Zonotopes as Bounding Volumes

Leonidas J. Guibas* An Nguyen* Li Zhang†

Abstract

Zonotopes are centrally symmetric polytopes with a very special structure: they are the Minkowski sum of line segments. In this paper we propose to use zonotopes as bounding volumes for geometry in collision detection and other applications where the spatial relationship between two pieces of geometry is important. We show how to construct optimal, or approximately optimal zonotopes enclosing given set of points or other geometry. We also show how zonotopes can be used for efficient collision testing, based on their description via their defining line segments — without ever building their explicit description as polytopes. This implicit representation adds flexibility, power, and economy to the use of zonotopes as bounding volumes.

1 Introduction

Zonotopes have long been studied in combinatorial geometry, polyhedral combinatorics, algebraic geometry, and other parts of mathematics. Yet, except for their use in solving systems of polynomial equations [13], their usefulness in applications of interest to science and engineering has been limited. In this paper we propose to develop the use of zonotopes (and especially zonogons and zonohedra, the $\mathcal{R}^2$ and $\mathcal{R}^3$ cases) as versatile bounding volumes for pieces of underlying geometry in modeling applications. Bounding volumes are useful in collision detection, distance computation, penetration depth computation, surface fitting, and many other geometric processes.

A zonotope Z is defined by line segment generators $s_1, s_2, \ldots, s_n$ in $\mathcal{R}^d$. The zonotope is simply the Minkowski sum of its line segment generators. Equivalently, a zonotope is simply an affine image of the unit cube from $\mathcal{R}^n$ to $\mathcal{R}^d$. It is obviously a convex polytope and its facets are parallelepipeds defined by $(d-1)$-tuples of its generators. Note that a zonotope must be a centrally symmetric convex polytope. While in 2-D any centrally symmetric convex polygon is a zonogon, that is no longer the case in 3-D or higher dimensions. However, many familiar polyhedra are zonotopes, including cubes and parallelepipeds, truncated octahedra, and rhombic dodecahedra. In 3-D the facets of a zonotopes are parallelograms defined by pairs of generators. The collection of all those facets sharing a particular segment generator form a band (zone) wrapping around the zonotope — a fact which justifies the name *zonotope*.

Since a zonotope is always centrally symmetric, every facet has an opposite congruent facet on the other side. The combinatorics of the faces of a zonotope are equivalent to those of the vertices in an arrangement of great hypercircles in a sphere of one less dimension. This can be most easily seen by considering the space of d-dimensional hyperplanes tangent to the zonotope. The space of all their d-dimensional normal unit vectors can be seen as a unit sphere, equivalent to an oriented projective $(d-1)$-space. For any given unit vector, there is a unique hyperplane normal to the vector and tangent to the zonotope at some face. Under this map each generator gives rise to a great circle; thus the facets of the zonotope are in 1-1 correspondence with the vertices of this spherical arrangement under this tangent space map. Note that all facets belonging to a zone, map to the vertices on the great circle defined by their shared generator.

Spherical arrangements can be mapped to hyperplane arrangements by a simple projective map. Because of these two correspondences, we can both estimate the size and construct zonotopes by using the corresponding classical bounds for hyperplane arrangements. In particular, zonotopes in $\mathcal{R}^d$ have complexity (including number of facets) that is $O(n^{d-1})$, and can be constructed within the same time bound. In particular, a zonohedron may have complexity $O(n^2)$.

A variety of bounding volumes have been used for collision detection. These include axis-aligned bounding boxes [4], bounding boxes in general orientation (OBBs) [10], spheres [12, 17], and many more. Note that the first two are in fact special cases of zonotopes. In general, the trade-off involved in selecting a bounding volume shape is between the tightness of fit for the underlying geometry and the simplicity of testing the

*Department of Computer Science, Stanford University, Stanford, CA 94305. E-mail: guibas, anguyen@cs.stanford.edu.

†Systems Research Center, Hewlett-Packard Labs, 1501 Page Mill Road, Palo Alto, CA 94304. E-mail: l.zhang@hp.com.

intersection between bounding volumes. All currently proposed bounding volumes are shapes of constant description complexity, that is, each of them is defined by a fixed number of parameters.

Several facts make zonotopes an intriguing possibility as a bounding volume:

- Zonotopes are closed under Minkowski sum and difference; this implies that testing for intersection between two zonotopes can be implemented by testing for point inclusion in their Minkowski difference.
- The list of generators is an efficient *implicit* representations of the zonotope; for example, in $\mathcal{R}^3$, a zonotope of size $O(n^2)$ can represented by only n generators. Furthermore, operations such as Minkowski sum and difference are trivial to express in terms of generator lists.
- Zonotopes allow for bounding volumes of *variable complexity*, within a unified framework. For example, when constructing a bounding volume hierarchy, one can use zonotopes with more generators at higher levels in the hierarchy, where there are few hierarchy nodes but the complexity of the enclosed geometry may lead to a bad fit using only few generators. As we will show, mixing space and space-time volumes [3, 11, 12] is another example.

2 Summary of the Results

In this paper we present efficient algorithms for finding tight zonotopes enclosing some underlying polyhedral geometry and for using zonotopes as bounding volumes in collision detection applications. Throughout, we aim to represent zonotopes via their collection of generators, and not as explicit polytopes.

Specifically:

- We give an $O(n \log^2 n)$ algorithm for computing the minimum area zonotope enclosing a set of n points in $\mathcal{R}^2$.
- We give an algorithm to find a zonotope enclosing n given points in $\mathcal{R}^d$ with generators along k given directions and minimizing the sum of the generator lengths, in time $O(nk^{d-1} + k^{O(d)})$.
- We give an algorithm to find a zonotope enclosing n given points in $\mathcal{R}^d$ whose total generator length is within $(1 + O(\epsilon))$ of the optimum, in time $O(n\epsilon^{-(d-1)^2} + \epsilon^{-O(d^2)})$.

Furthermore, given zonotopes in $\mathcal{R}^3$ specified by their generators, we can:

- Decide whether two zonotopes with n generators in total intersect or not, in time $O(n \log^2 n)$ time.
- When repeated intersection testing is required, as in physical simulations, we describe how to implement efficiently some of the classical methods (such bounding volume hierarchies, or tracking closest feature pairs) using only the implicit description of zonotopes via their generators.
- We show how to easily build bounding *space-time volumes* for zonotopes, for use in collision detection applications where it is critical that no collisions be missed.

Although many questions remain open, the developments in this paper show the potential benefits of using zonotopes in other areas of science and engineering.

3 Zonotope Fundamentals

Formally, a zonotope is a Minkowski sum of a finite set of line segments. An alternative view is to define a zonotope by its center and generator vectors. The zonotope Z centered at p, with generators $v_1, v_2, \cdots, v_n$, is the point set $\{p + \sum_i \delta_i v_i |, -1 \leq \delta_i \leq 1, \text{ for all } 1 \leq i \leq n\}$. We write $Z = (p, \langle v_1, v_2, \cdots, v_n \rangle)$. For simplicity, we assume throughout the paper that the zonotopes are non-degenerate, i.e. any d generators are linearly independent. In d-dimensions, a zonotope with n generators has complexity $O(n^{d-1})$. As we remarked, the topology of the boundary of a zonotope matches the topology of a line arrangement [19]. We describe this duality in three dimensions.

In three dimensions, the faces on a zonotopes are parallelograms. Let us take the z-axis as pointing up. The boundary of a zonotope can be decomposed into two pieces: the upper hull Z^+ and the lower hull Z^-. We project Z^+ on the xy plane and obtain the tiling $\mathcal{T}$ of a convex polygon Q, the projection of the vertical silhouette of Z^+. Each tile T_{ij} in $\mathcal{T}$ is the projection of a face on Z^+ and is a translation of parallelogram $\{xv_i' + yv_j' \mid -1 \leq x, y \leq 1\}$ where v_i' denotes the projection of v_i on the xy-plane. Now consider the plane $P = \{z | z = 1\}$. For each generator v_i of Z, we draw a plane P_i passing through the origin and perpendicular to v_i. Let ℓ_i be the line $P_i \cap P$. Denote by Π the arrangement of $\{\ell_i \mid 1 \leq i \leq n\}$. The dual diagram of $\mathcal{T}$ is isomorphic to the line arrangement Π: each parallelogram T_{ij} in $\mathcal{T}$ corresponds to a vertex between ℓ_i and

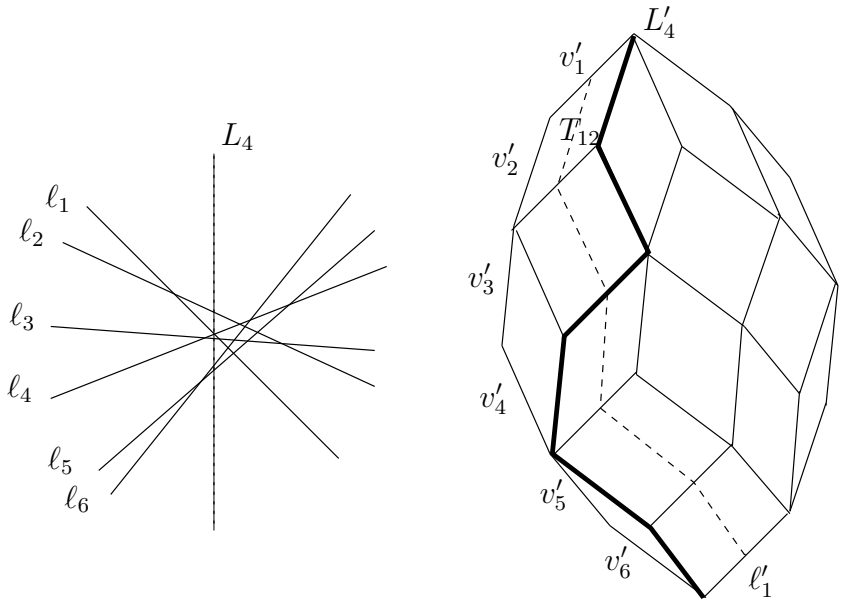

Figure 1: The dual between the line arrangement Π and the tiling $\mathcal{T}$.

ℓ_j; each vertex in $\mathcal{T}$ to a face in Π; and each edge in $\mathcal{T}$ to an edge in Π (Figure 1).

In the later sections, we will exploit this duality between line arrangements and zonotopes to design efficient algorithms for zonotope intersection testing.

4 Smallest Enclosing Zonotopes

In this section we focus on algorithms for computing smallest enclosing zonotopes for some underlying geometry in $\mathcal{R}^d$. Many possible definitions of 'smallest' can be used, including volume, surface area, total length of generators, etc. A first basic observation is that it suffices to consider only the convex hull of the underlying geometry, since a zonotope is convex. Since we are only concerned with polyhedral geometry, from now on we will focus on computing the optimal enclosing zonotope of a set of points, which we may assume to be the vertices of a given convex polytope. We consider the cases $d = 2$ and $d \geq 3$ separately.

4.1 Minimum Area Enclosing Zonotope in $\mathcal{R}^2$

In $\mathcal{R}^2$ we can give a fast algorithm to compute the minimum area enclosing zonotope. Our input can be assumed to be a convex polygon $\mathcal{H}$ of n vertices $P_i, 1 \leq i \leq n$, in $\mathcal{R}^2$. We show that the smallest area zonotope $\mathcal{Z}$ that contains all the points P_i can be computed in $O(n \log^2 n)$ time.

We first look at a much simpler problem, when the center O of the zonotope is specified. In $\mathcal{R}^2$, a zonotope is simply a centrally symmetric polygon, or *zonogon*. If the zonogon has a center at O and contains all the points P_i, it must also contain all the reflection points P'_i of P_i through O, and thus it contains the convex hull of the set of $2n$ points P_i and P'_i. This convex hull is centrally symmetric around O, and so it is the minimum area zonogon centered at O and containing all the points P_i. We call this zonogon $\mathcal{Z}(O)$.

In the general setting, only the points P_i are given. We need to find the center O that minimize the area of $\mathcal{Z}(O)$. For notational simplicity, we will allow indices outside the range $[1 \,..\, n]$ and identify P_i with P_{i+n} and P_{i-n} for each i.

For a given center O, the vertices of $\mathcal{Z}(O)$ are either original points P_i or reflected points P'_i. By grouping the original points together, and respectively, the reflected points, we can describe $\mathcal{Z}(O)$ as a circular sequence of vertices in a counterclockwise order of the form: $\mathcal{S} = P_{a_1} ... P_{b_1} P'_{a'_1} ... P'_{b'_1} \cdots P_{a_k} ... P_{b_k} P'_{a'_k} ... P'_{b'_k}$. We call this sequence the *combinatorial description* of $\mathcal{Z}(O)$.

We denote $[\mathcal{P}]$ the area of a polygon $\mathcal{P}$, and define the function $f : \mathcal{R}^2 \to \mathcal{R}$, $f(O) = [\mathcal{Z}(O)]$, for each point $O \in \mathcal{R}^2$. Note that given $\mathcal{H}$ and O, we can construct $\mathcal{Z}(O)$ in $O(n)$ time, and thus f can be evaluated at any point $O \in \mathcal{R}^2$ in $O(n)$ time. To find the global minimum of f, we first establish some properties of f.

In a region where the combinatorial description of $\mathcal{Z}(O)$ is a constant $\mathcal{S}$, exploiting the symmetry of

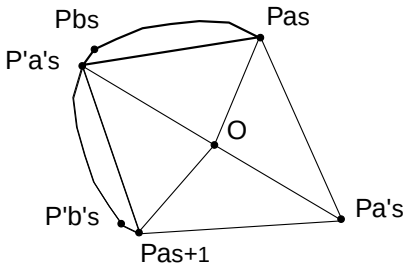

Figure 2: A part of a zonotope

$\mathcal{Z}(O)$, we have that, see Figure 2:

$$\begin{aligned} f(O) &= \sum_{s=1}^{k} \Big([OP_{a_s}P'_{a'_s}P_{a_{s+1}}] + \\ &\quad [P_{a_s}...P_{b_s}P'_{a'_s}] + [P'_{a'_s}...P'_{b'_s}P_{a_{s+1}}] \Big) \\ &= \sum_{s=1}^{k} \Big([P_{a'_s}P_{a_s}OP_{a_{s+1}}] + 2\,[P_{a_s}...P_{b_s}P'_{a'_s}] \Big) \\ &= \sum_{s=1}^{k} \Big([P_{a'_s}P_{a_s}P_{a_{s+1}}] - [OP_{a_s}P_{a_{s+1}}] \\ &\quad +4\,[P_{a_s}...P_{b_s}O] - 2\,[P_{a_s}...P_{b_s}P_{a'_s}] \Big) \\ &= \sum_{s=1}^{k} \Big([P_{a'_s}P_{a_s}P_{a_{s+1}}] + 4\,[P_{a_s}...P_{b_s}O] \\ &\quad -2\,[P_{a_s}...P_{b_s}P_{a'_s}] \Big) - [P_{a_1}...P_{a_k}]\,. \end{aligned}$$

It follows from the above equation, in each region where $\mathcal{S}$ is a constant, f is an affine function of O. The coefficient of the linear term of that affine function comes from the term $4\,[P_{a_s}...P_{b_s}O]$ and thus, depends only on the edges of $\mathcal{H}$ appearing on $\mathcal{Z}(O)$.

For an edge P_tP_{t+1} of $\mathcal{H}$, let s be such that P_s be the point furthest from P_tP_{t+1} among all the points P_i. Let ℓ_t be the directed line connecting the midpoint of P_sP_t to the midpoint of P_sP_{t+1}.

It is easy to see that the edge P_tP_{t+1} of $\mathcal{H}$ is an edge of $\mathcal{Z}(O)$ iff all the points P'_i are on the left side of the directed line P_tP_{t+1}, i.e. iff P'_s is on the left side of P_tP_{t+1}, and thus, iff O is on the left side of ℓ_t. If $\mathcal{A}$ denotes the arrangement of the lines ℓ_t, then it is clear that f is a piecewise affine function over $\mathcal{A}$. We have thus shown:

Lemma 4.1

The area function f is piecewise affine, and its domain decomposition is given by an arrangement of n lines.

Given a line ℓ, let us consider the restriction f_ℓ of f onto ℓ. It is clear that f_ℓ is a piecewise affine function as well. We can compute the intersections of ℓ with the lines ℓ_t, and sort these intersections in $O(n \log n)$ time. After that, when O moves along ℓ, we can track the edges of $\mathcal{H}$ appearing on or disappearing from $\mathcal{Z}(O)$. This way, we can easily compute the function f_ℓ in $O(n)$ additional time. Thus,

Lemma 4.2

The restriction of f onto any line ℓ can be computed in $O(n \log n)$ time.

By rotating the plane, we can assume, without lost of generality, that the line ℓ is a vertical line. We further assume that the line ℓ and the points P_i are in general position, and thus ℓ is not parallel to any of the edges in $\mathcal{H}$. When the point O is at $-\infty$ along ℓ, the edges in $\mathcal{H}$ appearing on $\mathcal{Z}(O)$ are edges on the upper hull of $\mathcal{H}$. When O moves upward and crosses one of the line ℓ_t, if that line corresponds with an edge in the upper hull of $\mathcal{H}$, that edge disappears from $\mathcal{Z}(O)$. If the line ℓ_t corresponds with an edge in the lower hull of $\mathcal{H}$, that edge appears on $\mathcal{Z}(O)$. Observe that in both cases, the slope of f_ℓ increases. As the result, the slope of f_ℓ is monotonically increasing. Thus,

Lemma 4.3

The restriction f_ℓ of f onto any line ℓ is a unimodal function.

It is well known that a planar function which is unimodal over all lines in its domain is itself unimodal. As a direct consequence,

Corollary 4.4

The function f is unimodal.

Let ℓ be a vertical line, and let O_ℓ be the point where f_ℓ achieves its minimum. From Lemma 4.2, O_ℓ can be computed in $O(n \log n)$ time. If we compute the linear coefficient of f at O_ℓ, we can decide whether O_ℓ is the global minimum of f, and if not, using the unimodal property of f, we can tell whether the global minimum must be on the left or the right of ℓ.

As f is piecewise affine on $\mathcal{A}$, it has a global minimum at one of the vertices of $\mathcal{A}$. We can use binary search, with the help of a slope selection algorithm [5], to locate that vertex. There are $O(n^2)$ candidate vertices at the beginning. In each search step, we reduce the number of candidate vertices in half, by first running the slope selection algorithm to obtain a vertical line separating the set of candidates into 2 subsets of

equal size, then decides which of the subset contains the global minimum. There are $O(\log n)$ search steps, each costing $O(n \log n)$ time for running the slope selection algorithm to obtain a vertical line, and another $O(n \log n)$ time to decide what side of that vertical line the optimal vertex is on. The total cost of locating the global minimum of f is $O(n \log^2 n)$. Thus,

Theorem 4.5 *The minimum area zonotope containing a set of n points in $\mathcal{R}^2$ can be computed in $O(n \log^2 n)$.*

In $\mathcal{R}^d$, $d \geq 3$, the computation of the optimal enclosing zonotope for a point set is more complicated, because not every centrally symmetric polyhedral body is a zonotope. In fact, most centrally symmetric bodies cannot be approximated arbitrarily closely by zonotopes — those that can are know as *zonoids*. Unlike in $\mathcal{R}^2$, we know of no simple way to find an optimal enclosing zonotope even when the zonotope center is given. The volume of a zonotopes in $\mathcal{R}^d$ is also more complicated, being the sum of the volumes of all possible parallelepiped formed by d-tuples of the generators. So for $d \geq 3$ we consider two simpler problems. In both problems, we use the sum of the total length of the generators as the measure of optimality. In the first problem, we consider the case where the directions of the generators are given, and in the second problem, we consider the task of finding an approximately optimal enclosing zonotope.

4.2 Discrete Oriented Enclosing Zonotope

Given a set of points $P = \{p_1, p_2, ..., p_n\}$, and a set of unit vectors $V = \{v_1, v_2, ..., v_k\}$, we would like to compute a point p and coefficients $c_1, c_2, ..., c_k$ such that the discrete oriented zonotope $\mathcal{Z}_d = (p, \langle c_1 v_1, c_2 v_2, ..., c_k v_k \rangle)$ contains all points in P.

What makes this problem easier is the fact that the combinatorial structure of a zonotope depends on the direction of its generating vectors, and not on the their length, and thus the combinatorial structure of all discrete oriented zonotopes with the same direction vector set are the same. We can compute the hyperplane arrangement dual to $\mathcal{Z}_d$ in $O(k^{d-1})$ time, then, for any $(d-1)$-tuple of directions, we can find the normal direction of the two zonotope faces corresponding to that tuple, and find the two extreme points among P along that direction. It is clear that we only need to look at these extreme points when computing $\mathcal{Z}_d$.

By rearranging the points if necessary, we can assume without lost of generality that the first n' points $p_1, p_2, ..., p_{n'}$ are the extreme points, $n' = O(k^{d-1})$.

To compute the coefficients c_i's, we solve the following linear programming:

$$min \sum_{i=1}^{k} c_i$$
subject to:
$$p_j = p + \sum_{i=1}^{k} b_{ij} v_i, \ \forall j, \ 1 \leq j \leq n'$$
$$-c_i \leq b_{ij} \leq c_i, \ \forall i, j, \ 1 \leq i \leq k, 1 \leq j \leq n'$$

We can solve this linear programming in polynomial time using interior methods [18]. There are $O(k^d)$ equations and constraints, and thus, the cost of solving this linear programming is $O(k^{O(d)})$. Thus,

Theorem 4.6 *The minimum total length discrete oriented zonotope $\mathcal{Z}_d$ having generators along k given directions and containing a given set of n points can be computed in $O(nk^{d-1} + k^{O(d)})$ time.*

4.3 Approximate Minimum Total Length Zonotope

In this subsection, we would like to compute a zonotope $\mathcal{Z}_a$ containing the set of points $P = \{p_1, p_2, ..., p_n\}$ such that the total length of the generators of $\mathcal{Z}_a$ is within $O(\epsilon)$ of the the total length of the minimum total length zonotope $\mathcal{Z} = (p, \langle w_1, w_2, ..., w_r \rangle)$ enclosing P.

We consider the set of unit vectors $S = \{v_1, v_2, ..., v_k\}$ that tessellates the sphere of directions so that for any unit vector u, there is a vector v_j in S such that $|u - v_j| < \epsilon$. It is clear that we can do so with $O(1/\epsilon^{d-1})$ vectors. Let $e_1, e_2, ..., e_d$ be the unit vectors along the axes of some coordinate system, and let $R = S \cup \{e_1, e_2, ..., e_d\}$.

For each generator vector w_i of $\mathcal{Z}$, let v_{ij} be a vector in S such that $w_{ij} = w_i/|w_i| - v_{ij}$ satisfies $|w_{ij}| < \epsilon$. Let $D = \sum_{i=1}^{r} |w_i|$. It is clear that $\left| \sum_{i=1}^{r} |w_i| \, w_{ij} \right| \leq \sum_{i=1}^{r} |w_i| \, |w_{ij}| \leq \epsilon D$, and thus

$$\begin{aligned} \mathcal{Z} &\subset p \oplus \oplus_{i=1}^{r} w_i \\ &\subset p \oplus \oplus_{i=1}^{r} (|w_i| \, v_{ij}) \oplus \oplus_{i=1}^{k} (|w_i| \, w_{ij}) \\ &\subset p \oplus \oplus_{i=1}^{r} (|w_i| \, v_{ij}) \oplus \oplus_{i=1}^{d} (\epsilon D e_d) \end{aligned}$$

Thus, $\mathcal{Z}$ is contained inside a discrete oriented zonotope with direction vector set R. We call this zonotope $\mathcal{Z}'$. Clearly, the total length of generators in $\mathcal{Z}'$ is $(1 + d\epsilon)D = (1 + O(\epsilon))D$. We compute $\mathcal{Z}_a$ with direction vector set R having the minimum total length.

The total length of generating vectors of $\mathcal{Z}_a$ is less than the total length of $\mathcal{Z}'$, and thus is within $O(\epsilon)$ the total length of $\mathcal{Z}$. Thus,

Theorem 4.7 *Given a set of n points in $\mathcal{R}^d$, and an $\epsilon > 0$. An approximate enclosing zonotope of the point set with total length of generators within $O(\epsilon)$ of the optimal one can be computed in $O(n\epsilon^{-(d-1)^2} + \epsilon^{-O(d^2)})$.*

5 Collision Detection Between Two Zonotopes

We now describe algorithms for testing if two zonotopes intersect in $\mathcal{R}^3$. We consider two scenarios. One is the static collision detection in which we only need to detect the collision between two static zonotopes. The other is the dynamic collision detection in which repetitive collision detections are needed between two zonotopes that may be in different position or orientation. In latter case, preprocessing is allowed to accelerate the subsequent collision detection. Dynamic collision detection arises in the applications dealing with dynamic scenes such as moving objects. All of these problems have been studied extensively for convex bodies. Of course, a zonotope is a convex object. Any algorithm for convex objects applies to zonotopes as well. However, the explicit representation of a zonotope with n generators needs $\Theta(n^2)$ storage. Direct application of the existing algorithms to zonotopes becomes inefficient. Therefore, the major challenge is to design efficient algorithms that work for implicitly represented zonotopes. We show in this section that many algorithms developed for convex bodies have efficient counterparts for zonotopes.

5.1 Static collision detection

For this problem we need to 'anchor' zonotopes at particular points of space. Thus we will specify zonotopes by giving their center, followed by a list of their line segment generators. We treat the segment generators as vectors emanating from the origin. Note that when the center coincides with the origin, a zonotope coincides with its centrally symmetric reflection through the origin.

Given two zonotopes $Z_1 = (p, \langle v_1, \cdots, v_k \rangle)$ and $Z_2 = (q, \langle w_1, \cdots, w_m \rangle)$, we wish to decide whether Z_1 intersects Z_2, i.e. whether $Z_1 \cap Z_2 = \emptyset$. The following is well-known.

Lemma 5.1 *$Z_1 \cap Z_2 \neq \emptyset$ iff $q - p$ is in the zonotope $(0, \langle v_1, \cdots, v_k, w_1, \cdots, w_m \rangle)$.*

The above lemma reduces the collision detection between zonotopes to the point membership problem of a zonotope: given a point p and a zonotope $Z = (0, \langle u_1, u_2, \cdots, u_n \rangle)$ (here $n = k + m$), determine whether $p \in Z$. Of course, we may compute the explicit representation of this zonotope and apply a standard algorithm for point inclusion convex bodies. This algorithm, however, will run in at least $\Omega(n^2)$ time as the number of vertices of a zonotope can be quadratic in terms of the number of generators. In this section, we present an algorithm for intersection detection with only $O(n \log^2 n)$ running time.

Recall that the boundary of Z can be decomposed into the upper hull Z^+ and the lower hull Z^-. A point is in Z iff it is directly below Z^+ and directly above Z^-. We therefore further reduce the problem to determining whether a point is directly below Z^+ or/and above Z^- (the problems are symmetric). Let Q be the boundary of the projection of Z^+ and the tiling $\mathcal{T}$ of Q be the projection of Z^+ on the xy plane (Figure 1). For a point p, consider its projection p_0 on the xy plane. If p_0 is outside of Q, p is not directly below Z^+. Otherwise, we locate the parallelogram T that contains p_0 and decide if p is above or below the corresponding facet on Z^+.

The tiling $\mathcal{T}$ can be viewed as a monotone planar subdivision. We will use a method similar to [8] to locate the point. Namely, we perform a binary search on the monotone separators to determine the two adjacent separators that sandwich the point. We will show below that each separator consists of n line segments and can be computed in time $O(n \log n)$. Since we perform $O(\log n)$ comparisons against separators in total, the algorithm runs in time $O(n \log^2 n)$. In what follows, we shall show how to construct a separator in $O(n \log n)$ time.

We will now exploit the previously mentioned tangent space duality between zonotopes and arrangements. For a zonotope Z, let Π be the line arrangement defined in Section 3. Order all the vertices in Π according to their x coordinates, and index the vertices according to their order. We can assign the same index to the corresponding parallelogram in $\mathcal{T}$. Clearly, the order is consistent with the "right-to" relationship[1] in $\mathcal{T}$. Denote by L_k the vertical line that just right to the k-th vertex in Π. Suppose that L_k crosses the lines $\ell_{i_1}, \ell_{i_2}, \cdots, \ell_{i_n}$ from left to right (Figure 1). Define the corresponding pseudo vertical line L'_k in the tiling $\mathcal{T}$ as follows. We start from the top vertex of Q and form a monotone chain S by extending $v'_{i_1}, v'_{i_2}, \cdots, v'_{i_n}$ one

[1] We use "right-to" instead of "above" notation just for exposition convenience.

by one. Since L_k separates all the vertices with indices $\leq k$ from those $> k$, L'_k separates all the faces with indices $\leq k$ from those $> k$, i.e L'_k is the k-th separator, from left to right, in $\mathcal{T}$. We can compute L_k in $O(n \log n)$ time by the optimal slope selection algorithm [5]. Therefore, we have that:

Theorem 5.2 *For any two zonotopes with n generators in total, we can decide whether they intersect in $O(n \log^2 n)$ time.*

5.2 Dynamic collision detection

In dynamic collision detection, we may need to perform intersection testing for two zonotopes repetitively when they are in different configurations. Dynamic collision detection has been a central subject in motion planning, dynamic simulation, and computer animation. The typical methods include Minkowski sum method for translational motions, hierarchical method, and local walking techniques. We will describe how to implement those methods efficiently for zonotopes.

Minkowski sum method. When only translation is allowed, one standard technique is the Minkowski sum method[2]. In such a method, we compute the Minkowski sum of two convex objects and reduce the collision detection problem to a point containment problem in the Minkowski sum. According to the earlier duality, this is very similar to point location in line arrangements. Point location in line arrangements is a very well studied topic in Computational Geometry. The best known trade-off between preprocessing and query time is roughly $O(\frac{n}{\sqrt{m}})$ query time by using $O(m)$ preprocessing time and space. Unfortunately, we are unable to achieve the same bound for our problem. Instead, we have the following weaker trade-off.

Lemma 5.3 *For any zonotope with n generators, for any $n \leq m \leq n^2$, we can preprocess it into a data structure with $O(m)$ space so that the membership query can be answered in time $O(\frac{n^2}{m} \log^2 n)$.*

Proof: We compute a $(1/r)$-cutting Δ of the dual arrangement: a set of $O(r^2)$ interior disjoint trapezoids that refine the arrangement of r lines so that each trapezoid is crossed by $O(n/r)$ lines. We then map the cutting to the tiling $\mathcal{T}$ of Q. Each line ℓ_i is mapped to a pseudo line ℓ'_i which is the bisector of the strip corresponding to the generator v_i (Figure 1). We perturb each vertical thread to its right and map to its corresponding pseudo vertical line as defined before. This way, we obtain a planar subdivision Δ' of Q. Each cell in the subdivision corresponds to a trapezoid in Δ. Further, the line-trapezoid incidence relationship in (Π, Δ) is preserved. Thus, we can compute and store a $O(\log n)$ query time point location data structure for Δ' and associate the crossing lines with each cell in Δ'. For any query point, we first locate it in Δ' and then use the algorithm as shown in Theorem 5.2 to locate the point. The first step takes $O(\log n)$ time, and the second step takes $O(\frac{n}{r} \log^2 n)$ time as each cell is crossed by $O(n/r)$ lines. As for the preprocessing, since each pseudo line has complexity $O(n)$, the complexity of the arrangement of r pseudo lines is $O(nr)$. In addition, each cell needs to store $O(n/r)$ lines and there are $O(r^2)$ cells. The storage in total is $O(nr)$. By setting $r = m/n$, we obtain the bound as claimed. ∎

The above algorithms can also be used to compute the polyhedral distance defined by the zonotope.

Corollary 5.4 *Given a polyhedral metric defined by a zonotope with n generators, for any $n \leq m \leq n^2$, we can construct a data structure using $O(m)$ storage and in $O(m \log n)$ time, so that the distance between any two points can be computed in time $O(\frac{n^2}{m} \log^2 n)$.*

Proof: For any two points p, q, we locate the face intersected by the ray from the origin to $q - p$ and then compute the Minkowski distance. The intersection can be reduced to point location in the mapping of the boundary of the zonotope to a sphere centered at the origin. Same technique and bound apply. ∎

Hierarchical method. In the hierarchical method [6, 9], a series of bounding volumes are computed to approximate the object with higher and higher accuracy. The collision detection between two objects is by starting from the coarsest level of the bounding volumes and descending until we separate two bounding volumes or detect the collision between the two objects. Here, we wish to emulate the hierarchical method for implicitly represented zonotopes.

Denote by $H(A, B)$ the Hausdorff distance between two point sets A and B and by $D(A)$ the diameter of a point set A. What is crucial in bounding volumes construction in [9] is a the well-known approximation property of convex bodies: for any convex object A

[2]The Minkowski sum method works for rotations too, but it raises the dimension from three to six and increases the complexity signifi cantly.

in three dimensions and for any $\epsilon > 0$, there exists another convex body B with $O(1/\epsilon)$ vertices so that $H(A, B) \leq \epsilon D(A)$ [7]. There are similar results for zonotopes. In [1], it is shown that a unit ball in $\mathcal{R}^3$ can be approximated within Hausdorff distance ϵ by a zonotope with $O(1/\epsilon)$ generators. The proof is constructive but only works for Euclidean balls. In [2], it is proven that in d-dimensions, any zonoid A can be approximated within $\epsilon D(A)$ by a zonotope with $O(1/\epsilon^{2+\tau} d)$ generators, for any $\tau > 0$. But the proof is non-constructive. In the following, we show that for any zonotope with n generators, we can construct an approximation efficiently.

Lemma 5.5 *For any zonotope A in $\mathcal{R}^3$ and for any $\epsilon > 0$, there exists a zonotope B with $O(1/\epsilon^2)$ generators, so that $H(A, B) \leq \epsilon D(A)$. Further, B can be computed in $O(n)$ time where n is the number of the generators of A.*

Proof: Suppose that $A = (0, \langle v_1, v_2, \cdots, v_n \rangle)$. By symmetry, we can assume that all the v_i's have positive z components. We normalize every v_i and each v_i corresponds to a point p_i on the unit hemisphere. Now, we subdivide the unit hemisphere into k patches so that for any two points p, q in the same patch, the angle between op and oq is bounded by $O(1/k^{1/2})$. Suppose that the patches are $C_1, C_2, \cdots, C_k$. For each C_j, pick a point q_j in C_j and denote by $n_j = \overrightarrow{oq_j}$.

Now divide all the v_i's into clusters according to the patches they are in. Define $V_j = \{v_i | p_i \in C_j\}$. For each vector $v_i \in V_j$, we form two vectors: $v_i^1 = (v_i \cdot n_j) n_j$ is the projection of v_i on the direction n_j, and $v_i^2 = v_i - v_i^1$. Set $u_j = \sum_{v_i \in V_j} v_i^1$, for $1 \leq j \leq k$. Consider the zonotope $Z = (0, \langle v_1^2, v_2^2, \cdots, v_n^2 \rangle)$. Suppose that $(0, \langle w_1, w_2, w_3 \rangle)$ is the tightest axis aligned bounding box of Z. (w_1, w_2, w_3 can be computed easily by projecting each v_i^2 to the x, y, z axes). Now, consider the zonotope $B = (0, \langle u_1, u_2, \cdots, u_k, w_1, w_2, w_3 \rangle)$. We claim that $H(A, B) \leq \epsilon D(A)$ if $k = c/\epsilon^2$ for some constant $c > 0$.

Because $v_i = v_i^1 + v_i^2$, we have that

$$\begin{aligned} A &\subset (0, \langle v_1^1, v_2^1, \cdots, v_n^1 \rangle) \oplus (0, \langle v_1^2, v_2^2, \cdots, v_n^2 \rangle) \\ &\subset (0, \langle u_1, u_2, \cdots, u_k \rangle) \oplus Z \\ &\subset (0, \langle u_1, u_2, \cdots, u_k \rangle) \oplus (0, \langle w_1, w_2, w_3 \rangle) = B. \end{aligned}$$

Denote by $\|v\|$ the Euclidean length of the vector v. For any point $p \in B$ of the form $p = \sum_i \alpha_i u_i + \sum_j \beta_j w_j$, let $q = \sum_i \alpha_i \sum_{v_j \in V_i} v_j \in A$. It suffices to prove that $\|pq\| \leq \epsilon D(A)$ if $k = c/\epsilon^2$ for some constant $c > 0$. First, it is easy to verify that $\|pq\| \leq c_0 \sum_i \|v_i^2\|$ for some $c_0 > 0$. According to the property of the subdivision, we have that $\|v_i^2\| \leq c_1/k^{1/2} \|v_i\|$ for some $c_1 > 0$. Therefore $\|pq\| \leq c_2/k^{1/2} \sum_i \|v_i\|$. Further, it is not hard to see that $D(A) \geq c_3 \sum_i \|v_i\|$ for some constant $c_3 > 0$. Set $c = (\frac{c_2}{c_3})^2$. Then we have that $\|pq\| \leq \epsilon D(A)$, if $k = c/\epsilon^2$. Therefore, $H(A, B) \leq \epsilon D(A)$, and B has $O(1/\epsilon^2)$ generators. ∎

Local walking method. In a local walking method [14, 15], the closest pair of features (vertices, edges, or faces) between two convex objects is tracked for two objects. It is shown in [14] that a simple local walking strategy is guaranteed to find the closest pair of features between two convex volumes. If the motion is small, then the closest pair at any time step should not be "far" from the previous step, and therefore the walking should terminate in a small number of steps. Now, we show that the local walking method can be applied to zonotopes as well. What is crucial in Lin-Canny's method is the ability of discovering the neighboring features of any given feature. This is easy for an explicitly represented polyhedron. However, again we cannot afford to construct the explicit representation of a zonotope. Instead, we show that it is easy to construct the neighborhood of any feature on the fly. For simplicity, consider the walking from face to face in $\mathcal{T}$. On each face of $\mathcal{T}$, there are four choices to choose to which neighbor to exit. We have that

Lemma 5.6 *For any zonotope with n generators, after preprocessing with $O(n)$ space and $O(n \log n)$ time, we can perform the face-to-face walking in $O(\log^2 n)$ time per step.*

Proof: Again, by duality, the walk is to determine the vertices adjacent to the vertex dual to a face in $\mathcal{T}$. This can be done by maintaining a dynamic convex hull data structure. The classical algorithm by Overmars and van Leeuwen [16] gives us the desired bound. ∎

6 Zonotopal Space-Time Volumes

As we mentioned, an important benefit of zonotopes is that their description complexity can be varied or adjusted according to the application needs. In this section we illustrate how this can be exploited for collision detection involving *space-time volumes* [3, 11, 12].

Consider a simple scenario where we have two zonotopes P and Q moving rigidly in $\mathcal{R}^3$. We are interested in verifying that their paths do not collide. In a typical implementation, the dynamics of P and Q are controlled by an integrator. At each time time step the positions of the bodies are updated by the dynamics module, and a new collision test is performed, using (say), the algorithm described in Section 5. Note that in this approach, the rate of collision checking is determined entirely by the system dynamics. It is possible that collision may be missed, if they happen if P and Q overlap, and then stop overlapping, within a single time step. It is also possible that unnecessary collision checks are done, as when the two bodies are far away.

A way to address both of these concern is to do collision checking not on P and Q, but on the portions of space swept by P and Q during a period of time Δt, the so-called space-time volumes of the two bodies. In this section we show how to enclose these space-time volumes in bounding zonotopes. Note that if these bounding volumes are disjoint, then P and Q cannot collide at any time during the interval Δt. If the volumes intersect (either the bounding zonotopes, or the actual space-time volumes swept by the bodies), the P and Q may or may not collide during Δt. Note that P and Q collide if they occupy the same space at the same time. Our space-time volumes are 3-D and are the spatial projections of the real 4-D space-time volumes. Thus, if P collides with Q's location at a different time during Δt, this will lead to a space-time volume intersection, even though P and Q have not collided. When such collisions are detected, the interval Δt can be cut in half and the process repeated, until either a real-collision is detected, or non-intersection is confirmed. We omit further details here.

If the body P just translates during the interval Δt, then its translation vector v can just be added to the list of generators for P to produce a zonotopal description for the exact space-time volume swept by P. The new zonotope needs to be anchored at the origin of P, translated by $v/2$. This simple case illustrates the power of the zonotope description. Of course we must handle the case of a more general rigid motion during Δt. Besides translation, there can be a rotational component as well. Let z be the rotation axis and θ the rotation angle; we assume that $\theta < 180°$—a condition that should be easy to satisfy since in general Δt is small. The rotational component causes each vertex of P to move along a circular arc centered on the z axis, on a plane normal to the axis. Figure 3 below depicts these vertex arcs, projected onto a plane normal to the z-axis and moved to a common origin.

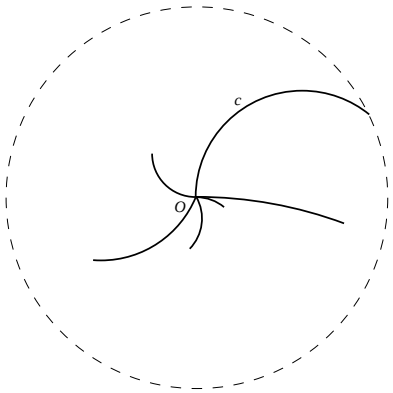

Figure 3: The rotational motion of the vertices of P.

If we can enclose these arcs in a tight-fitting parallelogram (a 2-D zonotope), we augment the generators of P with the translation vector v and these two 'rotational' generators, to produce a space-time bounding volume for the rigid motion of P. The resulting zonotope needs to be centered at the center of P, offset by $v/2$ and the offset between the common origin of the arcs and the parallelogram center in Figure 3. The fact that this zonotope bounds all placements of P during the rotation follows, because the space-volume contain all vertices of each such placement, and therefore (by convexity) all of P throughout Δt.

Since we expect Δt to be small, we also expect the set of arcs we need to enclose to be small in length. However, the number of such arcs can be $\Theta(n^2)$ (where n denotes the number of generators of P, so we wish to avoid looking at all these arcs individually. Looking at Figure 3, we claim that a particular arc c is contained in a circle centered at the origin and passing through its other endpoint, because by assumption each arc spans an angle of less than $180°$. Thus all arcs are fully enclosed in a circle centered at the origin, whose radius is the distance to the the most distant endpoint of any of the arcs. This circle can in turn be enclosed in a square, which forms our bounding parallelogram.

It remains to show how to compute the distance of the vertex of P most distant from the rotation axis z. To do so it suffices to project all generators of P onto a plane normal to z and simply compute the 2-D zonotope generated by them $O(n \log n)$ time, then select the most distant of the $2n$ vertices thus formed. Thus we have shown that:

Theorem 6.1 *Given a rigid motion of P over interval*

Δt, a space-time bounding zonotope for all placements of P can be computed by adding three generators to P. These generators can be computed in $O(n \log n)$ time.

Note that, since our space-time volumes are spatial projections of the true 4-D space time volumes, we need not assume that P moves with constant velocity and angular acceleration during its rigid motion. All that matters is the set of spatial positions occupied and not the times at which they were. It would be interesting to explore the idea of working directly with 4-D zonotopes that are bounding volumes in true space-time, but we have not explored that path since we currently lack an efficient intersection test for 4-D zonotopes.

7 Conclusions

We have proposed the use of zonotopes as bounding volumes for intersection testing and other applications. Our work generates many open questions, including:

- In $\mathcal{R}^3$, how well can a zonotope approximate a given centrally symmetric convex polyhedron?
- How fast can such an optimal (in terms of volume) zonotope be computed?
- What can we say about the number of generators of this zonotope?
- How do we intersect efficiently 4-D space-time zonotopal volumes?
- How can bounding volume hierarchies based on zonotopes be constructed for arbitrary polyhedral geometry?

References

[1] U. Betke and P. McMullen. Estimating the sizes of convex bodies from projections. *Journal of London Mathematics Society*, 27:525–538, 1983.

[2] J. Bourgain, J. Lindenstrauss, and V. Milman. Approximation of zonoids by zonotopes. *Acta Mathematics*, 162:73–141, 1989.

[3] S. Cameron. Collision detection by four-dimensional intersection testing. In *Proc. IEEE Internat. Conf. Robot. Autom.*, pages 291–302, 1990.

[4] J. D. Cohen, M. C. Lin, D. Manocha, and M. K. Ponamgi. I-collide: An interactive and exact collision detection system for large-scale environments. In *Proc. ACM Interactive 3D Graphics Conf.*, pages 189–196, 1995.

[5] R. Cole, J. Salowe, W. Steiger, and E. Szemerédi. An optimal-time algorithm for slope selection. *SIAM J. Comput.*, 18(4):792–810, 1989.

[6] D. P. Dobkin and D. G. Kirkpatrick. Fast detection of polyhedral intersection. *Theoret. Comput. Sci.*, 27(3):241–253, December 1983.

[7] R. M. Dudley. Metric entropy of some classes of sets with differentiable boundaries. *J. Approx. Theory*, 10:227–236, 1974.

[8] H. Edelsbrunner, Leonidas J. Guibas, and J. Stolfi . Optimal point location in a monotone subdivision. *SIAM J. Comput.*, 15(2):317–340, 1986.

[9] J. Erickson, L. J. Guibas, J. Stolfi , and L. Zhang. Separation sensitive collision detection for convex objects. In *Proc. 10th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 327–336, 1999.

[10] S. Gottschalk, M. C. Lin, and D. Manocha. OBB-tree: A hierarchical structure for rapid interference detection. *Comput. Graph.*, 30:171–180, 1996. Proc. SIGGRAPH '96.

[11] Philip M. Hubbard. Space-time bounds for collision detection. Technical Report CS-93-04, Dept. of Computer Science, Brown University, 1993.

[12] Philip M. Hubbard. Collision detection for interactive graphics applications. *IEEE Trans. Visualization and Computer Graphics*, 1(3):218–230, September 1995.

[13] B. Huber and B. Sturmfels. A polyhedral method for solving sparse polynomial systems. *Math. of Computation*, 64:1541–1555, 1995.

[14] M. C. Lin and J. F. Canny. Effi cient algorithms for incremental distance computation. In *Proc. IEEE Internat. Conf. Robot. Autom.*, volume 2, pages 1008–1014, 1991.

[15] B. Mirtich. V-Clip: Fast and robust polyhedral collision detection. Technical Report TR97-05, MERL, 201 Broadway, Cambridge, MA 02139, USA, July 1997.

[16] M. H. Overmars and J. van Leeuwen. Dynamically maintaining confi gurations in the plane. In *Proc. 12th Annu. ACM Sympos. Theory Comput.*, pages 135–145, 1980.

[17] I. J. Palmer and R. L. Grimsdale. Collision detection for animation using sphere-trees. *Comput. Graph. Forum*, 14(2):105–116, June 1995.

[18] M. H. Wright. Interior methods for constrained optimization. In A. Iserles, editor, *Acta Numerica 1992*, pages 341–407. Cambridge University Press, New York, USA, 1992.

[19] G. M. Ziegler. *Lectures on Polytopes*, volume 152 of *Graduate Texts in Mathematics*. Springer-Verlag, Heidelberg, 1994.