A Survey on Techniques for Computing Penetration Depth.

Shashidhara K. Ganjugunte

Abstract. Penetration depth (PD) is a measure that indicates the amount of overlap between two objects. The most popular version is the translational penetration depth, which is the minimum amount of distance by which the two overlapping objects must be translated in order to separate them. A variant of this classical version of the problem is computation of generalized penetration depth (GPD), in which rotational motion is also allowed in addition to translation. In this paper, we survey important results related to computation of PD and GPD for objects in $\mathbb{R}^2$ and $\mathbb{R}^3$.

1 Introduction

For any two objects $\{A, B\} \subseteq \mathbb{R}^n$ the Euclidean distance between A and B is defined as:

$$d(A, B) = \min_{\substack{a \in A \\ b \in B}} ||a - b||.$$

However, if A and B intersect, this measure is 0 and does not indicate the extent of intersection. Penetration depth (also referred to as intersection depth) is a measure that indicates the extent of overlap. In our discussion we shall assume that the object A is movable while the object B is fixed. Given a direction u, the *directional penetration depth* between A and B, $\rho_u(A, B)$ is defined as the minimum amount of translational motion along the direction u, required to make the interior of A disjoint from B. Note, that if the two objects are disjoint then $\rho_u(A, B) = 0$. The (overall) *penetration depth*, $\rho(A, B)$ is defined as minimum amount of translational motion required in any direction required to make the interior of A disjoint from B, i.e.

$$\rho(A, B) = \min_{t \in \mathbb{R}^n} \{||t|| : A \cap \operatorname{int}(B + t) = \emptyset\},$$

where $B + t$ denotes translation of the object B by the vector t.

This notion of penetration depth can be studied by endowing additional degrees of freedom to the object A, e.g. allowing rotation. But doing so increases the complexity of computing the penetration depth. Most of the research till date has been in computing the translational depth, and recently there has been some work in allowing rotational motion to the object A. In this survey, we will depict results related to the computation of penetration depth in dimensions 2 and 3.

Determining intersection depth between two objects is useful in a variety of domains. For example, in interactive graphics it is useful in computing the proximity between the objects. In penalty based simulation methods PD is used to compute the non-penetrating constraint force. More generally, in application domains such as VLSI layout design, robotics, computer-aided-design etc where proximity of objects is important, computation of PD can be very helfpul. In this paper we present survey various techniques used to compute penetration depth between objects in $\mathbb{R}^k$, for $k = 2$ or 3.

This paper is organized as follows: in section 2 we survey results for computing translational penetration depth in $\mathbb{R}^2$, in section 3 we present results related to computing translational penetration depth between polytopes in $\mathbb{R}^3$, in section 4 we describe recent work on generalized penetration depth where rotational motion is also allowed and we conclude section 5 with a few open problems.

2 Penetration depth computation in $\mathbb{R}^2$.

In this section we survey results for computing penetration depth of simple polygons in two dimensions. For two convex polygons A, B, with n_1 and n_2 vertices respectively, Keerthi et. al [13] propose an algorithm that computes the directional penetration depth in $O(\log n_1 \log n_2)$ time and overall penetration depth in $O(n_1+n_2)$ time. We will present the more efficient algorithms developed by Dobkin et. al [7] to compute penetration depth of two polygons which have improved running time for the case of convex polygons. They also develop algorithms to efficiently compute penetration depth when one of polygons is non-convex and the other is convex. Their approach is based on computing the Minkowski sum of the two convex polygons and using it to compute the penetration depth.

The *Minkowski sum* of two sets A and B is defined as:

$$A \oplus B = \{a + b : a \in A, b \in B.\}$$

The following lemmas, by Guibas and Stolfi [11], shows how computing Minkowski sum can help in computing penetration depth.

Lemma 1. *For any two polyhedra A, B and two vectors s, t, we have $(A+s) \cap (B+t) \neq \emptyset$ iff $s - t \in A \oplus -B$.*

Lemma 2. *If A, B are convex polygons, with n_1 and n_2 vertices then $A \oplus B$ is also a convex polygon with $n_1 + n_2$ vertices and can be computed in time $O(n_1 + n_2)$ by merging the edges of A and B in slope order.*

Let A, B be two polyhedra with n_1 and n_2 vertices respectively. Choose a point $a \in A$ as the origin and the reference point for the placement of the polyhedron A. For the polyhedra B let b denote the reference point of B, with respect to the origin above. By the lemma 1, computing the penetration depth in a given direction α is same as computing a minimum t such that $b + t\alpha \notin A \oplus (-B)$. Therefore, finding ρ_α is same as finding the intersection point of a ray with $A \oplus (-B)$. By adopting the techniques of Guibas and Stolfi [11] such an intersection point can be found in $O(\log n)$ time, where $n = \max(n_1, n_2)$. To determine the overall penetration depth, it is required to find the point on $A \oplus -B$ closest to b. This can be done in $O(n)$ time by computing the distance of b to every edge of $A \oplus -B$ and choosing the minimum. The *medial axis* of a polygon is defined as the locus of all points which are equidistant from at least two points from the boundary of the polygon. For a convex polygon the medial axis can be computed in $O(n)$ time. By building a point location data structure for the medial axis diagram, one can compute in $O(\log n)$ time the penetration depth $\rho(A, B+t)$ after $O(n)$ time preprocessing.

When A is allowed to be nonconvex, $A \oplus -B$ can have $\Omega(n_1 n_2)$ vertices. In order to extend these results to the case where A is a nonconvex polygon and B is a convex polygon, Dobkin et. al propose the following approach:

1. Compute a triangulation of A, let $A = \bigcup_{i=1}^{k} T_i$, where T_i is a triangle.
2. Find an "implicit" representation of $R_i = T_i \oplus -B$.
3. Find an "implicit" representation for the boundary of $R = \bigcup_{i=1}^{k} R_i$.

The Minkowski sum of triangles T_i with $-B$ require storing the 3 edges of the triangles T_i in slope order among the edges of B. This is done implicitly by storing an index in the appropriate position of the edge array used to store B. The union of the convolved rectangles R consists of convex arcs and line segments. In the implicit representation, only the edges are explicitly stored. We denote this explicitly stored boundary of R by Λ. Once the boundary of R has been computed, the directional PD can be computed by intersecting a ray from b in the given direction u

with each of the $O(n_1)$ bounding edges and arcs of R and computing the point on the boundary closest to b. This can be done in $O(n_1 \log n_2)$ time as computing the intersection of a ray with a line takes $O(1)$ time and computing the intersection of a ray with a convex arc of n_2 vertices takes $O(\log n_2)$ time. To compute the overall penetration depth, the following steps are executed:

1. Compute $r = \min_{p \in A} ||b - p||$.
2. Compute $V(b, r) = V(b) \cap D(b, r)$, where $V(b)$ is the visibility polygon of b and $D(b, r)$ is the disck of radius r centered at q.
3. Compute $\rho(A, B + b) = \min_{p \in V(b,r)} ||b - p||$.

Steps, 1, 2 take $O(n_1 \log(n_1 + n_2) + n_2)$ time while step 3 takes $O(n_1 + n_2)$ time, so the above procedure takes $O(n_1 \log(n_1 + n_2) + n_2)$ time.

Computing the penetration depth between two non-convex polygons is substantially harder as the Minkowski sum can have complexity $\Omega(n^4)$.

3 Penetration depth computation in $\mathbb{R}^3$

In this section we present some of the key algorithms developed to compute penetration depth between two polytopes in three dimensions.

3.1 Penetration depth using Minkowski sums

Cameron and Culley propose an algorithm [5] that uses the Minkowski sum of of the two polytopes A, B as input and finds the point on the boundary of $M = A \oplus -B$ closest to the origin O. The input to their algorithm is a list H of half-spaces representing M, such that for each $h \in H$, there is a facet of M that lies in h.

1. First find the point p_1, which is the orthogonal projection of the origin onto the hyperplane h_f representing the farthest halfspace from O in h.
2. If $p_1 \notin M$, find a list L_f of halfspaces that intersect h_f. The intersection of the halfspaces is a polygon P_f.
3. If the furthest linear half-space from p_1 does not contain any edge of P_f, it is removed from L_f, this is continued until the furthest linear half-space has an edge of P_f.
4. The algorithm then recursively finds the point p_2 closest to p_1 on P_f.
5. If p_2 lies in the interior of an edge then it will be the point closest to the origin

6. Otherwise, exhaustively search for the point closer to the origin by first finding all the half spaces in $H \setminus h_f$ that pass through p_2 and considering all edges that intersect the sphere with origin as center and radius $||p_2||$ as candidates.

Their algorithm needs the Minkowski sum as input, and also does not have theoretical analysis.

3.2 Directional penetration depth using Dobkin-Kirkpatrick hierarchy

For the case of convex polytopes, Dobkin et. al [7] propose an algorithm which after linear time preprocessing, computes directional penetration depth in $O(\log n_1 \log n_2)$ time for any direction. In the preprocessing step they build two hierarchies of polytopes, namely the *inner* and *outer* hierarchies. These are popularly known as *Dobin and Kirkpatrick* hierarchies [8–10]. In the discussion below, we closely follow the notation of [7]. Let $V(P)$ denote the vertex set of a polytope P, and let $H(P)$ denote the set of planes containing the facets of P. A polytope sequence $P_1, P_2, \ldots, P_k$ is an inner (resp. outer) hierarchy of a polytope P, if $\forall 1 \leq i < k$:

1. $P_1 = P$, and P_k is a simplex.
2. $P_{i+1} \subset P_i$ and $V(P_{i+1}) \subset V(P_i)$ (resp. for outer hierarchy, $P_{i+1} \supset P_i$ and $H(P_{i+1}) \subset H(P_i)$).
3. $V(P_i) \setminus V(P_{i+1})$ is an independent set in P_i (resp. for outer hierarchy, $H(P_i) \setminus H(P_{i+1})$ bounds an independent set of facets in P_i).

Dobkin et. al [9] prove that for every convex polytope P, such hierarchies of height $O(\log |P|)$ can be constructed in linear time. One advantage of building such a hierarchy is that we can find the first intersection of a directed line or that of a line or a plane translating from infinity in the direction u, with P in $O(\log |P|)$ time. So, given a polytope A with constant complexity $\rho_u(A, P)$ can be computed in $O(\log n)$ time. In order to compute the $\rho_u(A, B)$ for convex polytopes A, B, the algorithm devised by Dobkin et. al, alternates between two phases one of which tries to find an intersecting pair A_i, B_i of the corresponding hierarchies and the other that computes $\rho_u(A_{i-1}, B_{i-1})$. The algorithm terminates once $\rho_u(A_1, B_1)$ is determined. Since the depth of the hierarchy is $O(\log n)$, the overall running time is $O(\log n_1 \log n_2) = O(\log^2 n)$.

3.3 Deep Dual space: An incremental algorithm to compute PD

We now describe an incremental algorithm due to Kim et. al [14]. For a convex polytope P the *width* $W(P)$ is defined as the minimum distance between two parallel planes supporting P. The penetration depth is related to width, as for example, $\rho(A, A) = W(A)$. Let $\mathcal{Z}_{AB}$ denote the set of pairs of planes (Z_1, Z_2) such that Z_1 supports the polytope A and Z_2 supports polytope B and the outer facet normals are in opposite directions for Z_1 and Z_2. The following equation relates width and PD:

$$\rho(A, B) = \min_{(Z_1,Z_2)\in\mathcal{Z}_{AB}} d(Z_1, Z_2),$$

where $d(Z_1, Z_2)$ is the distance between planes Z_1, Z_2.

The following lemma by Houle and Toussaint [12] which was used in their width computation algorithm is useful in restricting the candidate set of support planes to be considered to compute $\rho(A, B)$.

Lemma 3. *The width of a point set $P \subseteq \mathbb{R}^3$ is the minimum distance between parallel supporting planes and the planes that realize this minimum distance pass through either an antipodal vertex-facet (VF) pair or an antipodal edge-edge(EE) pair of the convex hull P.*

In order to find the supporting planes passing through VF or EE pairs, Houle and Toussaint use a duality transform to map the vertices, edges and facets to circular convex regions, arcs of the great circle and points on the normal (Gaussian) diagram of the polytopes. The antipodal pairs can be found by overlapping the upper and the lower hemispheres and finding the intersections. Kim et. al design an incremental algorithm that adapts this methodology but constructs the Minkowski sums (or normal diagrams in dual space) incrementally as required. Their algorithm is outlined below:

1. *Initalization:* Choose a vertex from each of the two polytopes, such that there is a plane supporting both polytopes and passes through these two vertices. This pair is referred to as the vertex hub pair. Typically, these are chosen two be extremal vertices in the direction of the vector δc and $-\delta c$, where δc is the vector difference in centroids of the two polytopes.
2. At each step expand the surface of Minkowski sums by choosing another vertex hub pair that is closest from the origin of Minkowksi sums.

3. Update the current PD value and iterate until convergence.

The above algorithm converges to a local minima, and may not find exact PD, and in the worst case can take upto $O(n^2)$ time.

3.4 A fast randomized algorithm to compute PD

Every facet of the Minkowski sum $M = A \oplus -B$ is produced from the Minkowksi sum of an facet, edge, vertex of A with a vertex, edge, facet of B respectively. The facets produced by the Minkowski sum of either a facet or a vertex of B can be computed in $O(n_1 + n_2 \log(n_1 + n_2))$ time [6]. So, in order to find the PD efficiently, it suffices to find the shortest distance from the origin to the facets of M generated by an edge of A and an edge of B. Although there are potentially $\Omega(n_1 n_2)$ of such pairs, not all of them can generate a facet of M. Based, on this observation Agarwal et. al [1, 2] propose an algorithm that first creates a family of pairs of subsets of the edges and uses them to reduce the number of edge pairs to be considered. Their algorithm is outlined below:

The family of edge pairs is denoted (as in [1]) $\mathcal{F} = \{(A_1, B_1), \ldots, (A_k, B_k)\}$ where A_i are subset of edges of A and B_i are subset of edges of B are such that every pair of edges $(a_i, b_i) \in (A_i, B_i)$, $1 \leq i \leq n$, generate a facet of M and for every facet of M generated by a pair of edges say (v_a, v_b) there is a pair of subsets $(A_v, B_v) \supset (v_a, v_b)$ with $1 \leq v \leq n$. Also, For all (A_i, B_i), lines supporting the edges in A_i and B_i are vertically separated. Additionally, $\mathcal{F}$ can be paritioned suitably, for details of the partition properties we refer the reader to the original paper. Once such a decomposition is computed, their algorithm proceeds as follows:

1. Find Δ^* the minimum of distances from the origin to the facets of M generated by vertices or faces.
2. For each pair $(A_i, B_i) \in \mathcal{F}$ choose $(a, b) \in (A_i, B_i)$ and compute the distance Δ_i from the origin to the affine hull of $(a \oplus -b)$.
3. Return minimum of $\{\Delta^*\} \cup \{\Delta_i\}_{i=1}^{n}$.

Using the properties of the family $\mathcal{F}$ mentioned above, Agarwal et. al show that the above algorithm runs in $O(n_1^{3/4+\epsilon} n_2^{3/4+\epsilon} + n_1^{1+\epsilon} + n_2^{1+\epsilon})$ randomized expected time, for any $\epsilon > 0$.

They also propose approximation algorithms that given a parameter $\delta > 0$ compute the penetration depth in time $O(n_1 + n_2 + (\log^2(n_1 + n_2))/\delta)$, within a factor of $1 + \delta$. If the penetration is shallow, with K_δ being the number of facets of M within a distance of $1 + \delta(\rho(A, B))$, they also propose an approximation algorithm that computes the penetration depth in time $O(n_1 + n_2 + K_\delta \log(n_1 + n_2) + (\log^2(n_1 + n_2))/\delta)$ with an approximation factor of $1 + \delta$.

3.5 Miscellaneous

Bergen et. al [4] propose an algorithm for estimating a lower bound on penetration depth between convex polytopes by expanding a polyhedral approximation of the Minkowski sum. Kim et. al [15] propose an algorithm to compute an upper bound on PD for general polyhedral models by decomposing the polyhedra into convex components, and performing a closest point query using rasterization hardware. The worst case complexity of their algorithm can be $O(n^4)$.

4 Generalized Penetration Depth

In this section we describe the salient aspects of recent work on penetration depth by Zhang et. al [18], wherein rotational motion is also allowed along with translation. More precisely, let the two polytopes $\{A, B\} \subseteq \mathbb{R}^3$, with B being fixed and A allowed to translate and rotate. The configuration space (C-space) $\mathcal{C}$ can be described by six parameters $(x, y, z, \phi, \theta, \psi)$, where x, y, z are usual cartesian co-ordinates denoting translational motion and ϕ, θ, ψ are the Eulerian angles denoting rotational motion. Given a configuration u, denote the placement of A at u by $A(u)$, and let $p(u)$ denote the position of a point $p \in A$ at u. Given two configurations u, v let γ be a curve that connects them. Let γ be a curve traced by a point $a \in A$ when the configuration changes from u to v and let $L(a, \gamma)$ be the it's arc length. Let $\varGamma$ be the set of all curves connecting u, v. Zhang et. al define a distance metric in the C-space as follows:

$$D_g(u, v) = \min_{\gamma \in \varGamma} \max_{a \in A} L(a, \gamma).$$

Let $q = u - v$, an upper bound for the above metric is given by:

$$D_g(u, v) \leq \pi_x(q) + \pi_y(q) + \pi_z(q) + R_\phi \pi_\phi(q) + R_\theta \pi_\theta(q) + R_\psi \psi(q), \quad (1)$$

where π is the projection operator and R_ϕ, R_θ, R_ψ are maximum distance from any point on A to the X, Y, Z axes respectively.

Using this metric, the generalized penetration depth ρ^g is defined as follows:

$$\rho^g(A, B) = \min_{v \in \mathcal{C}} \{D_g(u, v) : \text{int}(A(v) \cap B) = \emptyset\}.$$

For convex polyhedra in $\mathbb{R}^3$, they prove that $\rho(A,B) = \rho^g(A,B)$, where $\rho(A,B)$ is the penetration depth with only translation allowed. They also present an algorithm to compute the generalized penetration depth in the case where A the movable object is convex and the complement of B, namely $\bar{B}$ is also convex, which is outlined below:

1. Find a configuration u such that $A(u) \subseteq \bar{B}$.
2. Using the upper bound in equation 1 as an objective function iteratively compute a configuration v that yields a tighter bound.

In order to compute the containment they adapt algorithms described by Milenkovic et. al [16] and Grinde et. al [3] which compute a locally optimal containment by reducing the problem to an instance of linear programming. Once the containment configuration is computed, ρ^g can be found using equation 1.

For the general case of two non-convex objects, they provide an algorithm that computes a lower bound by finding the inner convex convers, which are subsets of the polyhedra whose union covers the polyhedra. The algorithm then computes the minimum pairwise penetration depth between these covers as a lower bound. The running time for computing the lower bound is $O(n_a n_b)$ where n_a, n_b are the total number of features (vertices, edges and facets) of the convex covers of A and B respectively. They also provide an algorithm that computes an upper bound which is based on the notion of a piecewise linear separator. Given disjoint non-convex objects A, B it is known (see Mount [17]) that either a single plane separates them or a set of linear surfaces separates them. Their algorithm enumerates all possible separating planes and convex separtors by evaluating the convexity of boundary of B. For each separator, using the technique to compute the PD between a convex object and convex complement described above they compute an upper bound of the penetration depth. The minimum over all the separators gives an overall upper bound. For details we refer the reader to the original paper.

5 Conclusion

Penetration depth measures the extent of overlap between two intersecting objects. In this paper, we presented a survey of results related to computing the penetration depth between two objects in $\mathbb{R}^2$ and $\mathbb{R}^3$. Desigining efficient algorithms for computing penetration depth for non-convex objects seems to be hard. As such, there are several (to the best of our knowledge) open problems such as:

1. Can directional penetration depth between two non-convex polygons in $\mathbb{R}^2$ be computed in sub-quadratic time?
2. Can the translational PD between non-convex polygons (respectively polytopes) be computed in sub-quartic time (respectively in less than $O(n^6)$ time) with perhaps a constant factor of approximation?
3. Can efficient approximation algorithms with tight bounds be devised for computing *generalized* penetration depth between two non-convex polyhedra in $\mathbb{R}^3$?

References

1. P. K. Agarwal, L. J. Guibas, S. Har-Peled, A. Rabinovitch, and M. Sharir. Computing the penetration depth of two convex polytopes in 3d. In *SWAT '00: Proceedings of the 7th Scandinavian Workshop on Algorithm Theory*, pages 328–338, London, UK, 2000. Springer-Verlag.
2. P. K. Agarwal, L. J. Guibas, S. Har-Peled, A. Rabinovitch, and M. Sharir. Penetration depth of two convex polytopes in 3d. *Nordic Journal of Computing*, 7(3):227–240, 2000.
3. G. R. B and C. T.M. Containment of a single polygon using mathematical programming. *European Journal of Operational Research*, 92(2):368–386(19), 1996.
4. V. D. Bergen. Proximity queries and penetration depth computation on 3d game objects. In *Game Developers Conference*, 2001.
5. S. A. Cameron and R. K. Culley. Determining the minimum translational distance between two convex polyhedra. pages 591–596, 1986.
6. B. Chazelle, H. Edelsbrunner, L. J. Guibas, and M. Sharir. Algorithms for bichromatic line-segment problems polyhedral terrains. *Algorithmica*, 11(2):116–132, 1994.
7. D. P. Dobkin, J. Hershberger, D. G. Kirkpatrick, and S. Suri. Computing the intersection-depth of polyhedra. *Algorithmica*, 9(6):518–533, 1993.
8. D. P. Dobkin and D. G. Kirkpatrick. Fast detection of polyhedral intersections. In *ICALP '82: Proceedings of the 1982 International Colloquium on Automata, Languages and Programming*, pages 154–165, 1982.
9. D. P. Dobkin and D. G. Kirkpatrick. A linear algorithm for determining the separation of convex polyhedra. *Journal of Algorithms*, 6(3):381–392, 1985.
10. D. P. Dobkin and D. G. Kirkpatrick. Determining the separation of preprocessed polyhedra: a unified approach. In *Proceedings of the seventeenth International Colloquium on Automata, languages and programming*, pages 400–413, New York, NY, USA, 1990. Springer-Verlag New York, Inc.
11. L. J. Guibas and J. Stolfi. Ruler, compass, and computer : the design and analysis of geometric algorithms. Technical Report SRC-RR-37, Hewlett Packard Laboratories, Feb. 14 1989.
12. M. E. Houle and G. T. Toussaint. Computing the width of a set. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 10(5):761–765, 1988.
13. S. S. Keerthi and K. Sridharan. Efficient algorithms for computing two measures of depth of collison between convex polygons. Technical report, Department of Computer science and Automation, Indian Institute of Science, Bangalore, 1989.

14. Y. J. Kim, M. C. Lin, and D. Manocha. Incremental penetration depth estimation between convex polytopes using dual-space expansion. *IEEE Transactions on Visualization and Computer Graphics*, 10(2):152–163, 2004.
15. Y. J. Kim, M. A. Otaduy, M. C. Lin, and D. Manocha. Fast penetration depth estimation using rasterization hardware and hierarchical refinement. In *SCG '03: Proceedings of the nineteenth annual symposium on Computational geometry*, pages 386–387, New York, NY, USA, 2003. ACM Press.
16. V. J. Milenkovic. Rotational polygon overlap minimization. In *SCG '97: Proceedings of the thirteenth annual symposium on Computational geometry*, pages 334–343, New York, NY, USA, 1997. ACM Press.
17. D. M. Mount. Intersection detection and separators for simple polygons. In *SCG '92: Proceedings of the eighth annual symposium on Computational geometry*, pages 303–311, New York, NY, USA, 1992. ACM Press.
18. L. Zhang, Y. J. Kim, G. Varadhan, and D. Manocha. Generalized penetration depth computation. In *SPM '06: Proceedings of the 2006 ACM symposium on Solid and physical modeling*, pages 173–184, New York, NY, USA, 2006. ACM Press.