Shattering wood
Fracture wooden materials with Voronoi diagrams

Adam Alsegård*

TNCG13 - SFX: Tricks of the trade
Linköping University - Media Technology
Demo video

December 19, 2017

Abstract

The aim of this project was to simulate how wooden objects bend, break and splinter. The simulation was implemented with Python scripts for Maya rendering software. A basic scene with three falling logs was created and the logs were then fractured with Voronoi diagrams to resemble wood. The results are somewhat satisfying but no simulation of the actual fibers has taken place due to shortage of time. The report ends with a discussion of the results and multiple possible further improvements.

I. Introduction

Simulate shattering wood. So easy and yet so hard. There are multiple ways of destroying wood on a large scale. Most SFX studios use their own plug-in or one of the abundant proprietary destruction softwares, for example Pixelux DMM (Digital Molecular Matter), ShatterFX or Pulldownit.

These plugins can simulate a large scale destruction quite well but doesn't really simulate the constraints or anisotropic behaviour given by the wooden fibers. Wood have a very characteristic way of fracturing. It bends before it breaks, it is easier to break along the fibers than across and when it breaks it snaps hard. A generic example is shown in Figure 1.

Figure 1: *An example of how a wooden plank can snap.*

The objective of this project was to create a fracture simulation of wooden objects in Maya by implementing Python scripts.

*adam.alsegard@gmail.com

II. Theory

As it turns out there exist multiple articles on how to model, texture and render wood, such as the work by Yin et al. [1] and Liu et al. [2]. There are also some research going on about fracturing that can be applied to wood but is not really focused on wooden fibers, like the work by Akagi [3], Pfaff et al. [4] and by Schwartzman and Otaduy [5].

The most relevant recent work on wooden fibers is the method proposed by Sutherland [6], which is a fiber based modeling method. It represents the fibers as rods and achieves the anisotropic fracture behaviour by calculating fractures within and between the rods. Even though this method achieves physically correct fractures it is very slow in calculation and no implementation in a real rendering engine had been implemented.

A more in-depth analysis of the mentioned methods can be found in *Wood fracture simulations with a fiber based model* by Alsegård [7].

III. Implementation

Because the theory behind simulating wooden fibers (i.e. close-up fracturing of wooden object) seems to be lacking at the moment it was decided early on that this project instead would aim to implement a simple fracture simulation and use different techniques make it resemble wood as much as possible.

There are several techniques one can use to fracture/shatter objects in Maya. The technique this project focused on was an algorithm following Voronoi Diagrams, as proposed by Qiu [8].

Before one can destroy any object however they need to be created. Therefore the first step was to write a script that created the scene. The inspiration of the scene was the first example in this demo [1] made with Pulldownit where three logs falls on a concrete floor. A similar scene was created with a Python script in which the logs also were connected to a gravity field and the floor was set as a passive rigid body. Now the scene was set to create the fractures.

i. Voronoi fracturing

The first step of the fracturing algorithm is to generate points in and/or around the objects. For this a Gaussian distribution was used in combination with polar coordinates. The logs were created as polygon cylinders with the circle in x,y-axis and the *thickness* in z-axis. The polar coordinates were used for x and y coordinates with a Gaussian distribution used for the radius. A standard deviation of half the width of the logs were used. The angle was generated with the default random function provided in Python.

A thickness of 2 was used to generate the z coordinate. This gives an interval of [-1, 1] around the center of the fracture. The reason such a small thickness was used will be discussed in Section V.

The second step is then to duplicate the original object for every point. All the rest of the operations will be performed likewise on every duplicate. For every point, find the vector to all other points. For each vector find the midway point and the perpendicular vector in that point. Then slice the corresponding duplicate with a plane going through the perpendicular vector (using Maya's built-in function *polyCut*).

This will leave a hole in the mesh. To close the hole (i.e. triangulate the mesh) the *polyCloseBorder* function was used. Then a specific material can be assigned to the cut face if desired.

When all duplicates have been sliced to shards they are defined as a set of rigid bodies with collision margin set to 0.0, collision shape set to *hull* and glue shapes turned on. Otherwise the shards will cause an explosion in the simulation as the shards share vertices and therefore already are colliding when using the default settings.

The artist can then choose a number of parameters to control the characteristics of the simulation, such as the threshold of the collision, how brittle the objects are and so forth.

IV. Results

After the fracturing was implemented a few tests were rendered. When the tests were good enough a bark texture was added to the outer hull of the logs and another wooden texture was added to the inner parts that had been cut. A background image, lighting and shadows were also added before the final render of the scene (see Figure 2).

Figure 2: *A frame from the final render.*

V. Discussion

Unfortunately a lot of the original ideas for the project were never implemented due to a very significant shortage of time. Which in turn leads to a great number of possible future improvements.

The first would be to add a graphic user interface so the artist can change the parameters in the script with more ease. With a

[1] https://youtu.be/xfYm8y-SQlc

few tweaks to the placement of the points (i.e. the origin, radius and thickness), physical attributes of the objects and the collision constraints a wide range of materials can be simulated with the implemented fracture script. If new scenes are desired then new scripts needs to be implemented as well.

The triangulation generated by *polyCloseBorder* always uses as few triangles as possible. One significant improvement would be to increase the subdivision of these faces because then more variation to the cut edges could be generated. Wooden materials always have their interior following the patterns of the fibers, something that is not modelled in these script so far. With more triangles this could be modelled better. Using so few triangles also introduces distortions to the inner textures.

That leads us to the biggest flaw of the project, which is that no actual simulation of the wooden fibers has taken place. To do this a significant amount of additional time had to be put into the project. Something that the author unfortunately did not have.

Another improvement would be to calculate the fractures depending on the actual collision. Right now the fracture solely depends on the generated points, which are calculated as a pre-process. Even though this is the usual work-flow in the industry it could be interesting to see if it is possible to implement a procedural fracture process instead.

However, even though the project did not come as far as the author had hoped it still produced results, and the results actually resembled wood somewhat. The biggest reason why it works is the generated pattern of points. Wood have long cuts, along the fiber direction. This was achieved with a small thickness for the generated points. Thus is almost resembles a 2D-Voronoi diagram and the cuts will be long with a few short splinters in the middle of the collision.

VI. Conclusion

This project managed to produce wood-like fractures with Voronoi diagrams. The scripts can be easily extended to fracture a wide range of different objects and with the right expertise in Maya they can be used for large scale destruction. Both of wooden materials and other brittle materials such as glass and concrete.

However, to actually be able to simulate a close-up fracture of wood much more work has to be put into research on how the wooden fibers interact and break. That will be left for another project.

References

[1] Xin Yin, Tadahiro Fujimoto, and Norishige Chiba. Cg representation of wood aging with distortion, cracking and erosion. *The Journal of the Society for Art and Science*, 3(4):216–223, 2004.

[2] Albert Julius Liu, Zhao Dong, Miloš Hašan, and Steve Marschner. Simulating the structure and texture of solid wood. *ACM Trans. Graph.*, 35(6):170:1–170:11, November 2016.

[3] Yasuhiro Akagi and K Kitajima. A study on the animations of swaying and breaking trees based on a particle-based simulation. *WSCG*, 20:21–27, 2012.

[4] Tobias Pfaff, Rahul Narain, Juan Miguel de Joya, and James F. O'Brien. Adaptive tearing and cracking of thin sheets. *ACM Transactions on Graphics*, 33(4):xx:1–9, July 2014.

[5] Sara C. Schvartzman and Miguel A. Otaduy. Fracture animation based on high-dimensional voronoi diagrams. In *Proc. of the ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games*, 2014.

[6] Sean M. Sutherland. Fibre based modeling of wood dynamics and fracture. M.sc. thesis, The University of British Columbia, 2012.

[7] Adam Alsegård. Wood fracture simulations with a fiber based model. Essay 1; TNCG13 - SFX: Tricks of the trade, 2017.

[8] Ocean Tian Qiu. Maya python: Create a voronoi shatter tool by scripting. `https://oceanqiu.wordpress.com/`

2016/04/21/
maya-python-how-to-create-a-voronoi-shatter-plug-in-by-only-scripting-1/.
Published: 2016-04-21 Accessed:
2017-12-19.