Chop It Up!
Animation–Driven Modeling, Simulation, and Shading in the Kitchen

Patrick Coleman and Eric Froemling
Pixar Technical Memo #07-13 Pixar Animation Studios

1 Animated Chopping

In Disney Pixar's *Ratatouille*, the creation of believable cooking environments with all their complexity has been an important element in presenting a rich world that helps draw the audience into the story. Part of that complexity arises in the preparation of food before cooking. To create complex animations of food in preparation, we designed a system that uses an animated cutting object, such as a knife, to procedurally model, simulate, deform, and prepare for shading various geometric food models as they are sliced, chopped, peeled, or otherwise broken apart. The motion of a knife (or other object) is analyzed relative to the food model to determine a sequence of cutting operations that will remodel the object as a collection of pieces. As each new piece is created, it is added to a physical simulation to generate believable response motion. We transfer surface shading parameterizations and scalar fields to resulting faces that correspond to surfaces on the original object, and we generate additional scalar fields to assist users in shading new internal surface faces. This approach to creating chopping effects entirely dependent on an animated knife allows animators to focus on character performance without needing to consider the complex modeling and motion associated with chopping.

2 Algorithms

We determine a sequence of cuts by locating the extrema of a knife's motion relative to a user–provided cutting direction, most often downward. We convert the motion of each slice into a geometric representation—the *slicing surface*—with which we break the model into smaller pieces by way of polygonal Boolean operations. To incorporate pre–existing food model animation into the slicing surface, we trace the knife motion in the food object's potentially–changing object space and transform this to a world space representation at the time the cut is applied. This also allows us to create cuts for food models that are animated relative to a static cutting object, such as a grater or a slicer. From the moment of cutting, a physical simulator controls of the motion of the new pieces (Figure 1, left). Users can apply nonlinear deformations or other procedural motions both before and after the cut. This allows for the layering of effects such as bending or peeling over the simulation.

To simulate the chopped pieces, we use a rigid body simulation library that constrains velocities to resolve collisions[1]. We also experimented with analytic collision resolution; the complexity of the simulation and unrealistic interactions commonly present in the original animation cause implausibly large response forces and impractical increases in computation time. For further optimization, we only simulate the dynamics of chopped pieces from the moment they are cut until they come to a stable rest. When users require specific control over the initial response to a cut, they can disable the use of the cutting object as a collider in favor of instantaneous translational and rotational impulses. We apply these in the coordinate frame of the cutting object to produce consistent response motions as orientations change. To model the adhesion of small food pieces to real knives, we constrain them to the knife's transformation for a small and randomly distributed length of time.

We render our models as creased subdivision surfaces [DeRose et al. 1998]. After labeling faces as members of internal or external sets using a pair of distance metrics, we crease edges that partition the sets as well internal edges that result from multiple cuts at different angles. To shade the cut model, we use the internal/external labels to blend between the shader of the original model and a user–provided internal shader. External scalar fields, surface parameterizations, and texture space coordinates are copied from the original model, and we apply interpolated values to the new external vertices created by the cut. We apply scalar fields representative of surface distance and centroid distance to the internal vertices, similar to the underlying parameterization of Owada et al. [2004], although we use the scalar fields to create procedural shaders (Figure 1, right).

Figure 1: A chopped model undergoes simulation (left). Shading of a sliced model using procedurally generated scalar fields (right). (c) Disney/Pixar. All Rights Reserved.

3 Discussion

We have used our chopping system for the production of a number of shots in *Ratatouille* that required the chopping, slicing, and peeling of food models. As the nature of film production requires us to provide user control over any aspect of shape, motion, and shading, we have decoupled the stages of the chopping system. This, along with our integration into the Maya animation package, allows artists to alter the resulting shapes or to blend the simulation with keyframe animation. We typically apply the model cutting independently of simulation, although running them together allows us to create chopping motions with repeated cutting interactions. While our chopping system is focused on the cutting of food, we believe our approach and overall design are applicable to the creation of more general complex breaking motions from simple interactions.

References

DeRose, T., Kass, M., and Truong, T. 1998. Subdivision Surfaces in Character Animation. In *Proceedings of SIGGRAPH 1998*, ACM Press, New York, 85–94.

Owada, S., Nielsen, F., Okabe, M., and Igarashi, T. 2004. Volumetric Illustration: Designing 3D Models with Internal Textures. *ACM Trans. Graph. 23*, 3, 322–328.

[1] Open Dynamics Engine. http://www.ode.org