Fast Penetration Depth Computation

Department of Computer Science University of North Carolina at Chapel Hill February 2004

Background

Let *P* and *Q* be two intersecting polyhedra. The penetration depth (PD) of *P* and *Q*, *PD(P,Q)*, is the minimum translational distance that one of the polyhedra must undergo to render them disjoint. PD is often used in contact resolution for dynamic simulation, force computation (e.g. in haptic rendering), tolerance verification for virtual prototyping, motion planning of autonomous agents, etc. We present two classes of PD algorithms, respectively for convex polytopes and for general polyhedral models.

A general framework to compute the PD is based on Minkowski sums. It is well known that one can reduce the problem of computing the PD between *P* and *Q* to a minimum distance query on the surface of their Minkowski sum (difference). More specifically, the *PD(P,Q)* is defined as a minimum distance from the difference vector between the origins of *P* and *Q*, O_{Q-P}, to the surface of the Minkowski sum of *P* and *-Q*. It is also known that the computational complexity of computing the Minkowski sums can be as high as $O(n^2)$ for convex polytopes and $O(n^6)$ for non-convex polyhedra, where n is the number of features. Therefore, instead of explicitly computing the Minkowski sums, our algorithm utilizes a local walking technique for convex polytopes, and a hybrid approach (combining image-space and object-space methods) for non-convex models to compute the PD.

DEEP: PD Algorithm for Convex Objects

DEEP is an incremental algorithm to compute PD for convex polytopes in 3D. The algorithm incrementally marches towards a "locally optimal" solution by walking on the surface of the Minkowski sum. The surface of the Minkowski sums is computed *implicitly* by constructing a local Gauss map. DEEP performs incremental computations and exploits spatial and temporal coherence between successive frames. In practice, the running time is observed to be "almost constant"—a fraction of a millisecond on a 1 GHz PC, when there is high motion coherence present in the environment. Furthermore, DEEP is able to compute the optimal PD between the underlying polytopes in most cases.

PD Algorithm for Non-Convex Objects

Our algorithm to compute the PD for non-convex objects is based on the decomposition property of Minkowski sums, i.e. if $P = P_1 \cup P_2$, then $P\text{-}Q = (P_1\text{-}Q) \cup (P_2\text{-}Q)$. We use a surface-based convex decomposition of the boundary of non-convex objects and utilize the graphics rasterization hardware to estimate the PD. We do not explicitly compute the boundary of the union or any approximation to it. Rather we perform the closest point query using a multipass algorithm that computes the closest point from the origin to the boundary of pairwise Minkowski sums. The resulting maximum depth fragment at each pixel computes an approximation to the PD, up to the image-space resolution used for this computation. Given this estimate, we further refine it using an object-space incremental algorithm that performs a local walk on the Minkowski sum.

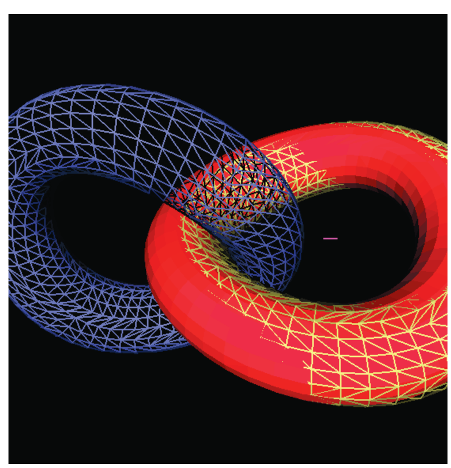
Highlights

- DEEP: An incremental algorithm based on Dual-space Expansion for Estimating Penetration depth between convex polytopes in 3D.
- DEEP utilizes motion coherence present in the environment and exhibits constant running time in practice.
- Our general PD algorithm for non-convex polyhedra uses a combination of object-space and image-space techniques.
- The general algorithm computes pairwise Minkowski sums of decomposed convex pieces, performs a closest point query using rasterization hardware and refines the estimated penetration depth by object-space walking.

We improve the performance of the algorithm using a number of acceleration techniques. These include hierarchical representations based on convex bounding volumes, use of model simplification algorithms, object-space culling approaches, and image-space acceleration techniques applied to the multipass algorithm.

The resulting algorithm includes a pre-computation phase as well as a runtime query. The pre-computation phase consists of the following steps:

1. Decompose the boundary of each polyhedron into convex patches using a greedy approach.
2. Compute a bounding volume hierarchical representation of the model. Each node in the tree corresponds to a convex polytope and each leaf is a convex hull of a decomposed convex patch.
3. Use model simplification algorithms to compute a lower polygon count approximation of the interior nodes.



Our penetration depth (PD) algorithm computes the minimum translational distance to separate two intersecting rigid polyhedral models. In this figure, a yellow wire-framed torus intersects with a blue wire-framed torus. Based on the PD result, the yellow torus is translated to the red solid-colored torus that does not intersect with the blue torus any longer.

Given two polyhedra and their bounding volume representations, the runtime phase of the algorithm proceeds in the following manner:

1. Compute an upper estimate to the amount of PD. Let that estimate be d_{est}. Initially we compute an estimate based on the root nodes of each tree.
2. At each level of the two hierarchies, repeat the following steps:
   a. Consider all pairwise combinations of nodes at the current level. Cull away all the pairs that are non-overlapping and are more than d_{est} apart.
   b. Compute pairwise Minkowski sums of the rest of the node pairs that have not been culled away.
   c. Perform the closest point query using the rasterization hardware to compute a PD estimate.
   d. Extract the penetration features in the object-space. Perform a local walk and refine the PD estimate using incremental algorithms.

Project Members

Dinesh Manocha, professor *(dm@cs.unc.edu)*
Ming C. Lin, professor *(lin@cs.unc.edu)*
Young J. Kim, postdoctoral researcher *(youngkim@cs.unc.edu)*
Miguel A. Otaduy, graduate research assistant *(otaduy@cs.unc.edu)*

Research Sponsors

U.S. Army Research Office, Department Of Energy ASCI Program, National Science Foundation, Office of Naval Research, Intel Corporation, Government of the Basque Country

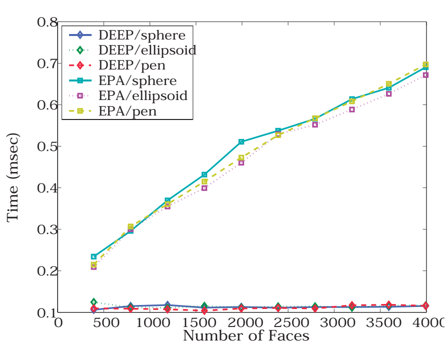
Selected Publications

Kim, Y. J., M. C. Lin, and D. Manocha. "DEEP: Dual-Space Expansion for Estimating Penetration Depth Between Convex Polytopes," *Proc. ICRA* 2002.

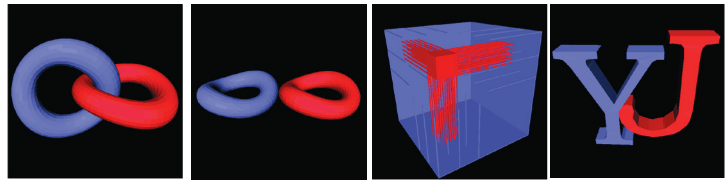
Kim, Y. J., M. A. Otaduy, M. C. Lin, and D. Manocha. "Six-Degree-of-Freedom Haptic Display Using Localized Contact Computations," *Proc. Haptic Symposium* 2002.

Kim, Y. J., M. A. Otaduy, M. C. Lin, and D. Manocha. "Fast Penetration Depth Computation Using Rasterization Hardware and Hierarchical Refinement," *Department of Computer Science technical report*, University of North Carolina, 2002.

Benchmark

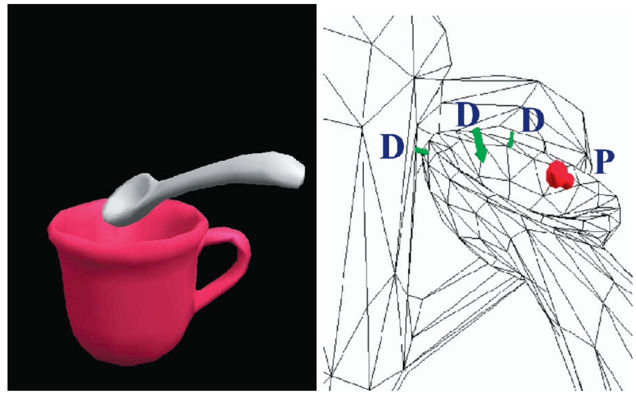


PD Benchmarking for Convex Models (DEEP). The figure shows the performance of DEEP for a sphere, an ellipsoid and a pen model with a fixed PD amount and variable polygon count. We also compare the performance of DEEP with another recent PD algorithm, the Expanding Polytope Algorithm (EPA).

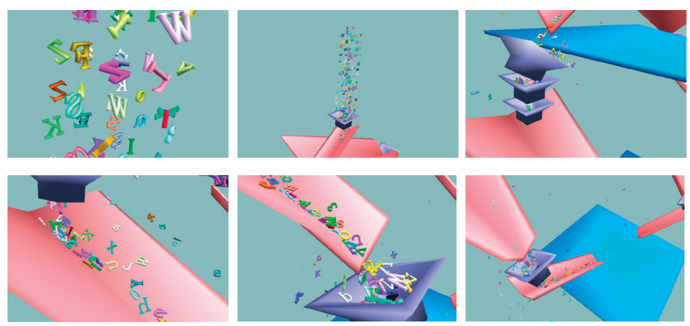


PD Benchmarking for Non-Convex Models. From left to right: interlocking tori, touching tori, interlocked grates, and touching alphabets. The running time varies from a fraction of a second to a few seconds, based on the model complexity and the relative configuration of the two polyhedra.

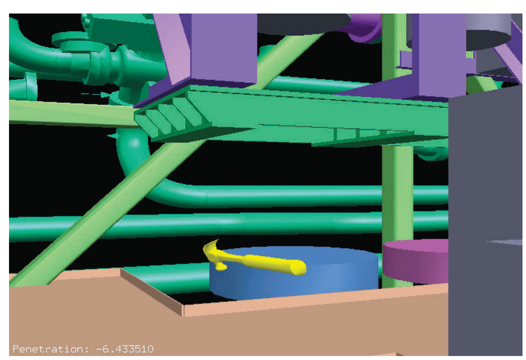
Applications



Six Degree-of-Freedom (DOF) Haptic Rendering. We use DEEP compute contact forces for localized contact areas between non-convex models. The green and red arrows in the figure on the right indicate contact forces.



Rigid-Body Dynamic Simulation. Our non-convex PD algorithm is used to perform "smarter time stepping" in a dynamic simulation.



Tolerance Verification. Our non-convex PD algorithm is used to check for positive and negative distances between a tool (yellow hammer) and the nearby structures along a planned maintenance path in a virtual machine room.

For More Information

- gamma.cs.unc.edu/DEEP
- gamma.cs.unc.edu/6DOFLCC
- gamma.cs.unc.edu/PD