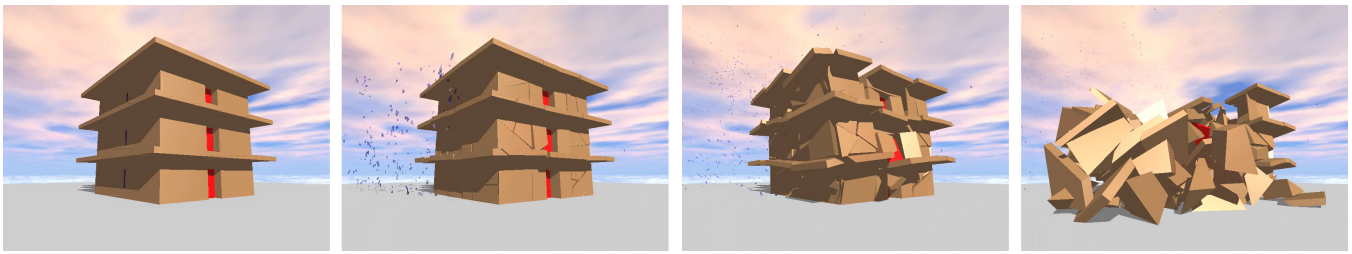
A Fast Fracture Method for Exploding Structures

Gabriel Taubman *
Brown University

Edwin Chang †
Brown University



1 Introduction

Explosions and the blast waves that follow in their wake are amongst the most devastating forces created by man. Their simulation is of use in fields ranging from game design to film effects. Previous work in fracture generation has been divided into accurate, yet computationally expensive methods such as O'Brien et al. [1999], and less believable real-time techniques as found in Martins et al. [2001].

Our work aims to strike a middle ground by providing a much more believable and visually attractive simulation than Martins et al. [2001] at a fraction of the time of O'Brien et al. [1999]. We follow the work of Martins et al. [2001] which uses a connected voxel method as a basis for fracture, and approximates blast wave forces using the Friedlander equation [Baker 1973] to achieve real-time performance. We then expand upon this in two ways: We precompute fracturing our structures into arbitrary meshes of convex polyhedra instead of using a warped voxel grid, and we propagate force along the connections between these fractured components. This adds a much greater level of realism to the simulation because pieces broken off of the structure are of arbitrary shape and resemble fractured material.

2 Method

Creating The Fractured Components

The shape of the fracture components is an integral part of the realism behind this method because objects shattered into cubes appear highly unrealistic. Also, the size of each fracture component must vary depending on the characteristics of the material. For instance it would not be realistic for a wall to shatter into tiny shards.

Starting with a 2-D floor plan of a structure, our method first synthesizes a 3-D mesh made of convex blocks. The material properties of each block determine the minimum volume of its fracture components. Finally, each block is recursively sliced by a random plane through it and its sub-components' centroids until each component is of the specified minimum volume. This is what enables glass to be divided into shards, and walls into chunks. Lastly, all fracture components are linked to any adjoining neighbors.

Physical Simulation of Fractured Components

We perform physical simulation by treating the entire scene as a graph of connected components with the fractured components as nodes, and the links between them as edges. The simulation is a two step process. First, we calculate the new position of each connected component group as a whole. If there are any collisions between connected components, finer scale collision testing is performed. Then for every object in the scene, the blastwave induced impulse is calculated. If the relative impulse between two fracture components is greater than their link threshold, their link is broken. Next, force is propagated along the links so that a collision in one area of a component can make another area shatter. Finally, the connected component graph is updated and the objects are moved to their new positions.

3 Conclusion & Future Work

Our fracture method is robust enough to simulate object-object fracturing. For example, shown in Figure 1, we shattered a window by launching a skull through it. Some areas for further development include the ability to slice non-convex objects with planes. This would allow to shatter any mesh. Also, we could introduce new parameters such as how much fracture transfers across a link. This would let us specify a level of brittleness for each material.

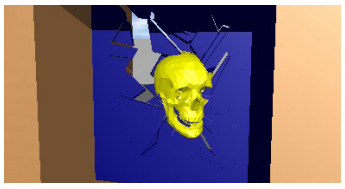


Figure 1: A skull shattering a window

References

BAKER, W. E., 1973. Explosion in air.

MARTINS, C., BUCHANAN, J., AND AMANATIDES, J. 2001. Visually believable explosions in real time. In *Proceedings of Computer Animation*, Computer Animation, Seoul Korea, 237–247.

O'BRIEN, J. F., AND HODGINS, J. K. 1999. Graphical modeling and animation of brittle fracture. In *Proceedings of SIGGRAPH 1999*, ACM Press / ACM SIGGRAPH, Computer Graphics Proceedings, Annual Conference Series, ACM, 137 – 145.

*e-mail: gtaubman@cs.brown.edu
†e-mail: ewchang@cs.brown.edu