbf{o})$ are of the same sign; thus $\mathbf{c}$ and $\mathbf{o}$ are on the same side of the face f and $\mathbf{o} \in M$. On the other hand, if $\mathrm{d}(f) > 0$, $\mathbf{c}$ and $\mathbf{o}$ are on opposite sides of f and therefore $\mathbf{o} \notin M$.

It is now clear that we are in fact 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 $\mathbf{co}$, by computing the maximum signed distance of all faces in M. From Eq. (1), we see that $\mathrm{d}(f)$ can be computed without even applying the duality transformation on M. However, it becomes apparent that $\mathrm{d}(f)$ is a convex function by considering $\mathrm{d}_{\hat{\mathbf{o}}}(\hat{\mathbf{v}})$ in the dual space. Since $\hat{M}$ is convex, a vertex $\hat{\mathbf{v}}$ in $\mathcal{V}(\hat{M})$ attaining a local maximum signed distance must also attain the global maximum signed distance among all vertices in $\mathcal{V}(\hat{M})$.

We may then define our objective function as the signed distance of a face f in $\mathcal{F}(M)$ and let $d_{\max} = \max\{\mathrm{d}(f) | f \in \mathcal{F}(M)\}$ denote the maximum signed distance of all faces in $\mathcal{F}(M)$. Now the problem of detecting collision between two polyhedra P and Q can be transformed to finding the optimal value $d_{\max}$. Starting from any face on $\mathcal{F}(M)$, we can go to the next face that has the largest signed distance among all immediate neighbours of the current face. By this local search, we shall visit faces with increasing signed distance and eventually stop at a face with a locally maximum signed distance. Due to the convexity of the objective function $\mathrm{d}(f)$, this local search scheme will lead to the *optimal face*, f_{opt}, that attains the maximum signed distance $d_{\max}$ among all faces in $\mathcal{F}(M)$.

As a further remark, let α be the distance between the point $\mathbf{o}$ and the intersection of the directed line $\mathbf{co}$ and f_{opt}. If P and Q are separate, α is their *separating distance* along the direction $\mathbf{co}$, which is the distance that Q can be moved along $\mathbf{co}$ until it touches the fixed P. If P and Q are overlap, α is then their *penetration distance* along $\mathbf{co}$, which is the distance that Q can be moved along $\mathbf{co}$ until it touches the fixed P externally. Further details can be found in section 6.

The remaining problem now is to find efficiently the optimal face f_{opt}. Note that it is possible to have more than one f_{opt} that attain $d_{\max}$. This happens when the line $\mathbf{co}$ intersects M at an edge or a vertex.

3.3 Searching for the optimal vertex

A naive way to search for f_{opt} is to first construct the Minkowski difference $M \equiv P \oplus \bar{Q}$ and perform the search on the vertices in M. However, the time complexity of constructing M is $\mathcal{O}(mn)$ in the worst case, where m and n are the number of vertices of P and Q, respectively. Moreover, it would take a long time to traverse the faces on M by advancing one edge to the neighbour at each step. To avoid the high cost of constructing M, our searching scheme is based on three successive search phases for f_{opt} within three subsets of the faces of M, without completely constructing M.

Note that each face $f \in \mathcal{F}(M)$ must belong to either one of the sets $\mathcal{F}_{fv}$, $\mathcal{F}_{vf}$ or $\mathcal{F}_{ee}$. Our algorithm is to search for f_{opt} in each of the three sets separately and successively in a local manner. Three procedures for this searching will be described in the following subsections. The correctness of these procedures are based on three theorems whose proofs can be found in the Appendix.

3.3.1 Procedure Search-FV

This procedure is to search for a face in $\mathcal{F}(M)$ having the maximum signed distance among all faces in $\mathcal{F}_{fv}$. We first choose a face $f_0 \in \mathcal{F}(P)$ with normal vector $\mathbf{n_p}$. We may use heuristic search in a preprocessing step so that the face $F(f_0, \mathbf{v}_0) \in M$ formed by f_0 and its supporting vertex $\mathbf{v_0}$ in $\bar{Q}$ is close to f_{opt} on M. The heuristic search for f_0 will be discussed in detailed in section 4.2. The following pseudocode describes the procedure:

```
PROCEDURE SEARCH-FV
    $(\mathrm{d}_m, f_m) = $ SIGNEDDISTANCE-FV$(f_0)$
    For each iteration $i$
        If $\mathrm{d}_m > 0$,
            Return $(\mathrm{d}_m, f_m)$.
        Obtain the $n$ faces $f_i^0, f_i^1, \ldots, f_i^{n-1}$ that are adjacent to $f_i$ in $P$
        For each face $f_i^j$,
            $(\mathrm{d}_i^j, f_{m,i}^j) \leftarrow$ SIGNEDDISTANCE-FV$(f_i^j)$.
        If $\mathrm{d}_m < \mathrm{d}_i^k$, where $\mathrm{d}_i^k = \max\{\mathrm{d}_i^j | j \in [0, n-1]\}$
            $\mathrm{d}_m \leftarrow \mathrm{d}_i^k, f_m \leftarrow f_{m,i}^k, f_{i+1} \leftarrow f_i^k$;
        Otherwise,
            Return $(\mathrm{d}_m, f_m)$.

FUNCTION SIGNEDDISTANCE-FV$(f_p \in \mathcal{F}(P))$
    $\mathbf{v}_{\bar{q}} \leftarrow$ the supporting vertex of $\bar{Q}$ in the direction $\mathbf{n}$,
      where $\mathbf{n}$ is the normal of $f_p$.
    Obtain a face $f_m \in \mathcal{F}_{fv}$ where $f_m = \{\mathbf{x} + \mathbf{v}_{\bar{q}} | \mathbf{x} \in f_p\}$.
    $\mathrm{d}_m \leftarrow \mathrm{d}(f_m)$
    Return $(\mathrm{d}_m, f_m)$
```

The determination of the supporting vertex of $\bar{Q}$ given a direction $\mathbf{n}$ in the subroutine SIGNEDDISTANCE-FV can be accelerated by using the hierarchical representation of a polyhedron as described in [6].

Theorem 1 *The face $f_m \in \mathcal{F}(M)$, computed by* SEARCH-FV*, attains the maximum signed distance among all faces in $\mathcal{F}_{fv}$, i.e.,* $\mathrm{d}(f_m) = \max\{\mathrm{d}(f) | f \in \mathcal{F}_{fv}\}$.

3.3.2 Procedure Search-VF

This procedure is similar to SEARCH-FV in that it searches for the a face in $\mathcal{F}(M)$ attaining the maximum signed distance among all faces in $\mathcal{F}_{vf}$. The only difference is that the roles of P and $\bar{Q}$ are interchanged and hence the pseudocode for the procedure is omitted for brevity. The first face $f_0 \in \mathcal{F}(\bar{Q})$ from which the search is started can be chosen by heuristic as for SEARCH-VF. An alternative is to choose a face $f_0 \in \mathcal{F}(\bar{Q})$ that is incident at $\mathbf{v}_{\bar{q}}$ in $\bar{Q}$ where $f_m = F(f_p, \mathbf{v}_{\bar{q}})$ is the face output by SEARCH-FV, since we have been approaching the optimal face f_{opt} at each step of the procedure.

Theorem 2 *The face $f_m \in \mathcal{F}(M)$, computed by* SEARCH-VF*, attains the maximum signed distance among all faces in $\mathcal{F}_{vf}$, i.e.,* $\mathrm{d}(f_m) = \max\{\mathrm{d}(f) | f \in \mathcal{F}_{vf}\}$.

3.3.3 Procedure Search-EE

By the procedures Search-FV and Search-VF, we determined the face f that attains the maximum signed distance among all faces in the set $\mathcal{F}_{fv} \cup \mathcal{F}_{vf}$. Starting from f, we shall search for the remaining faces in $\mathcal{F}_{ee}$.

Let $\langle e_p, e_{\bar{q}} \rangle$ be an edge pair with $e_p \in \mathcal{E}(P)$ and $e_{\bar{q}} \in \mathcal{E}(\bar{Q})$. As mentioned in section 2.1, if the arcs that are the Gaussian images of e_p and $e_{\bar{q}}$ intersect on S^2, a face $F(e_p, e_{\bar{q}}) \in \mathcal{F}_{ee}$ will be formed. We shall describe in details how to determine whether two arcs intersect in section 4.3.

The following procedure Search-EE makes use of a stack $\mathcal{S}$ to keep track of all candidate edge pairs $\langle e_p^i, e_{\bar{q}}^i \rangle$ which might give us the face in $\mathcal{F}_{ee}$ that attains the largest signed distance among all faces in $\mathcal{F}_{ee}$.

```
Procedure Search-EE
    $\mathrm{d}_m \leftarrow \max\{\mathrm{d}_m$ computed by Search-FV and Search-VF$\}$.
    $f_m \leftarrow$ the face in $\mathcal{F}_{fv} \cup \mathcal{F}_{vf}$ that attains $\mathrm{d}_m$.
    Push to $\mathcal{S}$ all possible edge pair $\langle e'_p, e'_{\bar{q}} \rangle$
      where
      $e'_p$ are edges of $f_p$ and $e'_{\bar{q}}$ are edges incident at $\mathbf{v}_{\bar{q}}$, if $f_m = F(f_p, \mathbf{v}_{\bar{q}})$, or
      $e'_p$ are edges incident at $\mathbf{v}_p$ and $e'_{\bar{q}}$ are edges of $f_{\bar{q}}$, if $f_m = F(\mathbf{v}_p, f_{\bar{q}})$.
    Repeat
        If $\mathrm{d}_m > 0$,
            Return $(\mathrm{d}_m, f_m)$.
        Pop $\langle e_p^i, e_{\bar{q}}^i \rangle$ from $\mathcal{S}$.
        If $\langle e_p^i, e_{\bar{q}}^i \rangle$ forms a face $f_i \in \mathcal{F}_{ee}$,
            $\mathrm{d}_i \leftarrow \mathrm{d}(f_i)$
            If $\mathrm{d}_i > \mathrm{d}_m$,
                $\mathrm{d}_m \leftarrow \mathrm{d}_i, f_m \leftarrow f_i$
            Push to $\mathcal{S}$ all possible edge pairs $\langle e_p^i, e_{\bar{q}}^j \rangle, \langle e_p^j, e_{\bar{q}}^i \rangle$,
              where $e_{\bar{q}}^j$ are edges incident at the two end vertices of $e_{\bar{q}}^i$,
              and $e_p^j$ are edges incident at the two end vertices of $e_p^i$.
    Until $\mathcal{S}$ is empty.
    Return $(\mathrm{d}_m, f_m)$.
```

Theorem 6 *The face $f_m \in \mathcal{F}(M)$, computed by* Search-EE, *attains the maximum signed distance among all faces in $\mathcal{F}(M)$, i.e., $f_m = f_{\mathrm{opt}}$ and $\mathrm{d}(f_m) = d_{\max}$.*

With the above theorems and the ideas presented, the following pseudocode describes our algorithm to detect whether two convex polyhedra P and Q overlap:

```
PROCEDURE CD-DUALITY($P$, $Q$)
    $(d_{\max}, f_{\max}) \leftarrow$ SEARCH-FV
    $(d_{\max}, f_{\max}) \leftarrow$ SEARCH-VF
    $(d_{\max}, f_{\max}) \leftarrow$ SEARCH-EE
    If $d_{\max} > 0$
        Report $P$ AND $Q$ ARE SEPARATE
    Otherwise
        Report $P$ AND $Q$ ARE OVERLAP
```

4 Implementation Issues

In this section, we shall highlight several important issues in implementing our algorithm so that the computations can be carried out efficiently.

4.1 To obtain a point c inside M

To compute the signed distance for a face in M, we need to determine an interior point $\mathbf{c} \in M$ which must not be the origin. The point $\mathbf{c}$ can be any arbitrary point as long as it is inside M. Therefore, we might store two distinct interior points, $\mathbf{p}_0, \mathbf{p}_1 \in P$ and $\mathbf{q}_0, \mathbf{q}_1 \in Q$, for each of P and 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.

4.2 To obtain the initial face in Search-FV

In section 3.3.1, we mentioned that a heuristic method can be used to get the initial face $f_0 \in \mathcal{F}(P)$ from where SEARCH-FV starts. The aim of this step is to optimize the searching for f_{opt} and therefore f_0 should be chosen such that $F(f_0, \mathbf{v}_0)$ is as close to f_{opt} as possible, where $\mathbf{v}_0$ is the supporting vertex of $\bar{Q}$ in the normal direction of f_0. Noting from the properties of duality that the normal of the plane $\hat{\mathbf{o}}$ is in fact the vector $\mathbf{s} = \mathbf{o} - \mathbf{c}$ (Figure 4), therefore f_{opt} must be front-facing with respect to $\mathbf{s}$ with $\mathbf{n}_{f_{\text{opt}}} \cdot \mathbf{s} > 0$, where $\mathbf{n}_{f_{\text{opt}}}$ is the normal of f_{opt}. We may then take the face $f_0 \in \mathcal{F}(P)$ with normal $\mathbf{n}_0$ such that $\mathbf{n}_0 \cdot \mathbf{s}$ is the greatest among all faces in $\mathcal{F}(P)$. $F(f_0, \mathbf{v}_0)$ may not be as close to f_{opt} especially when M is flat and elongated (Figure 6) and in fact this drawback is reflected in our experiments. However, this heuristic scheme can still efficiently eliminate most back-facing faces f with normal $\mathbf{n}_f$ in M with respect to $\mathbf{s}$ such that $\mathbf{n}_f \cdot \mathbf{s} < 0$.

4.3 To decide whether two arcs on S^2 intersect

In procedure SEARCH-EE, we often need to decide whether two arcs $A(e_p)$, $A(e_{\bar{q}})$ of the edge pair $\langle e_p, e_{\bar{q}} \rangle$ intersect on the Gaussian sphere S^2. Therefore, the computation has to be done efficiently. Now, let $\mathbf{a}, \mathbf{b}$ be the end points of $A(e_p)$, $\mathbf{c}, \mathbf{d}$ be the end points of $A(e_{\bar{q}})$ and $\mathbf{o}$ be the centre of S^2 (Figure 7). A_p and $A_{\bar{q}}$ intersect if and only if (1) $\mathbf{c}, \mathbf{d}$ are on

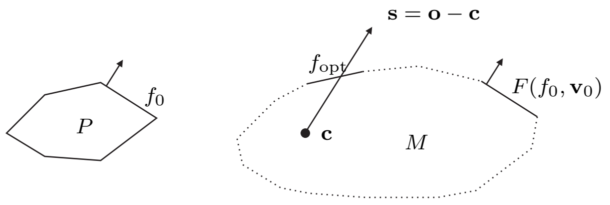


Figure 6: The face formed by the first face f_0 chosen by the heuristic scheme may not be close to f_{opt} in M.

different sides of plane **oba**; (2) $\mathbf{a}, \mathbf{b}$ are on different sides of plane **ocd**; and (3) $\mathbf{a}, \mathbf{b}, \mathbf{c}, \mathbf{d}$ are on the same hemisphere.

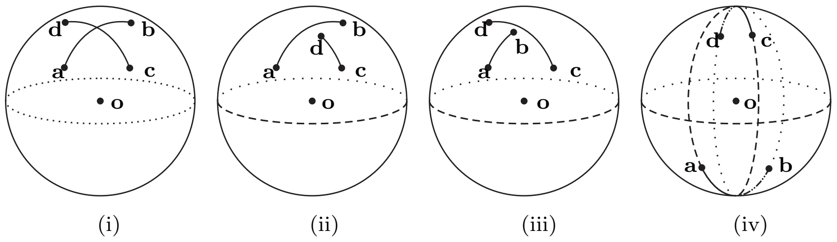


Figure 7: Determining whether two arcs intersect on S^2. Arcs intersect in (i). No intersection between arcs where (ii) only condition (1); (iii) only condition (2) and (iv) only condition (3) is violated.

Consider the *signed volume*, $|\mathbf{cba}| = \det[\ \mathbf{c}\ \ \mathbf{b}\ \ \mathbf{a}\ ]$, 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 they are at opposite sides of plane **oba**.

Therefore, the above three conditions can be formulated as (1) $|\mathbf{cba}| \times |\mathbf{dba}| < 0$; (2) $|\mathbf{adc}| \times |\mathbf{bdc}| < 0$; and (3) $|\mathbf{acb}| \times |\mathbf{dcb}| > 0$ since $\mathbf{a}, \mathbf{d}$ lie on the same side of plane **ocb** and all four points will be on the same hemisphere defined by **ocb**. Noting that only the quantities $|\mathbf{cba}|, |\mathbf{dba}|, |\mathbf{adc}|$ and $|\mathbf{bdc}|$ are required, since $|\mathbf{acb}| = |\mathbf{cba}|$ and $|\mathbf{dcb}| = |\mathbf{bdc}|$.

4.4 Frame Coherence

When the two polyhedra P and Q assume continuous motion from frame to frame, our algorithm may also exploit the temporal or frame coherence. At each time frame, we compute f_{opt} either in $\mathcal{F}_{fv}, \mathcal{F}_{vf}$ or $\mathcal{F}_{ee}$ that attains the maximum signed distance among all faces in $\mathcal{F}(M)$. In any case, we can determine quickly a face $f' = F(f_p, \mathbf{v}_{\bar{q}}) \in \mathcal{F}_{fv}$ that is as close to f_{opt} as possible.

For the next frame, since the position and orientation of P and Q will only have little changes, the new optimal face should also be close to f_{opt} and the use of f_p as the initial face for Search-FV will lead us to the new optimal face more rapidly.

5 Results

We have tested *CD-Dual* on a PC with a Pentium III 1.7GHz CPU. The algorithm is implemented in C++ and is compared against the enhanced version of the GJK method [4]. Since GJK computes also the minimum distance or the penetration distance of two polyhedra, its implementation has been modified so that it will complete computations once collision or non-collision is determined. Two polyhedra that are both approximations of some ellipsoids are obtained by randomly generating points on the surface of the ellipsoids. At each run, the two polyhedra are of the same ellipsoidal shape, size and number of vertices. The distance l between the two polyhedra varies and it is the distance that one object must be moved in one direction so that two objects become separate. A negative l corresponds to two penetrating objects while a positive l means that two objects are separate. For any fixed l, the orientation of one polyhedron is fixed while the other assumes 500 orientations; and 500 pairs of polyhedra are thus obtained. The two algorithms are then run on each pair of polyhedra for 300 times without using frame coherence and the average CPU time is recorded.

Figure 8 shows the experimental results with the varying number of vertices of the polyhedra. It is noticed that there are some peaks in the graphs for GJK near $l = 0$, i.e., when the two polyhedra are close to each other or their penetration distance is small. This is because in such situations, the GJK algorithm converges slowly. In contrast, *CD-Dual* has an almost constant running time for cases where the two polyhedra touch ($l = 0$) or intersect ($l < 0$), no matter what the penetrating distance is. This is because the face f_{opt} that attains the maximum signed distance has to be reached to confirm the colliding status. The graphs also show that the CPU time drops significantly for *CD-Dual* when the two polyhedra are separate ($l > 0$). The reason is that *CD-Dual* can determine separation once a face in M with positive signed distance is reached. The running time decreases gradually as the separating distance increases because there are more faces in M with positive signed distance and separation can therefore be determined more quickly. There is also mild increase in the running time for our algorithm when the number of vertices on each polyhedra increases. Although the graph shows favourable running time of *CD-Dual* over GJK, it should be reminded that the CPU time depends on the specific implementation and the optimization level of the algorithms.

Figure 9 shows the CPU time of both algorithms with the shape of the approximated ellipsoid varying. Notice that *CD-Dual* takes longer time to complete when the ellipsoid is more elongated in shape. As explained section 4.2, M is most likely to be elongated in this case, and therefore the first chosen face in $\mathcal{F}_{fv}$ is not close to the optimal face and it takes more steps to reach the optimal face.

The amount of time spent on each of the procedures Search-FV, Search-VF, and Search-EE is shown in the accumulated graph in Figure 10. Little time has been used in Search-VF since the procedure starts from the optimal face in $\mathcal{F}_{fv}$ which is often close

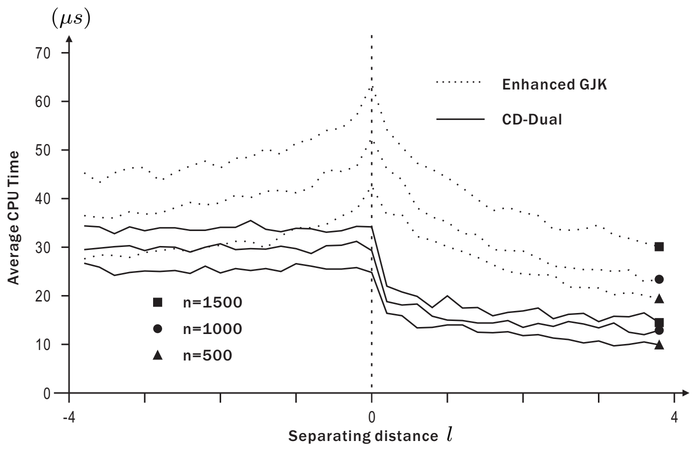


Figure 8: Varying number of vertices (n) on the polyhedra with a fixed size of the approximated ellipsoid (200:100:100, i.e., the sizes along the three major axes).

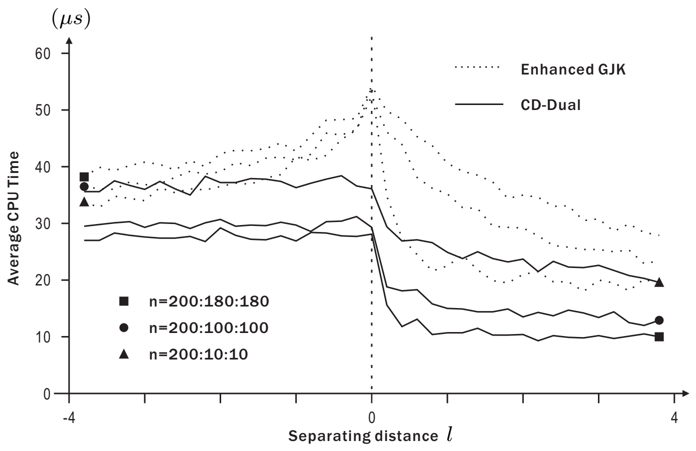


Figure 9: Varying ellipsoid shapes ($a : b : c$, i.e., the sizes along the three major axes) for the two polyhedra with fixed number of vertices ($n = 1000$).

to the target face in $\mathcal{F}_{vf}$, i.e., the face with maximum signed distance in $\mathcal{F}_{vf}$. Search-EE takes more time to complete as there are in general more $\mathcal{F}_{ee}$-type faces in M (although most such faces have been skipped by Search-FV and Search-VF) and the computation in each step involves the more complicated arc-arc intersection when compared to the other two procedures. However, when the separation distance increases, the time needed for Search-EE drops drastically as in most of the time a face in M with positive signed distance is reached quickly and the algorithm will report separation, sometimes even within the procedures Search-FV and Search-VF. Figure 11 shows how the searches proceed in each of the procedures in a typical scenario.

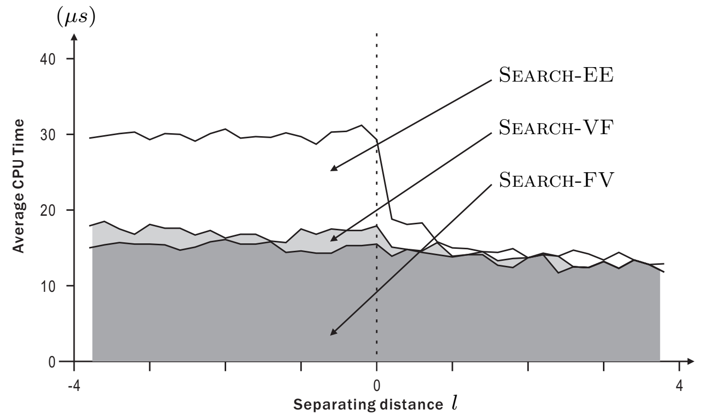


Figure 10: Accumulated graph showing time spent on each procedure with 1000 vertices on each polyhedron of size 200 : 100 : 100.

6 Estimating Minimum Separating Distance Along a Specified Direction

Let P and Q be two polyhedra. If P and Q are separate, the *minimum separating distance along a direction* $\mathbf{s}$ between P and Q is defined as the minimum distance that Q should move in the direction $\mathbf{s}$ (with P fixed) such that P and Q just touch each other (Figure 12(a)). To determine this distance using *CD-Dual*, we may choose any point $\mathbf{c}$ in $M \equiv P \oplus \bar{Q}$ such that $\mathbf{c} = k\mathbf{s}$, for some constant $k > 0$.

We first obtain two sets of points $\dot{\mathcal{P}}$ and $\dot{\mathcal{Q}}$ by applying an orthographic projection along $\mathbf{s}$ of all vertices of P and Q to a plane H normal to $\mathbf{s}$. This projection can be done in $\mathcal{O}(m+n)$ time (Figure 13(a)) where m and n are the number of vertices of P and Q, respectively. The next step is to construct the convex hull, $CH(\dot{P})$ and $CH(\dot{Q})$, of the points $\dot{\mathcal{P}}$ and $\dot{\mathcal{Q}}$, respectively, which can be done in $\mathcal{O}(m+n)$ time since the boundary vertices of $CH(\dot{P})$ and $CH(\dot{Q})$ are the silhouette vertices of P and Q as viewed along $\mathbf{s}$. P and Q intersect when Q is moved along $\mathbf{s}$ if and only if $CH(\dot{P})$ and $CH(\dot{Q})$ intersect and therefore the Minkowski difference $\dot{M}$ of $CH(\dot{P})$ and $CH(\dot{Q})$ contains the origin $\mathbf{0}$. Note that $\dot{M}$ can be built in $\mathcal{O}(m+n)$ time. Let $\dot{\mathbf{r}}_i = \dot{\mathbf{p}}_i - \dot{\mathbf{q}}_i$ be the vertices of $\dot{M}$ in anticlockwise order (Figure 13(b)). We may then quickly determine in $\mathcal{O}(m+n)$ 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 $\mathbf{0} \in \triangle\dot{\mathbf{r}}_0\dot{\mathbf{r}}_j\dot{\mathbf{r}}_{j+1}$, for some $j = 1, \ldots, l-2$. If $\mathbf{0} \notin \dot{M}$, the two polyhedra P and Q do not collide no matter how far Q is moved along $\mathbf{s}$ or $-\mathbf{s}$ and the point $\mathbf{c}$ cannot be found. Otherwise, let (u, v, w), where $u, v, w > 0$ and $u + v + w = 1$,

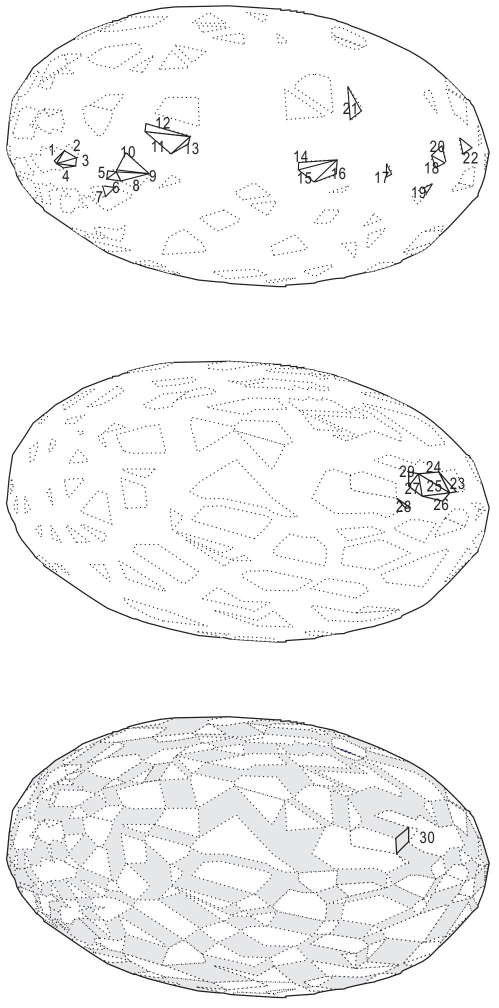


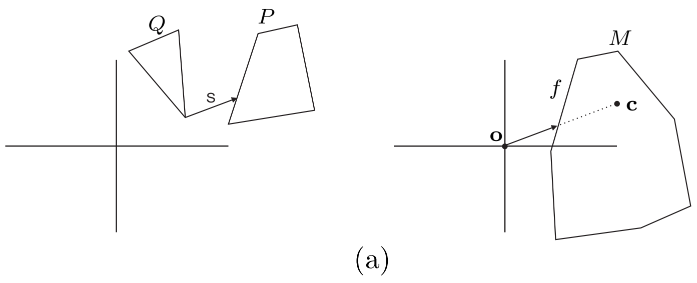
Figure 11: Search path on M. Top: In Search-FV (dotted regions contain faces in $\mathcal{F}_{fv}$); Middle: In Search-VF (dotted regions contain faces in $\mathcal{F}_{vf}$); and Bottom: In Search-EE (gray regions contain faces in $\mathcal{F}_{ee}$)

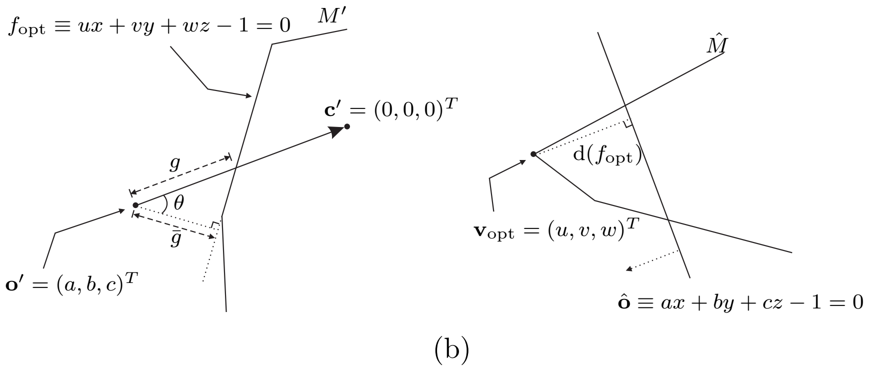
.

be the barycentric coordinates of $\mathbf{0}$ with respect to $\dot{\mathbf{r}}_0, \dot{\mathbf{r}}_j, \dot{\mathbf{r}}_{j+1}$. Then,

$$\begin{aligned}
\mathbf{0} &= u\dot{\mathbf{r}}_0 + v\dot{\mathbf{r}}_j + w\dot{\mathbf{r}}_{j+1} \\
&= u(\dot{\mathbf{p}}_0 - \dot{\mathbf{q}}_0) + v(\dot{\mathbf{p}}_j - \dot{\mathbf{q}}_j) + w(\dot{\mathbf{p}}_{j+1} - \dot{\mathbf{q}}_{j+1}) \\
&= (u\dot{\mathbf{p}}_0 + v\dot{\mathbf{p}}_j + w\dot{\mathbf{p}}_{j+1}) - (u\dot{\mathbf{q}}_0 + v\dot{\mathbf{q}}_j + w\dot{\mathbf{q}}_{j+1}) \\
&= \dot{\mathbf{p}} - \dot{\mathbf{q}},
\end{aligned}$$

where $\dot{\mathbf{p}} = u\dot{\mathbf{p}}_0 + v\dot{\mathbf{p}}_j + w\dot{\mathbf{p}}_{j+1}$ and $\dot{\mathbf{q}} = u\dot{\mathbf{q}}_0 + v\dot{\mathbf{q}}_j + w\dot{\mathbf{q}}_{j+1}$. Let $\mathbf{p} = u\mathbf{p}_0 + v\mathbf{p}_j + w\mathbf{p}_{j+1}$ and





$f_{\text{opt}} \equiv ux + vy + wz - 1 = 0$
M'
$\mathbf{c}' = (0,0,0)^T$
g
θ
$\bar{g}$
$\mathbf{o}' = (a,b,c)^T$
$\hat{M}$
$\mathrm{d}(f_{\text{opt}})$
$\mathbf{v}_{\text{opt}} = (u,v,w)^T$
$\hat{\mathbf{o}} \equiv ax + by + cz - 1 = 0$
(b)

Figure 12: (a) The minimum separating distance between P and Q along $\mathbf{s}$ when they are separate and the corresponding distance between $\mathbf{o}$ and $M = P \oplus \bar{Q}$; and (b) the relationship between the minimum separating distance g and the signed distance of $\mathrm{d}(f_{\text{opt}})$ in the dual space.

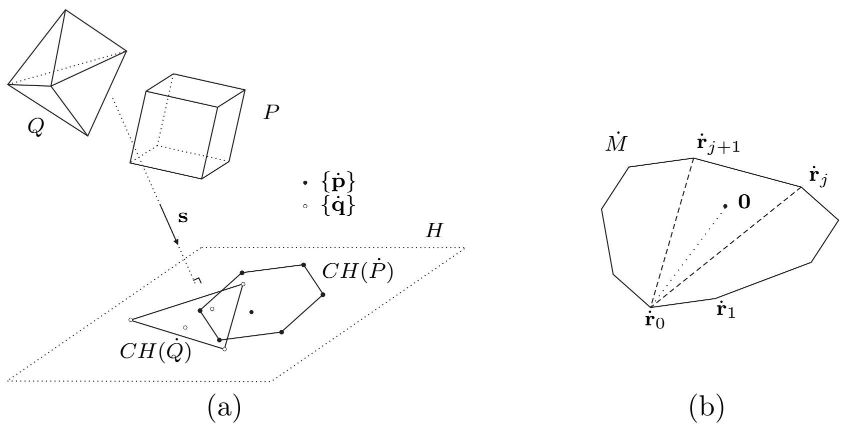


Figure 13: (a) Orthographic projection of P and Q along $\mathbf{s}$ to a plane H normal to $\mathbf{s}$; and (b) the Minkowski difference $\dot{M}$ of $CH(\dot{P})$ and $CH(\dot{Q})$ with the triangle $\triangle\dot{\mathbf{r}}_0\dot{\mathbf{r}}_j\dot{\mathbf{r}}_{j+1}$ containing the origin $\mathbf{0}$.

$\mathbf{q} = u\mathbf{q}_0 + v\mathbf{q}_j + w\mathbf{q}_{j+1}$, where $\mathbf{p}_0, \mathbf{p}_j, \mathbf{p}_{j+1}$ are vertices in P and $\mathbf{q}_0, \mathbf{q}_j, \mathbf{q}_{j+1}$ are vertices in Q that are projected to $\dot{\mathbf{p}}_0, \dot{\mathbf{p}}_j, \dot{\mathbf{p}}_{j+1}$ and $\dot{\mathbf{q}}_0, \dot{\mathbf{q}}_j, \dot{\mathbf{q}}_{j+1}$, respectively. Since the projection of P and Q along $\mathbf{s}$ to the plane H normal to $\mathbf{s}$ is an affine transformation that preserves ratio of area and therefore barycentric coordinates, $\dot{\mathbf{p}}$ and $\dot{\mathbf{q}}$ are the projected images of $\mathbf{p}$ and $\mathbf{q}$, respectively. We have $\mathbf{p} \in P$ and $\mathbf{q} \in Q$ because $u, v, w > 0$ and $u + v + w = 1$. Also $\mathbf{p} - \mathbf{q} \neq \mathbf{0}$ as P and Q are separate. Since $\dot{\mathbf{p}} = \dot{\mathbf{q}}$, we have $\mathbf{c} = \mathbf{p} - \mathbf{q} = k\mathbf{s}$, for some constant $k \neq 0$ and also $\mathbf{c} \in M = P \oplus \bar{Q}$. The minimum separating distance between P and Q is along $\mathbf{s}$ for $k > 0$ and along $-\mathbf{s}$ if $k < 0$.

As P and Q are separate, $\hat{\mathbf{o}}$ intersects $\hat{M}$ in $\hat{\mathbb{E}}^3_T$ and therefore the algorithm should be modified slightly so that it keeps on searching for f_{opt}, even if we have reached a face in M with positive signed distance. The distance between $\mathbf{o}$ and the intersection of f_{opt} with the line $\mathbf{oc}$ is then the required minimum separating distance.

Let g be the minimum separating distance. We would then establish the relationship between g and the signed distance $\mathrm{d}(f_{\text{opt}})$. Let $\mathbf{c}'$ be the origin of $\mathbb{E}^3_T$ as described in section 3.1. Let also $\mathbf{o}' = (a, b, c)^T$ (Figure 12(b)). Then $\hat{\mathbf{o}}$ is the plane given by $ax + by + cz - 1 = 0$. The face $f_{\text{opt}} \equiv ux + vy + wz - 1 = 0$ corresponds to the vertex $\mathbf{v}_{\text{opt}} = (\mathbf{u}, \mathbf{v}, \mathbf{w})^{\mathbf{T}}$ on $\hat{M}$ in $\hat{\mathbb{E}}^3_T$ and g is the distance between $\mathbf{o}'$ and the intersection of f_{opt} with the line $\mathbf{c}'\mathbf{o}'$. Now,

$$\mathrm{d}(f_{\text{opt}}) = \frac{au + bv + cw - 1}{\sqrt{a^2 + b^2 + c^2}}$$

Let $\bar{g}$ be the shortest distance from the point $\mathbf{o}'$ to the face f_{opt} and θ be the angle between the line $\mathbf{c}'\mathbf{o}'$ and the normal vector of f_{opt}. Then

$$\begin{aligned} \bar{g} &= \frac{ua + vb + wc - 1}{\sqrt{u^2 + v^2 + w^2}}, \quad \text{and} \\ \cos(\theta) &= \frac{au + bv + cw}{\sqrt{a^2 + b^2 + c^2}\sqrt{u^2 + v^2 + w^2}}. \end{aligned}$$

We have

$$\begin{aligned} g = \frac{\bar{g}}{\cos(\theta)} &= \frac{(ua + vb + wc - 1)\sqrt{a^2 + b^2 + c^2}}{au + bv + cw} \\ &= \frac{\mathrm{d}(f_{\text{opt}})(a^2 + b^2 + c^2)}{\mathrm{d}(f_{\text{opt}})\sqrt{a^2 + b^2 + c^2} + 1}. \end{aligned}$$

7 Conclusion

We have presented an efficient algorithm for detecting collision between two convex polyhedra P and Q. The algorithm is based on local searches in the three subsets $\mathcal{F}_{fv}$, $\mathcal{F}_{vf}$ and $\mathcal{F}_{ee}$ of all faces in the Minkowski sum $M = P \oplus \bar{Q}$. The search is guided by an objective function that is the signed distance of a face on M which is the signed distance of its dual vertex to a plane in the dual space. The function can be calculated easily. The global maximum signed distance of all faces on M indicates whether P and Q overlap. Due to the convexity of the dual of M, the face that attains the global maximum signed distance is guaranteed to be found by a local search on M. Moreover, by partitioning the set of faces of M into $\mathcal{F}_{fv}$, $\mathcal{F}_{vf}$ and $\mathcal{F}_{ee}$ and performing the local search in each partition successively, we are able to skip most of the $\mathcal{F}_{ee}$-type faces, which in general are of $\mathcal{O}(mn)$ in number where m and n are the number of vertices on P and Q, respectively. This then saves much of the computational time.

Experiments show that our algorithm is fast and takes constant time to finish, independent of penetration depth, when the two polyhedra touch or penetrate each other. It performs well especially when the two polyhedra are separate since the algorithm can terminate once the search reaches a face in M with positive signed distance. One drawback is that the algorithm takes longer time when M gets elongated in shape. Since different procedures are taken to deal with the three subsets of faces, more involved implementation work is also needed.

We have also shown how the algorithm can exploit geometric and temporal coherence in a dynamic environment. Although the algorithm does not compute the shortest distance between two polyhedra, it can be used to determine the minimum separating distance along a given direction in a natural way. We hope to extend the idea of the algorithm to deal with collision detection of convex bodies bounded by curved surface in the future.

References

[1] D. Baraff. Curved surfaces and coherence for non-penetrating rigid body simulation. *Computer Graphics*, 24(4): 19-28, 1990.

[2] G. van den Bergen. A fast and robust GJK implementation for collision detection of convex objects. *Journal of Graphics Tools*, 4(2):7–25, 1999.

[3] S. Cameron. A Comparison of Two Fast Algorithms for Computing the Distance Between Convex Polyhedra. *IEEE Transactions on Robotics and Automation*, 13(6):915–920, 1997.

[4] S. Cameron. Enhancing GJK: Computing Minimum and Penetration Distances between Convex Polyhedra. In *Proc. IEEE Int. Conf. on Robotics and Automation*, pp.3112–3117, 1997.

[5] J.D. Cohen, M.C. Lin, D. Manocha and M.K. Ponamgi. I-COLLIDE: An Interactive Detection System for Large-Scale Environments. In *Proc. ACM Interactive 3D Graphics Conf.*, pp. 189–196, 1995.

[6] D.P. Dobkin and D.G. Kirkpatrick. A linear algorithm for determining the separation of convex polyhedra. *Journal of Algorithms*, (6) 1985, pp. 381–392.

[7] D.P. Dobkin and D.G. Kirkpatrick. Determining the Separation of Preprocessed Polyhedra – A Unified Approach. In *Proc. 17th Int. Colloq. Automata Lang. Program*, LNCS 433, pp. 400–413, 1990.

[8] H. Eggleston. *Convexity*, Cambridge University Press 1966.

[9] E.G. Gilbert, D.W. Johnson, and S.S. Keerthi. A Fast Procedure for Computing the Distance Between Objects in Three-Dimensional Space. *IEEE Trans., Robotics and Automation*, RA(4)(1988), pp. 193–203.

[10] P. Jiménez , F. Thomas and C. Torras. 3D collision detection: a survey. *Computers and Graphics*, 25(2):269–285, 2001.

[11] M. Lin and D. Manocha. Fast Interference Detection Between Geometric Models. *Visual Comput.*, 11(10):542–561, 1995.

[12] M. Ponamgi, D. Manocha and M. Lin. Incremental Algorithms for Collision Detection Between General Solid Models. In *Proc. ACM Siggraph Sympos. Solid Modeling*, pp. 293–304, 1995.

[13] J. O'Rourke. *Computational Geometry in C*, Cambridge University Press 1993.

[14] J.G. Semple and G.T. Kneebone. *Algebraic Projective Geometry*, Oxford Science Publication 1952.

A Theorems and Proofs

Theorem 1 *The face $f_m \in \mathcal{F}(M)$, computed by* Search-FV*, attains the maximum signed distance among all faces in $\mathcal{F}_{fv}$, i.e.,* $\mathrm{d}(f_m) = \max\{\mathrm{d}(f) | f \in \mathcal{F}_{fv}\}$.

Proof: Consider the set of faces $\mathcal{F}_{fv}$ and its corresponding dual point set $\hat{\mathcal{F}}_{fv} = D \circ T(\mathcal{F}_{fv})$. If for every two faces $f_0, f_1 \in \mathcal{F}(P)$ that share an edge, we connect $\hat{F}(f_0, \mathbf{v}_0)$ and $\hat{F}(f_1, \mathbf{v}_1)$ by an edge, by the construction of $M \equiv P \oplus \bar{Q}$ and the properties of duality, we know that the point set $\hat{\mathcal{F}}_{fv}$ and the augmented edges form a polyhedron $\hat{W}$. Since $\hat{\mathcal{F}}_{fv} \subset \mathcal{V}(\hat{M})$ and $\hat{M}$ is convex, $\hat{W}$ must be convex too. Now, Search-FV searches locally for a vertex in $\hat{W}$ that attains the largest signed distance to the plane $\hat{\mathbf{o}}$. The search path also follows the adjacency of the faces in P and therefore is along the edges of $\hat{W}$. As $\hat{W}$ is convex, the dual vertex $\hat{f}_m$ (corresponding to the face f_m) attaining the local maximum that is returned by the procedure must be the one that attains the global maximum among all dual vertices in $\hat{\mathcal{F}}_{fv}$ (corresponding to the face set $\mathcal{F}_{fv}$). □

Theorem 2 *The face $f_m \in \mathcal{F}(M)$, computed by* Search-VF*, attains the maximum signed distance among all faces in $\mathcal{F}_{vf}$, i.e.,* $\mathrm{d}(f_m) = \max\{\mathrm{d}(f) | f \in \mathcal{F}_{vf}\}$.

Proof: Similar to proof of Theorem 1.

Recall the Gaussian image of a polyhedron as described previously, let us further denote by $S(\mathcal{F}_{fv})$, $S(\mathcal{F}_{vf})$ and $S(\mathcal{F}_{ee})$ the Gaussian images of the three types of faces $\mathcal{F}_{fv}$, $\mathcal{F}_{vf}$ and $\mathcal{F}_{ee}$, respectively.

Lemma 3 *Let $f = F(f', \mathbf{v}')$, $f' \in \mathcal{F}(P)$ (or $\mathcal{F}(\bar{Q})$) and $\mathbf{v}' \in \mathcal{V}(\bar{Q})$ (or $\mathcal{V}(P)$), be the face computed by* Search-FV *and* Search-VF *such that f attains the maximum signed distance among all faces in $\mathcal{F}_{fv} \cup \mathcal{F}_{vf}$. If $f \neq f_{\text{opt}}$, there exists at least a face $\tilde{f} = F(e_{f'}, e_{v'})$ where $e_{f'}$ is an edge of face f' and $e_{v'}$ is an edge incident at $\mathbf{v}'$, such that $\mathrm{d}(\tilde{f}) > \mathrm{d}(f)$.*

Proof: Without loss of generality, let $f = F(f_p, \mathbf{v}_{\bar{q}})$, $f_p \in \mathcal{F}(P)$ and $\mathbf{v}_{\bar{q}} \in \mathcal{V}(\bar{Q})$. Using the Gaussian image $G(M)$ of M, we now consider only vertices in the region $R_{\bar{Q}}(\mathbf{v}_{\bar{q}})$ and mark those vertices in $S(\mathcal{F}_{fv}) \cup S(\mathcal{F}_{vf})$ by white and those in $S(\mathcal{F}_{ee})$ by black as in Figure 14.

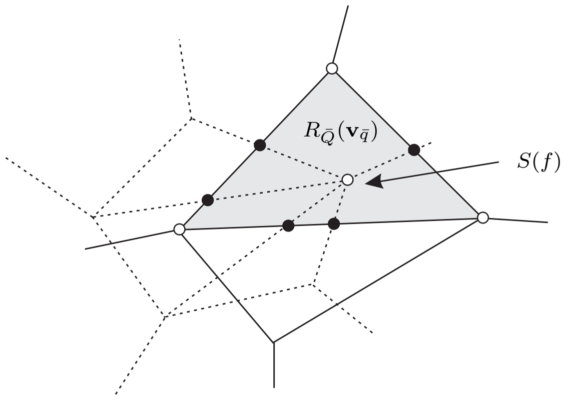


Figure 14: An illustration for proof of Lemma 3. $R_{\bar{Q}}(\mathbf{v}_{\bar{q}})$ is the grey region that $S(f)$ falls into; white vertices are in $S(\mathcal{F}_{fv}) \cup S(\mathcal{F}_{vf})$ while black vertices are in $S(\mathcal{F}_{ee})$.

Since $\mathrm{d}(f'') < \mathrm{d}(f)$ for all white vertices $S(f'')$ within $R_{\bar{Q}}(\mathbf{v}_{\bar{q}})$, there must be at least one black vertex $S(\tilde{f})$ within $R_{\bar{Q}}(\mathbf{v}_{\bar{q}})$ such that $\mathrm{d}(\tilde{f}) > \mathrm{d}(f)$; or otherwise, f will have the local maximum signed distance among all faces $\dot{f}$ with $S(\dot{f})$ in $R_{\bar{Q}}(\mathbf{v}_{\bar{q}})$. Due to the convexity of the signed distance function d, it also means that f attains the global maximum signed distance, contradicting $f \neq f_{\text{opt}}$. □

Lemma 4 *Let $\langle e_p, e_{\bar{q}} \rangle$ forms a face $\hat{f} = F(e_p, e_{\bar{q}}) \in \mathcal{F}_{ee}$. If $f \neq f_{\text{opt}}$ and $\mathrm{d}(f) > \max\{\mathrm{d}(\dot{f}) | \dot{f} \in \mathcal{F}_{fv} \cup \mathcal{F}_{vf}\}$, there exists a face $\tilde{f} = F(e_p, e'_{\bar{q}})$ or $\tilde{f} = F(e'_p, e_{\bar{q}}) \in \mathcal{F}_{ee}$ such that $\mathrm{d}(\tilde{f}) > \mathrm{d}(f)$, where $e'_p \in \mathcal{E}(P)$ is an edge incident at any of the two end vertices of e_p and $e'_{\bar{q}} \in \mathcal{E}(Q)$ is an edge incident at any of the two end vertices of $e_{\bar{q}}$.*

Proof: Let $S(f)$ be the Gaussian image of f in $G(M)$ (Figure 15). Then $S(f)$ is the intersection of arcs $A(e_p)$ and $A(e_{\bar{q}})$ (i.e., the Gaussian images of the edges e_p and

$e_{\bar{q}}$, respectively). For any neighbouring face f' of f in M sharing a common edge, the corresponding vertex in $G(M)$ must be connected to $S(f)$ by an arc. Hence, we just need to consider these neighbouring vertices of $S(f)$ which are divided into two types: those in $S(\mathcal{F}_{fv}) \cup S(\mathcal{F}_{vf})$ and those in $S(\mathcal{F}_{ee})$ which are marked by white and black in Figure 15, respectively.

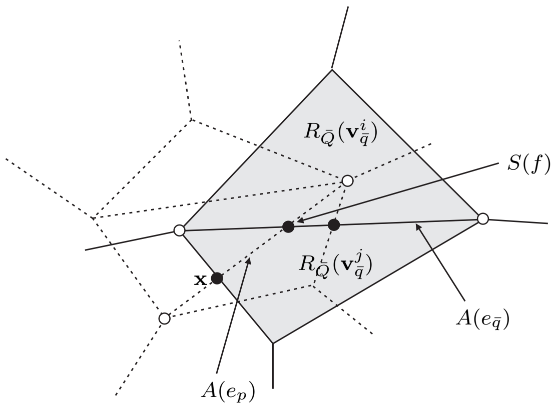

$R_{\bar{Q}}(\mathbf{v}^i_{\bar{q}})$
$S(f)$
$\mathbf{x}$
$R_{\bar{Q}}(\mathbf{v}^j_{\bar{q}})$
$A(e_{\bar{q}})$
$A(e_p)$

Figure 15: An illustration for proof of Lemma 4. Neighbouring vertices of $S(f)$; white vertices are in $S(\mathcal{F}_{fv}) \cup S(\mathcal{F}_{vf})$ while black vertices are in $S(\mathcal{F}_{ee})$. $A(e_p)$ exits $R_{\bar{Q}}(\mathbf{v}^i_{\bar{q}})$ and enters $R_{\bar{Q}}(\mathbf{v}^j_{\bar{q}})$ at $S(f)$.

Possible white vertices, i.e., both in $S(\mathcal{F}_{fv}) \cup S(\mathcal{F}_{vf})$ and is connected to $S(f)$ by an arc, can only be the end vertices of $A(e_p)$ and $A(e_{\bar{q}})$. On the other hand, possible black vertices, i.e., both in $S(\mathcal{F}_{ee})$ and is connected to $S(f)$ by an arc, can only be the intersections of $A(e_p)$ with some arcs in $G(\bar{Q})$ or the intersections of $A(e_{\bar{q}})$ with some arcs in $G(P)$.

Now, consider the neighbourhood of $S(f)$ along the arc $A(e_p)$. $A(e_p)$ must exit a region $R_{\bar{Q}}(\mathbf{v}^i_{\bar{q}})$ and enter a region $R_{\bar{Q}}(\mathbf{v}^j_{\bar{q}})$ in $G(\bar{Q})$. Therefore, vertices neighbouring to $S(f)$ along $A(e_p)$ must be the intersections of $A(e_p)$ and arcs bounding the regions $R_{\bar{Q}}(\mathbf{v}^i_{\bar{q}})$ and $R_{\bar{Q}}(\mathbf{v}^j_{\bar{q}})$ (e.g. vertex $\mathbf{x}$ in Figure 15). Since $\mathbf{v}^i_{\bar{q}}$ and $\mathbf{v}^j_{\bar{q}}$ are the two end vertices of the edge $e_{\bar{q}}$ in M, the faces corresponding to these intersections must be formed by the edge pairs $\langle e_p, e'_{\bar{q}} \rangle$, where $e'_{\bar{q}}$ are edges incident at the two end vertices of $e_{\bar{q}}$. Similarly, by considering neighbouring intersections of $S(f)$ along $A(e_{\bar{q}})$ in $G(M)$, the corresponding faces must be those by edge pairs $\langle e'_p, e_{\bar{q}} \rangle$, where e'_p are edges incident at the two end vertices of e_p. We have then for any white vertices $S(\dot{f})$, $\mathrm{d}(f) > \mathrm{d}(\dot{f})$. If for all black vertices $S(\tilde{f})$, $\mathrm{d}(\tilde{f}) < \mathrm{d}(f)$, f will attain the local maximum signed distance among all its faces in M. This means that f must attain the global maximum signed distance in M, which contradicts that $f \neq f_{\text{opt}}$. $\square$

Corollary 5 *Let $f \in \mathcal{F}_{fv} \cup \mathcal{F}_{vf}$, be the face that attains the maximum signed distance among all faces in $\mathcal{F}_{fv} \cup \mathcal{F}_{vf}$. If $f \neq f_{\text{opt}}$, we must have $f_{\text{opt}} = \tilde{f}$ where $\tilde{f} \in \mathcal{F}_{ee}$.*

In proving Lemma 3 and 4, when considering the Gaussian image of M, we only deal with the case where a vertex $S(f) \in S(\mathcal{F}_{fv}), f = F(f_p, \mathbf{v}_{\bar{q}})$ falls within a region $R_{\bar{Q}}(\mathbf{v}_{\bar{q}})$.

This corresponds to the case when $\mathbf{v}_{\bar{q}}$ is the supporting vertex of $\bar{Q}$ in the normal direction of f_p. In fact, the following two cases can happen too:

(2) $S(f)$ falls on an arc A bounding $R_{\bar{Q}}(\mathbf{v}_{\bar{q}})$, when the edge in $\mathcal{E}(\bar{Q})$ corresponds to A is parallel to f_p.

(3) $S(f)$ falls on a vertex $\mathbf{x}$ of $R_{\bar{Q}}(\mathbf{v}_{\bar{q}})$, when the face in $\mathcal{F}(\bar{Q})$ corresponds to $\mathbf{x}$ is parallel to f_p.

Nevertheless, we may apply the same arguments used in the proofs to $S(f)$ and each of the regions that A and $\mathbf{x}$ are adjacent to for case (2) and (3).

Theorem 6 *The face $f_m \in \mathcal{F}(M)$, computed by* SEARCH-EE*, attains the maximum signed distance among all faces in $\mathcal{F}(M)$, i.e., $f_m = f_{\text{opt}}$ and $\mathrm{d}(f_m) = d_{\max}$.*

Proof: The procedure SEARCH-EE first starts with a face $f = F(f', v')$ that attains the maximum signed distance among all faces in $\mathcal{F}_{fv} \cup \mathcal{F}_{vf}$. It then searches for the set of faces $\mathcal{S}$ formed by the edges of f' and the edges incident at v'. If $f \neq f_{\text{opt}}$, by Lemma 3, there must exist a face in $\tilde{f} \in \mathcal{S}$ such that $\mathrm{d}(\tilde{f}) > \mathrm{d}(f)$ and the searching process will then continue from $\tilde{f}$; otherwise, SEARCH-EE returns $f_m = f$.

When SEARCH-EE reaches a face $\dot{f} = F(e_p, e_{\bar{q}})$, it next searches for the set of faces $\acute{\mathcal{S}}$ formed by the edge e_p (or $e_{\bar{q}}$) and edges incident at the end vertices of $e_{\bar{q}}$ (or e_p). Lemma 4 guarantees that if $\dot{f} \neq f_{\text{opt}}$, there must exist at least a face $\acute{f} \in \acute{\mathcal{S}}$ such that $\mathrm{d}(\acute{f}) > \mathrm{d}(\dot{f})$. Also by Corollary 8, face f_{opt} must be in $\mathcal{F}_{ee}$. At each subsequent steps, f_m approaches f_{opt} and eventually reaches f_{opt} such that $\mathrm{d}(f_m) = \mathrm{d}(f_{\text{opt}}) = d_{\max}$. □