Benchmarks for Implementations of Set-Operations Algorithms on Polyhedral Solids

Michael Karasick

IBM Research Division, T.J. Watson Research Center, P.O. 218, Yorktown Heights, NY 10598

Abstract

Efficient and correct implementations (of algorithms) for set operations are important components of a useful solid modeling system. A measure of the efficiency of such an algorithm can be obtained from an analysis of the running time of the algorithm or by actually timing the implementation on known "benchmarks." Since correct finite-precision implementations are very difficult to obtain, we resort to benchmarking to test the numerical "robustness" of the implementation.

This paper describes a collection of benchmarks designed to test the robustness of finite-precision implementations of intersection algorithms for polyhedral solids. Most of the benchmarks are of the form: take a (representation of) a solid, rotate it slightly, and then intersect the perturbed solid with the original solid, and construct a representation of the result. If the rotation is very small then a finite-precision implementation of an intersection algorithm is unable to differentiate between the two. If the rotation is large, then the result is distinct from both of the input solids. Usually, there is a narrow range of rotations where implementations fail unpleasantly: They generate a representation of an impossible solid; they enter an infinite loop and do not terminate; or they simply stop. The benchmarks use a bisection technique to find this "region of instability." The magnitude of this region gives a measure of the numerical robustness of an implementation; and by understanding why an implementation does poorly on a particular benchmark, we can gain insight into how one might obtain a more numerically robust implementation.

These benchmarks were actually used to develop an implementation of an intersection algorithm. Surprisingly, for some of the benchmarks, the implementation has no region of instability. Other benchmarks caused the intersection algorithm to fail, and the reasons for these failures illustrate the difficulties in obtaining robust finite-precision implementations of geometric algorithms.