Critique for CS 448B: Topics in Modeling
The Voronoi-clip Collision Detection Algorithm

Richard Bragg
March 3, 2000

1. Citation

Mirtich B. (1997) *V-Clip: Fast and Robust Polyhedral Collision Detection*. Mitsubishi Electric Research Laboratory, Technical Report TR-97-05.

2. Synopsis

This paper presents the Voronoi-clip (V-clip) algorithm for polyhedral collision detection. The method is similar to that of the Lin-Canny algorithm [1] in that it tracks the closest pair of features between two polyhedra. However, the V-clip method offers superior robustness, requires less complex code, can better handle penetrating polyhedra, and is more useful for collision detection between non-convex polyhedra than the Lin-Canny method. The paper first gives an introduction to the most common algorithms used for collision detection. It then presents the theoretical background for the V-clip method, followed by the algorithm itself, presented in pseudocode. Finally, the V-clip algorithm is compared to two other commonly used collision detection algorithms in several different performance tests. The V-clip algorithm performs remarkably well in comparison to the other methods.

3. Summary

3.1 Problem Statement

Input: Two convex polyhedra, described by the features (vertices, edges, and faces) that form their boundaries.

Output: Identification of the two closest features between the two input polyhedra, a measure of the closest distance between those features, and an indication of penetration, if any.

3.2 Introduction

According to this paper, there are two main categories of collision detection algorithms: feature-based algorithms and simplex-based algorithms. Feature-based algorithms are so named because these algorithms perform geometric computations on the features (vertices, edges, and faces) or two polyhedra to determine if they are disjoint. The Lin-Canny algorithm [1] is the most well-known and commonly used of these types of algorithms, but it has two main drawbacks: it does not handle penetrating polyhedra, and it can report convergence at local minima that are not actually the closest pair of features globally.

On the other hand, simplex-based algorithms consider each polyhedron to be a convex hull of point sets, and then perform operations on simplices defined by subsets of those points. This type of algorithm was first introduced by Gilbert, Johnson, and Keerthi [2], and those that followed have basically been enhancements and variations of their method, which has come to be known as the GJK algorithm.

The V-clip method is a feature-based algorithm that attempts to address the shortcomings of the Lin-Canny method. By being able to handle the penetration condition, it can also be adapted to non-convex polyhedra by creating hierarchies of convex components. Each time the outermost convex object is penetrated, the hierarchy is broken down to the next level. For very complex

objects and systems, other methods are suggested to be better, such as the OBB-tree algorithm presented by Gottschalk, *et al.* [3].

3.3 Voronoi Regions and Planes

In order to understand the V-Clip algorithm, one must first understand a few basic concepts and terms. First and foremost are *Voronoi regions* and *Voronoi planes*. For a given feature X on a convex polyhedron, a Voronoi *region* is the set of points outside the polyhedron that are as close to X as to any other feature on the polyhedron. The bounding plane between two adjacent Voronoi regions is called the Voronoi *plane* of the two features that define those adjacent Voronoi regions. These planes are either vertex-edge or face-edge planes, depending on the features that they share.

The V-clip algorithm is then based on the basic theorem that if X and Y are a pair of features on separate disjoint polyhedra, then if X is in the Voronoi region of Y and vice versa, then X and Y are the closest features between the polyhedra, and the closest points between those features are the closest points between the polyhedra. A proof of this theorem was provided in the appendix to the paper.

3.4 Voronoi Clipping

The basic problem that repeatedly arises in the execution of the V-clip algorithm involves the following steps:

1. Select a pair of features X and Y from each polyhedron.
2. Determine if the closest point on Y to X is in the Voronoi region of X.
3. If not, update X to a neighboring feature that is either of a higher dimension than X, in which case it will decrease the inter-feature distance, or of a lower dimension than X, in which case it will keep the inter-feature distance constant. (It is for this reason that the algorithm should converge, as the inter-feature distance never increases.)

If Y is a vertex, steps 2 and 3 are relatively straightforward. However, if Y is an edge, often only a segment of that edge is within the Voronoi region of X and the rest of the edge is outside the region. This occurs when the edge passes through one of the Voronoi planes that define the region. The procedure for dealing with this situation is called *Voronoi clipping.*

In a nutshell, the clipping procedure clips any portions of the edge that penetrate the Voronoi planes of the Voronoi region. If the edge penetrates one or more of those planes, then a derivative test is performed at each intersection to determine whether or not to update X, and if so, which feature it should be updated to. On the other hand, if the edge lies completely outside the Voronoi region of X (*simple exclusion*) or is clipped, but has no unclipped segments in the Voronoi region of X (*compound exclusion*), then the ensuing update procedure depends on the feature type of X. Each feature type X has a slightly different protocol for selecting the updated feature when the edge Y is excluded from its Voronoi region.

3.5 Handling Penetration

The way that features are updated with the V-clip algorithm allows penetration between polyhedra to be handled efficiently. If a feature of one polyhedron is in the interior space of another polyhedron, a series of updates will eventually lead to the condition that an edge of one polyhedron will have its endpoints on opposite sides of a face of the other polyhedron. At this point, penetration is signaled.

3.6 Escaping Local Minima

An algorithm is also presented to avoid the problem of local minima, which occurs specifically when the closest features between two polyhedra are a vertex and a face, the vertex lies below the plane, and the vertex's neighboring edges are oriented away from the plane. If this case arises, the algorithm works by first testing whether or not the vertex is actually penetrating

the other polyhedron, and if not, updating the face to the face that is the maximum signed distance from the original face.

3.7 The V-Clip Algorithm

The entire V-clip algorithm is presented in pseudocode. It basically consists of an algorithm for each possible *state* (vertex-vertex, vertex-edge, vertex-face, edge-edge, edge-face) of the system. Each algorithm provides rules for the next update and how to detect convergence for that state, with the exception of the face-edge algorithm, which never returns convergence, but is the only state that can detect penetration.

3.8 Experimental Results

A series of tests were performed on four different pairs of polyhedra of increasing complexity to compare the performance and robustness of the V-clip algorithm with the Lin-Canny algorithm and an enhanced version of the GJK algorithm. The polyhedra used were 1) cubes, 2) regular icosahedrons, 3) polygonized disks with high complexity (60-sided polygons) faces, and 4) tessellated spheres with high overall complexity (642 vertices, 1920 edges, and 1280 triangular faces). A count of floating-point operations was used to compare the computational cost of each algorithm. Accuracy was also compared, as was the performance of each test in degenerate situations.

With the first three types of polyhedra (cubes, icosahedrons, disks), the V-Clip algorithm had the fewest floating-point operations of the three algorithms. With the spheres, the V-Clip algorithm also performed the best at high levels of coherence, but was surpassed by the enhanced GJK algorithm at lower levels of coherence. This is likely due to the fact that the enhanced GJK is a simplex-based algorithm rather than a feature-based one. As such, the GJK algorithm can "tunnel" through the polyhedron to achieve convergence, while the V-Clip algorithm must traverse the boundary surface.

The accuracy tests identified the V-Clip as far superior to the other two methods, never returning a distance value more than a predefined tolerance of 10^{-6} greater than the minimum distance returned by any of the three algorithms. The results of the degeneracy tests revealed that both the V-Clip and the GJK algorithms were flawless, although this result for the GJK algorithm required the proper tuning of its main tolerance. The V-Clip algorithm has no such tolerances to tune.

4. Comments

4.1 Contribution

Clearly the contribution of this paper to the field of collision detection is significant. The V-clip algorithm provides results that are superior in many ways to both the Lin-Canny and GJK algorithms. It is efficient in terms of floating-point operations, it can detect penetration and effectively handle degenerate situations. Finally, its code is elegantly simple and thus would be straightforward to implement. As such, I would expect that this paper will be a cornerstone in collision detection for some time to come.

4.2 General Comments

I was extremely impressed with this paper from start to finish, and my criticisms are correspondingly mild. In regards to the algorithm itself, I found very little to criticize. It seems as if the author carefully considered and fairly represented this algorithm in the context of the current state-of-the-art, and that he diligently explained each possible "problem" or "special case" that might arise in its implementation. The elegant simplicity of the algorithm (and its consequently reduced CPU cost) was beautifully outlined in pseudocode that could probably be translated into usable code by even a novice programmer. The experimental results indicated that its performance was also quite laudable relative to other methods, further securing its place

in the field. As it was presented, the algorithm seemed to have no significant drawbacks and based on the results, it should work for a significantly wide range of polyhedral complexities, provided that their shapes are convex.

An added bonus was that the paper was presented in such a way to effectively reach both the novice and the expert reader. The introduction assumed that the audience had little or no background in collision detection, and the author took great pains to define all of the key terms that were used. The proofs of the theorems were left for the appendix, so as not to muddle the reader's understanding of the main focus of the paper. Additionally, while I was reading the paper, nearly every time I felt a little confused by a statement, a figure was included to help clear things up. The use of pseudocode was also an instrumental companion to a clear understanding of the details and intricacies of the algorithms.

Despite my rousing endorsement of this paper's inclusion in the pantheon of quality collision detection literature, there were a select few criticisms that I will point out. First, I will admit that it seems possible that the tests that were used to compare the three different algorithms were selected specifically to "showcase" the strengths of the V-clip algorithm. Perhaps other tests could have been selected that would have better shown the strengths of the other tests. As it was, the only result that was reported to be in favor of a method other than the V-clip was the case for the highly complex tessellated sphere under conditions of low coherence, in which case the enhanced GJK method was reportedly superior. Although I do believe that the V-clip algorithm is probably superior to the other methods for most objective measures, the author may know of cases which this is not true that I, as a novice to this field, am not aware of.

I also found it interesting that tree-based collision detection methods were only mentioned in passing in the introduction, and never mentioned again in the rest of the paper. The author mentioned that algorithms that are able to detect penetration, such as the V-clip, could be used to build a hierarchy for use in collision detection calculations for non-convex objects. It would have been interesting to see how the V-clip algorithm used in this manner measured up to some of the other more celebrated hierarchy-based methods, such as the OBB-tree algorithm [3]. It also would have been interesting to see how some of those same hierarchy-based algorithms compared to the other algorithms for convex polyhedra.

Also, there was one concept that could have been explained a little better in my opinion, and that was the concept of "compound exclusion." While I was able to understand the concept in theory, I was not able to visualize an example of compound exclusion, and the one figure that supposedly contained an example of this phenomenon was not clear to me. This was really the only time that I felt that a specific figure was warranted but not provided, but this may be due to my inexperience in the field, and thus this is only a minor criticism.

Regarding this paper in the context of other collision detection literature, I was not able to find any direct contradictions to its claims, likely because it was published fairly recently (1997). In fact, as far as I can tell, this method currently may be considered a state-of-the-art method for collision detection.

5. Discussion Questions

- Now that we have looked at three different algorithms for collision detection (GJK, OBB-tree, V-clip), which do you think is the best for a given application? What are the issues involved in deciding?
- Often times when there are two or more very different solutions to a problem, a better solution than either of the original ones can be found by using some of the ideas from each solution. Is it possible that a better collision detection algorithm could be implemented using a combined feature-based and simplex-based approach? Why or why not?

6. References

1. Lin M.C. (1993) *Efficient Collision Detection for Animation and Robotics.* PhD thesis, University of California, Berkeley.

2. Gilbert E.G., Johnson D.W., Keerthi S.S. (1988) *A fast procedure for computing the distance between complex objects in three-dimensional space.* IEEE Journal of Robotics and Automation, 4(2): 193-203.

3. Gottschalk S., Lin M.C., Manocha D. (1996) *Obb-tree: A hierarchical structure for rapid interference detection.* In SIGGRAPH Conference Proceedings. ACM Press.