Anyone Can Cook –
Inside Ratatouille's Kitchen

Siggraph 2007 Course 30

August 5, 2007

Course Organizer: Apurva Shah, Pixar

Lecturers:

Jun Han Cho, Pixar

Athena Xenakis, Pixar

Stefan Gronsky, Pixar

Apurva Shah, Pixar

About This Course

The passion for cooking and food are the central theme of Pixar's recent film - Ratatouille. This complex and multi-faceted problem posed many challenges that were solved using diverse computer graphics and production techniques. In this course we will comprehensively cover all aspects including modeling, dressing, shading, lighting and effects.

The story called for working cooking stations and sloppy mess of a busy, functional kitchen. We will review some of the set concepts, visual framework and even dynamics simulation techniques that were used to create this illusion. We will illustrate with several examples including final plated dishes, mis-en-place setups and the Food Locker.

The challenge of shading food on Ratatouille was to work within the stylized look of the film and yet keep it recognizable and appealing to eat. We developed subtle illumination techniques that added up to a general approach we could use on a variety of objects. We will breakdown examples ranging from vegetables to plated dishes.

Lighting played a key role in making the food look appetizing, a task further complicated by different types of food such as bread, cheese, soup and wine that pushed the boundaries of standard surface based lighting. We will discuss our general approach to lighting food as well as specific challenges and solutions posed by the various dishes.

The film called for diverse cooking techniques ranging from chopping and peeling, to stirring and ladling. We will discuss Ratatouille’s creative problems and the underlying challenge they represent as well as our solutions to them. We will also apply it to various case studies like chopping carrots, rolling dough and preparing potato-leek soup.

Although the course specifically discusses how we dealt with food in Ratatouille we want to emphasis that the basic approach and techniques can be used for other complex, multi-disciplinary visual challenges.

Prerequisites

Intermediate knowledge of 3D workflow, procedures, and terminology is helpful but not required. Topics range from intermediate to advanced.

Lecturers

Jun Han Cho, Sets Technical Lead, Pixar Animation Studios

Han graduated from UC Berkeley in 1995, after which he worked for two years in video game production. In late 1997 he joined Pixar where he worked on A Bug's Life, For the Birds, Monster's Inc., Finding Nemo, The Incredibles and Ratatouille. His focus on Ratatouille was to oversee and build sets and props that couldn't be sculpted in Maya, as well as articulating them for use by animation. His favorite food is sang-gyup-sal (Korean bacon, spicy tofu paste and raw garlic wrapped in lettuce.)

Athena Xenakis, Shading Technical Director, Pixar Animation Studios

Athena received her BFA from School of Art & Design at Purchase College in 1999. Prior to joining Pixar, she was with BlueSky Studios where her credits included "Ice Age" and "Robots". She joined Pixar in 2004, and has enjoyed shading on Cars, Ratatouille and future projects. Her favorite food is mac n'cheese.

Stefan Gronsky, Technical Lighting Lead, Pixar Animation Studios

Stefan joined Pixar in 2001. He is primarily a lighter but he has worked in other areas at the studio including tools development and shading. His credits include Finding Nemo and The Incredibles, and he served as technical lighting lead on Ratatouille. Stefan holds a BA in Computer Science from the University of California, Berkeley. He believes that a well-made burrito approaches culinary perfection.

Apurva Shah, Effects Supervisor, Pixar Animation Studios

Apurva brings a wealth of expertise in 3D visual effects and animation to his role of senior technical director at Pixar Animation Studios. Apurva joined Pixar in 2001 to work on *Finding Nemo*. He served as the effects supervisor on Pixar's 2007 film - Ratatouille. Prior to Pixar, Apurva was with PDI/Dreamworks where he contributed to the animated features, Shrek and Antz as well as live action visual effects for Broken Arrow and Batman Forever. He has taught visual effects at the Art Institute as well as the Academy of Art College in San Francisco. His favorite food is Pav-Bhaji.

Syllabus

1. Modeling & Dressing – **Cho**
    1. Introduction 7
    2. The Working Kitchen – Framework for a Functioning Space 7
        1. Spatial Concepts 8
    3. Making Cuisine – Techniques for creating visual detail 12
        1. Volume Preserving Collisions 12
        2. Soft Body Simulation 13
        3. Layering 15
        4. Rigid Body Simulation 17
    4. Conclusion 20
2. Shading - **Xenakis**
    1. Introduction 22
        1. Overall Approach 22
    2. Art & Reference – finding the stylization 23
    3. Using Scatter to Capture Translucency 24
    4. Implementing the Look 26
        1. Shading Grapes – the basic model 27
        2. Cheese, Meats and Plated Food – adding complexity 28
        3. Methodologies: Rotten vs. Fresh, Raw to Cooked 30
        4. Bread 30
    5. Summary 32
3. Lighting – **Gronsky**
    1. Introduction 34
    2. Quality of Light 34
    3. Translucency 35
        1. Subsurface Scattering 35
        2. Gummi 36
    4. Surface Highlights 39
    5. Color 42
    6. Good Food vs. Bad Food 43

4. Cooking Effects - **Shah**
    1. Introduction 45
    2. Animation Interaction 45
    3. Directed Simulations 49
    4. Active Dressing 52
    5. Scene Integration 54
    6. Conclusion – making soup from the ingredients 55

Chapter 1

Dressing and Modeling Food

Han Cho

Pixar Animation Studios

1. Introduction

The food and the kitchen that it is made in are key components of our film *Ratatouille*. We treated them almost as if they were characters that grew and changed along with the story line. It was extremely important to create a kitchen that becomes a believable working space, and to populate it with food that looks real and delicious. We'll cover concepts we employed in organizing the spaces in our kitchen and then move unto the techniques with which we chose to add realistic level of detail into the components that made up our kitchen.

2. The Working Kitchen - Framework for a Functioning Space

The kitchen of *Ratatouille* began as a simple question, what about this place would make it identifiable? Our challenge was to orchestrate an experience for the audience as to what a restaurant kitchen might be like. As with all film, the advantage of the medium is that we need only to provide it with the appearance of functionality while masking its real life inaccuracies. We began structuring the kitchen aesthetic around a simple French culinary concept, the mis-en-place (translated to everything in its place). The mis-en-place is specifically this; for whatever dish is being prepared all of the ingredients and tools are pre-measured, arranged, collected, and located in a common place before cooking begins. This facilitates being able to find all of the necessary components as the intensity and pressure of cooking gets into full swing. This concept is simple, elegant, functional, and provides a grounding sense of relationship between the chef, the kitchen, and his food.

Visually, this translates into some specific structural rules we implemented in the dressing of the *Ratatouille* kitchen.

1. Groupings of ingredients and tools provide a visual staccato of density and rhythm as your eye travels along the kitchen surfaces.
2. Functionality in the kitchen can be organized spatially as to what occurs in each location (soup prep area, meat grill, fish grill, hot pass, cold pass, coffee station, etc.) Much like the simple concept of the mis-en-place, this organizational grouping of function further provides visual structure for the audience to absorb and understand.
3. Varying the combinations of these groupings and their density can alter the scene's intent or mood (i.e. is the kitchen preparing for dinner, is it after hours, it is lunch?)

In the following examples, the mis-en-place concept is used at a very small and personable relationship - food to chef. This same concept maps in visual and organizational structure up one level to each cooking station (i.e. Linguini's Corner) or the entire flow of the kitchen. The following examples are meant to show how simple ideas can be organized on screen to allow the audience to understand the function of the space (i.e. this is a meat cooking station, this is a salad preparation area).

Figure 1: Mis-en-place setups.

Figure 2: Mis-en-place setup shown in final scene.

2.1 Spatial Concepts

Our main challenge was to begin breaking up the kitchen and allowing it to change continually with the progression and intent of the story. This is important to the audience as when the concept is done well, the audience learns and sees more about the restaurant as the main characters knowledge and confidence increases. When done poorly, it can have the unwanted effect of leaving the audience feeling static and restless, as the visual environment they are being presented with would not grow with the film.

Here are some examples from the film of the kitchen set changing over the course of the production. The space is contained and controlled to affect these changes. Please note how the examples follow the mis-en-place concept as it pertains to locations of the set.

Figure 3: Kitchen in a lonely overwhelming state versus it being a serene place.

Figure 4: Our hero in a safe space which then transforms into a dangerous space from which to escape.

The following scenes are staged so that Remy is half covered as he moves through the space (this is not only for protection but mimics his inner self as he's not part of the human world), and half open every time his ambitions pull him out to new challenges. Not only does the set's staging change simply from covered to open, but in meaning from isolated with few props to Remy opened up and surrounded by shape and color.

Figure 5: Remy still hidden and in the open.

Cause and effect relationships of set to characters also drive changes in the dressing. Linguini's corner station grows and changes along with the story arc of the character. Starting from the chaos of someone who doesn’t belong, to a space that starts to come alive with color and warmth as Linguini gradually finds his place in the kitchen.

Figure 6: The set dressing grows with the character.

3. Making Cuisine – Techniques for Creating Visual Detail

Fine cuisine employs a strong ingredient of visual design to add to the appeal of the dish. The problem for us is, since the audience cannot taste nor smell the dish, how do we translate that visual design of the cuisine to the computerized world of our movie? The audience must rely on their eyes and their imagination to determine that the cuisine we show them is delicious and worth eating. The overall look and composition of the plate is itself a large problem, albeit one tackled by our art department. We'll focus instead on the technical challenge of properly translating that design into our computer world.

What we determined early on (and admittedly this has been a recurring problem for us on past shows as well,) is that computer models built by hand or by code tend to have a rigid posed look. They lacked a "found in nature" feel to them, which one might describe as weight or contact or sag or rest. Our hand sculpted models tended to not adequately show that the soft objects had a weight to them, causing them to sag and droop and deform according to their surroundings. Piles of food tended to not look natural. They looked as if someone had carefully stacked them together (which was true!) More importantly they didn't actually look like soft objects were pushing against each other. In the end we had hard plastic toys carefully stacked together and not a crate of soft vegetables randomly thrown in. With the advent and use of soft body and rigid body simulations we decided that we had a way to solve these problems.

Ratatouille is Pixar's first large scale use of simulation in set dressing and model creation. Since modelers and dressers had little to no experience in this area we had to go through a development phase where we experimented with how we were going to apply the simulation tool into our food modeling and dressing pipeline. The most important step in the pipeline was how we were going respond to criticism and direction from art. As we learned to use a tool we had to constantly ask, can we produce a desired look with this tool? Could we make major changes after the simulation? Would we be able iterate quickly on minor changes?

3.1 Volume Preserving Collisions

Our first pass at solving the problem of making soft objects appear at rest was done by binding a lattice to the object. We had available to us fixed point size lattices that have the ability to do approximate volume-preserving collision deformations.

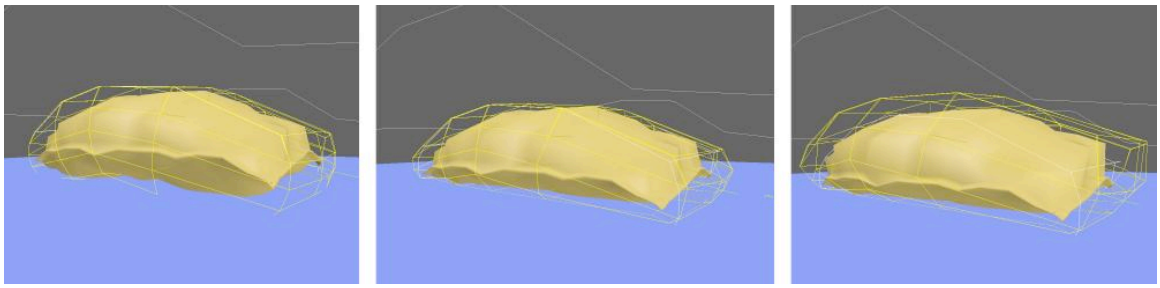


Figure 7: Potato sack with collision deformation enabled lattice. Blue surface is collision mesh the lattice is colliding against.

The results were very encouraging. Sacks had much better contact with each other and the ground. The positions and shapes of the sacks were also easily controlled with the lattices, satisfying the need for art to direct the final result. However, binding the lattice to a non-rectangular object took a great deal of time and energy. Each point of the lattice had to be repositioned to closely match the shape of the sack, so that the collision would be accurate. It was because of this, that we decided to not use this method in set modeling and dressing (although it was a significant resource for the animation of characters and props). If we had the ability to incorporate the volume-preserving collisions directly into the mesh – bypassing the lattice – we would have likely used this method in a number of our models.

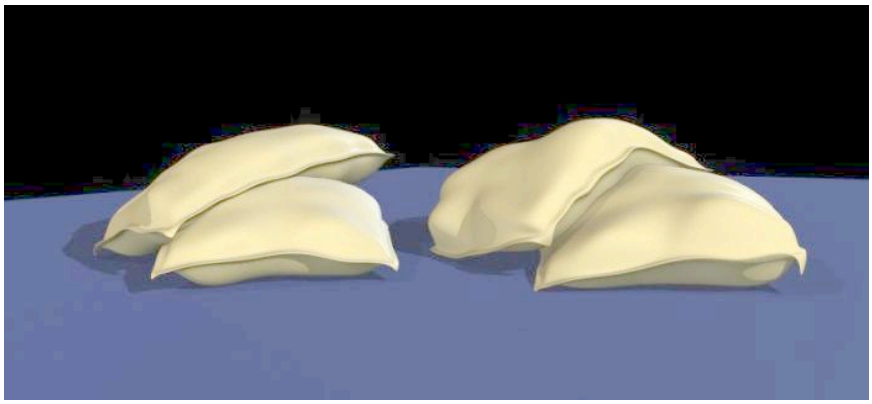


Figure 8: Potato Sack with collision deformation. Right sacks are with collision deformations (and some hand sculpting). Left sacks are without.

3.2 Soft Body Simulation

Another technique we decided to use was physBam soft body simulation [ITF04]. Unlike lattices no model specific setup and sculpting work was needed, with much of the setup work being done through scripting. In addition, simulation let us extend the scale of the project to include a large number of objects.

It made the most sense to try this to fill crates with soft objects like fruits and vegetables. Simulation is by nature a dynamic process, but in our use it needs to produce a static result. Simulations for filling crates with objects when run to their rest point led to unwanted results. Entropy would result in crates with contents that looked too flat. All the interesting visual details were simulated out.

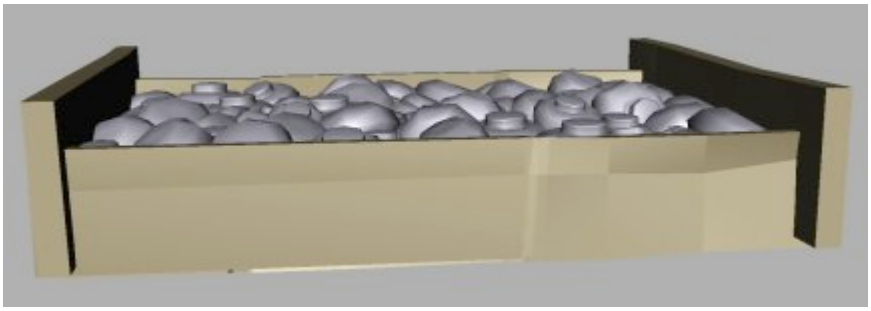


Figure 9: Crate simulated too far, looks flat and lacks appeal.

Obviously we couldn't just accept the results of a simulation. We needed a specific amount of control over the simulation so that we could bring art's designs to life. Adjusting numerical simulation parameters was one method, although it tended to be hard to art direct; there weren't always direct correlations from an art direction, for example "make this mound higher", to the parameter to adjust to achieve that direction. An easier method of control was to actually scrub through the simulation and pick what frames had the best shape and configuration.

This provided us a number of options over what the final static object would be. The ability to stop the simulation at any time turned out to be a powerful benefit.

Since in most cases we didn't see what's under the pile, we could build an artificial foundation to the pile in the shape we needed. This foundation was then used as a collision surface. We didn't have to worry about all the objects settling into the valleys of the foundation since we could stop the simulation before they did.

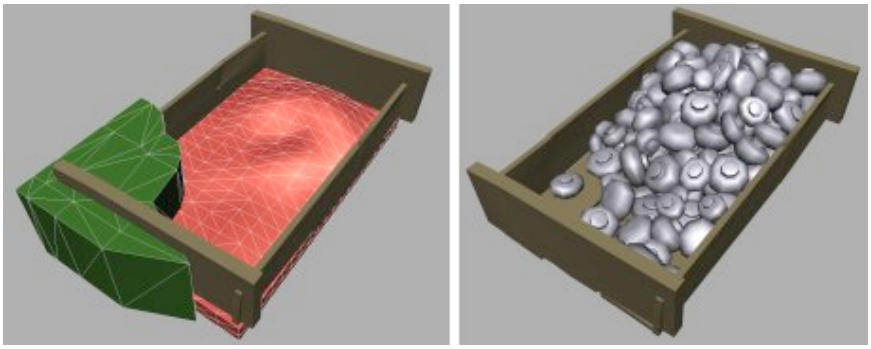


Figure 10: Crate showing collision meshes used to influence end position and final result

We weren't afraid to go back in to sculpt and position by hand. The soft body simulation handled the toughest task of fitting together the multiple pieces of fruit, leaving a small amount of final editing and beauty touchups to be done by hand. The arrangement of the fruit became real. There were no odd voids, the fruits actually looked like they were resting on each other and supporting each other, and most importantly making actual physical contact featuring the squishing that the contact implied.

Figure 11: Food crate filled with green bell peppers, un-simulated crate is on right. The bottom view clearly shows how well the peppers are now stacked in the crate.

Finally, we decided not to be concerned about certain details. Parts like stems would have been too difficult to simulate accurately, so they were left out and then reattached at the end.

3.3 Layering

The next challenge was to build a hero plate of food. The poached scallops dish is a hero food plate that occupied a lot of screen time and which also meant we needed a high level of art direction. The scallop plate was built using the lessons and techniques from the food crates, but in addition we broke up the plate into layers much like how the real version of this plate would be put together.

The leeks that comprised the bottom elements of the dish were simulated into place using the same techniques as the food crates. Leeks were given specific start positions and then allowed to fall under low gravity onto the plate. The low gravity kept them from sliding off each other too quickly before settling into a good rest pose.

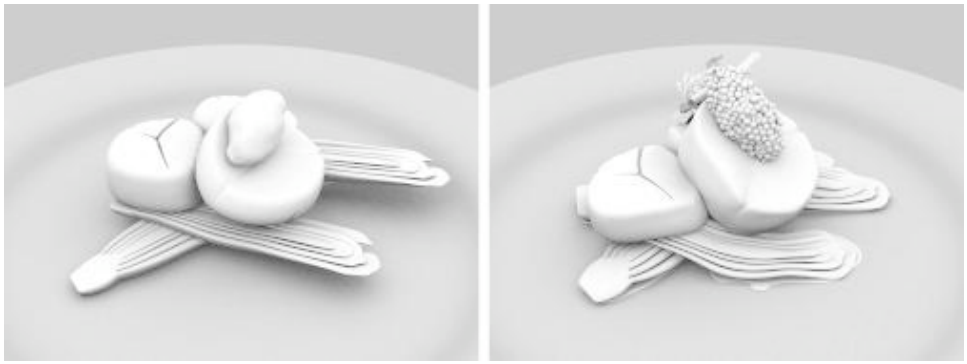
The scallops were placed in much the same manner as the leeks. The simulated frames for the scallop were inspected to pick out the best scallop shape. This shape is one that

emphasizes that the scallops have weight that causes it to sag as it rests on top of each other and the leaks, but not too much so they appear freshly cooked and not as if they had wilted.

A last physBam simulation was run to give the leeks and scallops a better fit together. With no gravity the ingredients were shrunk and then allowed to expand into each other, giving them a perfect fit. The results were usually subtle, but like spices used in cooking, really brought the plate together!

The caviar dollop was a new problem. There is an overall shape that feels like it has weight and is resting on top of the scallop. In turn the dollop is made up of a multitude of individual eggs, that each needed to feel as if they were truly squeezed in together. We could have tried to approach it like the crate of fruit and then throw away the container when we were done with the simulation. This approach would have faltered in that it would have been difficult to fully fill our overall shape.

Instead we modeled the overall shape of the caviar dollop as one object. This was soft body simulated as before on top of the scallop. The shape of the dollop then felt like it was resting on the scallop. The eggs were handled in a different way than a crate of fruit since they didn't have to feel as if they were piled into a mound. The eggs instead must only appear as if they occupy the entirety of the volume of the dollop. We could then remove simulation from contributing to the position of the individual eggs and use it only to make the eggs contact each other. A script was used to randomly fill the volume of the dollop with particles that were replaced by spheres (eggs). We did not care how we filled the space, only that there were enough eggs to fill the entire volume. Then physBam, with the dollop shell as a collision mesh, is used to grow the eggs until they contact and squeeze into each other.



(a) With no simulation (b) With simulation

Figure 12: Final Poached Scallops dish. (a) shows the dish with out simulation. (b) shows the dish with simulation.

Figure 13: Final Poached Scallops dish lit and in a scene.

3.4 Rigid Body Simulation

Although we've spent most of this time talking about how to make food look appetizing on screen, we'll move on to something a little different, trash. How does one make trash appealing? By appealing, what we really want to do is make the trash really look like a pile of trash on screen. Trash seemed like a terrific place to employ the effects of soft body simulation, composed of items that are rotting and wilted and have become soft and floppy. Unfortunately our trash piles had the problem of also being large in scale and while we would have liked to have thrown all the pieces into the simulation and have ended up with a believable pile of trash, the sheer number of elements and varying scales meant that our simulation times would have become unmanageable.

Again we tried to layer pieces of trash together, but unlike the scallops plate we didn't have discrete pieces that we could carefully place. Trash piles tended to be haphazardly thrown together so we wanted to be able to simulate in layers but also avoid showing where those layers would occur. What we ended up doing was taking two steps forward and one step back. We would simulate a layer of food, skim off the top of that layer and then simulate another layer on top of it. This avoided a top skin of the layer that felt as if it had no weight bearing down on it. A collision mesh around the outside and a mound like collision mesh on the inside gave us the ability to control the overall shape of the pile (and kept the pile from falling apart into an un-interesting shape.)

Figure 14: Final trash pile

As we moved onto an even bigger project, the compost-pile, we realized that even the method used for the trash pile would take us too long and that art directing the model would be a tedious process. Our main goal was to speed up the time it took to do simulations while being able to retain the same feel of contact and weight we had in the trash pile. The technique that we came up with relied mostly on rigid simulation using ODE [SMI06], combined with a judicious use of physBam soft body simulation.

Only the upper surface of the compost pile was modeled (the rest below was not seen.) The majority of the components that made up the pile were pieces that we pre-deformed into at least four different variants. Our attempts to rigid body simulate these pieces into the proper mound shape proved impossible if we tried to drop them into a natural pile. Story, art and layout had very specific requirements for the shape of the mound. The solution was to run the simulation upside down. The specific mound shape was built as a single collision mesh and then turned upside down to act as a bucket and catch the pieces dropped into it. Experience showed that if we used more pieces than were seen and then ran the simulation to a steady state (unlike the physBam simulations), the resulting pile looked like all its pieces were truly wedged together. Afterwards the pile could be turned right side up and all the extra pieces added for the simulation culled.

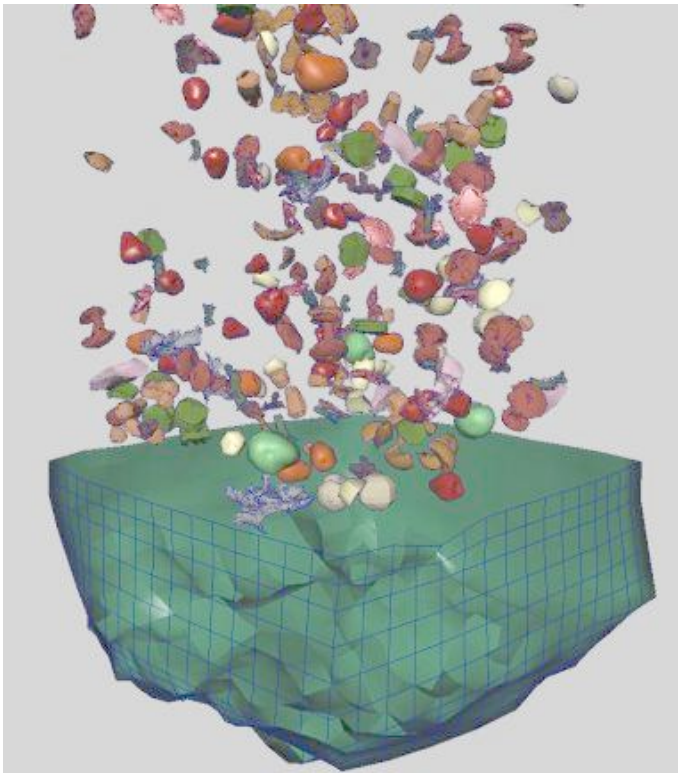
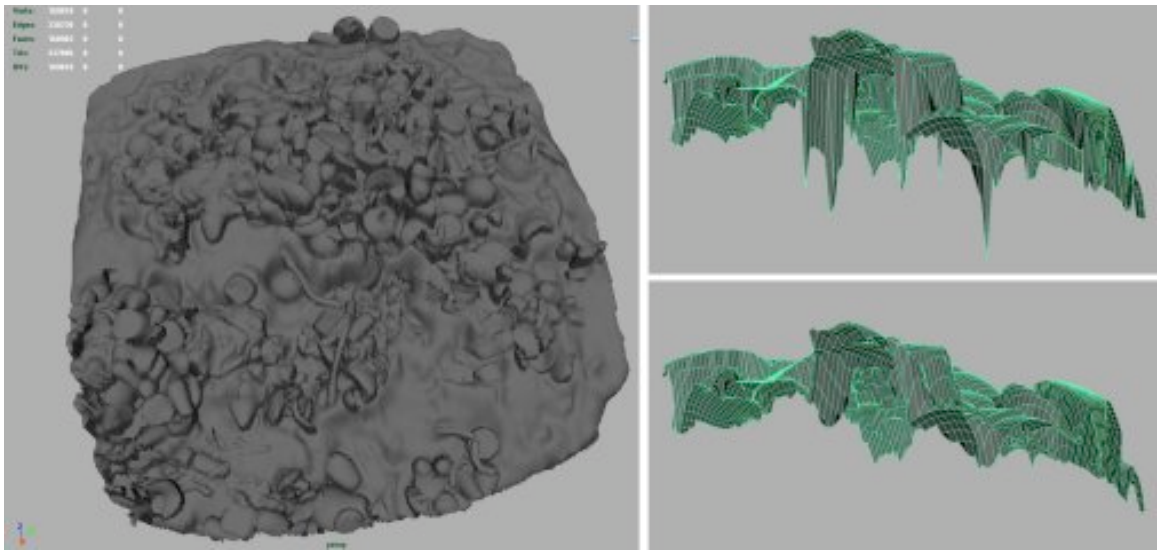


Figure 15: Initial state for compost pile simulation. Note the upside down green surface used as a collision mesh.

The end result was a simulated pile that had the proper density and contact. ODE simulations were much faster than physBam so we could quickly re-simulated in response to art direction.

The next step was to start adding detail to the model by layering components on top with a physBam simulation. A single mesh collision surface was made using a zmap render of the top surface of the mound. This collision surface proved to be faster to simulate against, as opposed to many individual collision meshes of the real pile. It was over this that we carefully simulated small groups of compost in specific areas using the methods we learned on the Poached Scallops dish. This step allowed art a high amount of freedom in directing the final look.

This result however had too much visual density. The eye quickly got tired of looking at all the individual pieces. So we ended up culling out large chunks of compost from the surface. We used the same method to create the collision mesh for the physBam simulation (but with a camera from below and with the original mound surface visible,) to create a "dirt" surface. The dirt surface fit snugly under and around the simulated compost that was left. Properly shaded, this surface gave us the rest of the mound shape as a visual background or rest point for the eye, making the simulated compost more appealing.



(a) dirt surface (b) collision surfaces

Figure 16: Surfaces used to compose compost pile. (a) shows the base dirt piece. (b) shows the collision surface used in physBam simulations. High frequency valleys were removed as shown in bottom picture as those caused problems with simulation.

Figure 17: Final Compost Pile

4. Conclusion

There were certain concepts that we wanted to embrace when we dressed the kitchen in Ratatouille. The main concept was the mis-en-place and its use to create clusters of dense visual detail without making the set itself too busy. Using this as a foundation we were able to make the set grow and change along with the characters as the story progressed.

The technical challenges were in creating a believable look to the food that comprised these visual clusters of information (or mis-en-place groupings.) Using techniques not normally considered for sets work, we were able to bring a whole new level of life to our sets models. The key lessons for us were in being flexible with what tools we used and how we used them. The results we believe added a whole new set of ingredients that make up the films final appeal.

Acknowledgments

Thanks to David Eisenmann for contributing to the set dressing lesson and being a great team leader. Jacob Brooks for figuring out a number of things we covered and producing some fantastic food models. Martin Nguyen for getting us started on using physBam for set modeling. Josh Jenny for making the compost pile. The modelers, Ivo Kos, Andrew Dayton, Raymond Wong, Gary Schultz, Jae H. Kim and Mike Krummhoefener, for building such beautiful models. The set dressers: Phat Phuong, Tom Miller and Suzanne Slatcher for putting it all together. Our manager and sous chef, Michael Warch and coordinators Paul McAfee and Evan Smith for keeping us on track. And finally, Michael Fong our supervising technical director.

References

ITF04 – G. Irving, J. Teran, and R. Fedkiw, Invertible Finite Elements For Robust Simulation of Large Deformation, Siggraph 2004

SMI06 – R. Smith, Open Dynamics Engine Users Guide, www.ode.org, 2006

Chapter 2

Shading Food: making it tasty for Ratatouille

Athena Xenakis
Erin Tomson
Pixar Animation Studios

1 Introduction

The challenge of shading food on Ratatouille was to work with a stylized look that fit into our world, yet still be readable and recognizable as something appealing to eat. We as humans have a built in sensory system to know what looks right to our eyes and stomach, and finding that acceptable (and tasty) look was the main focus. To achieve this we used subtle illumination techniques that became a general approach we could use for a variety of objects. In this course we will go over a brief technical overview, then descriptions of different concepts, techniques and systems to achieving the look.

1.1 Overall Approach

Our food shading was based on skin and scatter computations, which had improvements in speed and look internally since the Incredibles. The shaders were mainly a small variation on our in house skin. We took full advantage of this technology base, especially since there is a breadth of

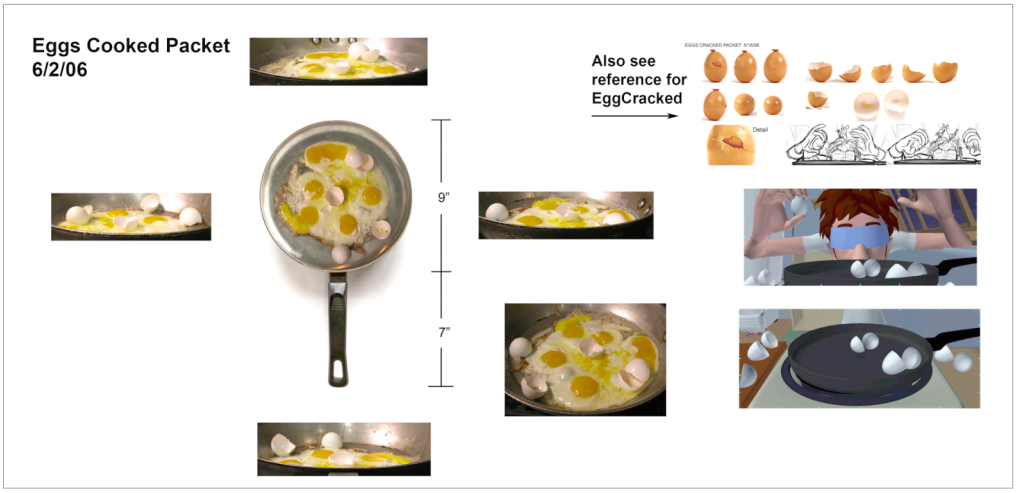
research and precidents in the industry [Jensen, Marschner, Levoy, Hanrahan 2001] [Jenson, Buhler 2002].
Skin has always posed a challenge, since there is a familiarity to what the surface qualities should be. Food had a similar problems when interpreting into cg, as well as shared properties of translucency and a living organic feel (or post-life in the case of food).

For most of the shaders we wanted to expediently feed the models into our pipeline quickly. We did this by taking advantage of Slim, which had a great level of flexibility. By creating networks that could input a variety of different shading elements into our templates (procedural noises, textures, etc..), then using scatter on the results, we added an additional level of visual complexity to our otherwise "typical" slim shaders.
For items that did not fit into this system easily, such as bread & wine, it became a special case shader that was dealt more on and individual development project basis.

2 Art & reference - finding the stylization

Our challenge, working closely with art direction, was to make our food appealing but not entirely "real", which could distract or stand out from the rest of the film. We did not directly apply any real world photographs as textures, but photos were heavily adapted, modified and painted over at times. (It was very easy to obtain plenty of reference, and there were many trips to the grocery store by Production).

Extensive photo references were used, including food prepared, cooked and lit for our own documentation. Consulting with chefs, and watching them work, we were able to document and



an example packet from the art department

control the whole process from raw food to meal. Part of the stylization choice was described as the doll house look. This look was created by scaling up some features larger then what they would usually be in relation to it's overall size, while maintaining realistic properties of the surface.

Working with art and direction we narrowed in on the factors that made it appealing. The key came down to these three important qualities:

- Softness
- Reflection
- Saturation

The softness was indication of light passing through the medium, things that do not exhibit internal light bounce tend to look hard and plastic. Saturation was an indication of freshness, and deep rich colors signify rich foods. Reflections (and specular) has a relationship to moisture, which most edibles appealing foods contain.

3 Using Scatter to capture translucency

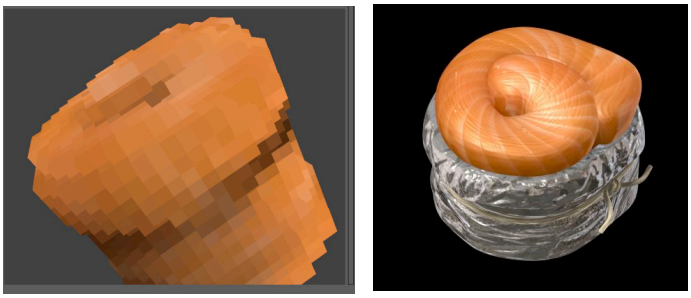
The subsurface scattering system used on Ratatouille was an extension of the system developed for The Incredibles. Both use a IIR filter to blur sampled data through a voxel grid. For The Incredibles, different scatter lengths were used for the red, green, and blue channels, which allowed for the characteristic warm glow of backlit skin while maintaining conservation of energy.

the Incredibles scatter model pushed resulting in waxy look

For Rataouille, an exaggerated subsurface scattering look was required that the Incredibles approach didn't accommodate. As the red scatter distance was pushed further, green and blue would dominate in areas where the red had been pushed out. This caused an undesirable "waxy" look. A different application of the technology was needed.

Rather than using different scatter lengths for each channel, the Ratatouille system simply scattered the irradiance uniformly, tinting the result both before and after scattering. This allowed for more artistic control in visualizing the light absorption, rather than differing scatter lengths per wavelength, which was more appropriate for the looks we were trying to achieve.

In addition to that, we non-linearly combined the diffuse and scattered illumination. Areas with no diffuse illumination would have full contribution from scatter, and areas where the diffuse illumination is bright would have little or no contribution from scatter. (essentially a max of the diffuse and scatter, smoothed to avoid discontinuities). Though not physically accurate, this allowed us to increase the amplitude of the scatter contribution without "blowing out" the bright diffuse areas. It also allowed us to maintain geometric detail in the diffuse component even with very large scatter distances.



Representation of a voxel grid with rgb irradiance and a resulting image

Unfortunately, sometimes geometric detail was maintained too well with this approach. To alleviate that, we introduced the concept of "diffuse softening". Diffuse softening was a second scatter pass with no tinting and a short scatter length. We used this in conjunction with the primary scatter contribution, by replacing diffuse with a mix of diffuse and the diffuse softening result.

Applying this softening only on diffuse allowed us to pickup displacement and specular surface information, thus maintaining sharp details via highlights and reflections. The overall visual effect of the diffuse softening and scatter illumination was a loose representation of bounced illumination.

Illumination *Adding subsurface and diffuse softenin*

4 Implementing the Look

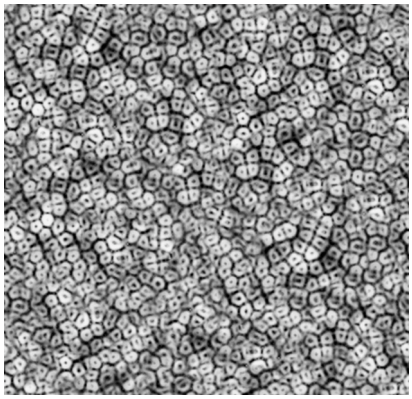
Once we had found our general translucent look, the next step was to apply this to a wide variety of materials. We had objects to shade such as crusty bread, raw fish, cooked meats, fruits, vegetables, cheeses and sauces. There were several techniques and implementation details we discovered along the way.

Since there were often large ridges, dents and holes achieved through displacement, it was important to use the displaced position in our scatter grids, especially for the diffuse softening element. The result was that we could push the intensity of displacements, and since they were captured inside the scatter grid, they were softened at render time.

Controlling the "wetness look" became our next concern. We controlled a lot of our surface shininess via reflection. Our surface roughness values were setup for sharp specular highlights, so they had a small contribution to the total illumination. Relying on reflection instead of specular to achieve wetness became a desirable part of the look. Trying to artifically increase our specular hilite would intensify color shaping which would provide too much tinting, leading to a rubbery or plastic feel. Since reflections were more malleable and controllable by lighting, it gave us a lot of flexibility. Though, this did make it more difficult at review time to evaluate our food shaders in still frames. Often times we had to render test turntables, and movement of light and reflections across the surface to establish that there was sufficient hilight response, it sometimes was less apparent depending on the angle.

Creating the shaders for each object then became a very artistic approach. The art packet was evaluated by the TD and oftentimes was implemented by one individual, without an additional texture painter. This lead to the richness of scatter and illuminations being balanced by one individual, and helped facilitate better control of the materials. Translucency was an abstract element to define as a painted texture (especially in relationship to standard rgb surface color), and it became easier to use hand painted control signals (float matte channels) in combination with a TD painted vertex varying data to control subsurface illumination.

We also often used procedural and tiling textures over custom painted textures per model. This allowed us to create interesting and complex patterns quickly for re-use via slim. It became important to share these elements since we had many objects going from raw to cooked, or cooked to rotten, and it would be prohibitively expensive to redo the work each time. The realization that the general shaders were the same across models, for say a grape and a tomato, and adapting a similiar shader for both became part of our technique for dealing with the volume of models.



A common tiling cellular texture that was used to modulate color and scattering

4.1 Shading Grapes - the basic model

A bunch of grapes is a good case study for our components of contributing illumination. For the shader we used tinting of scatter (with warmth), lightly patterned surface diffuse color, and allowing more translucency to dominate over the diffuse. Each grape was offset individually from the bunch by a random color shift in hue and value. We also used a very subtle diffuse blur, and a sharp reflection. With this the elements were in place for lighting to create an appealing image.

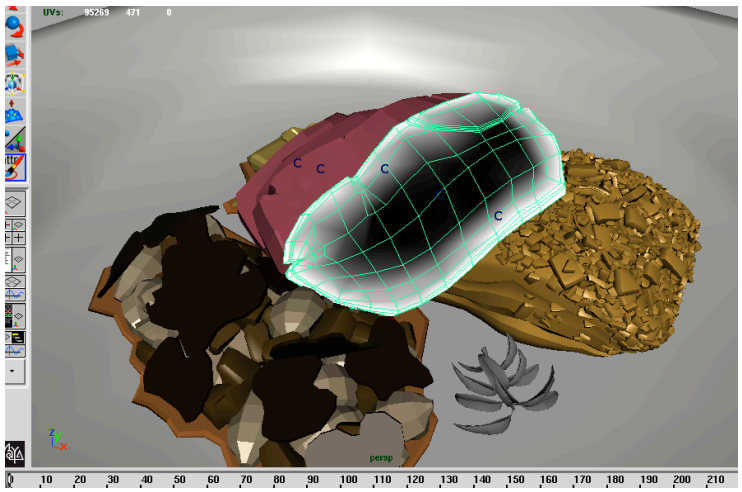
Grape shader

4.2 Cheese, Meats and Plated Food – adding complexity

Our cheese shaders were helped by strong scattering light, and having the displacement component in the scatter grid to accentuate ridges and holes. An intense diffuse blur amount was used for cheeses and meats to increase the fleshy look. Pushing the diffuse blur to the point where details were lost was actually beneficial for a waxy cheese look. Techniques, like paint integrated with procedural shading, such as painted vertex varying data to control subsurface contribution helped provide additional control in the shaders. Another key technique was using cellular textures, based on voronoi patterns to capture small details when added as a multiplier to scattering amount. This broke up the light transimission look, and gave the effect of subcutaneous details, like straition inside of meat.

The different material properties in one model (like this steak dish below) required multiple scattering grids to avoid bleeding of irradiance across discreet food types. It was undesirable to have reddish steak as part of your light potato shader. The exception to this came when shading a rotten garbage pile, where this bleed became a positive side effect for blending the shaders like a fake bounce irrandiance.

Another key to complexity was adding flecks of spices to our cooked dishes. Taking these flecks out of the diffuse blur was important, or else you would have blurry details and not the desired parsley or pepper speck look. An oily or buttery coating for all cooked foods, was another level of detail. If the objects were cooked together, they would have a consistent coating, and it was important to create that as a thin transparent layer on top of the shaders.



Vertex cluster weight on steak model

final steak image

intense scatter for cheese

4.3 Methodologies: Rotten vs. Fresh, Raw to Cooked

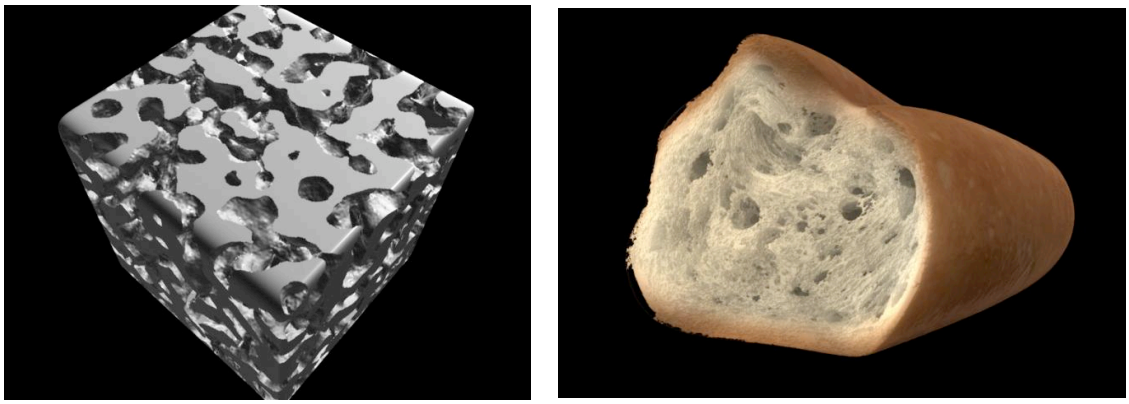
After working on food for some time, we realized that the method of starting with raw foods first was an easier approach to shading. It was more difficult to go from cooked to raw or rotten to fresh. A lot of detail in the cooked versions were helped by modeling simulation softening, and by adding displacement. Oftentimes we reused the same shaders, with some small modifications, such as adding brushed details, cooking browning, or having our cooked colors move closer together toward a warm brownish hue.

Variations of similar shaders

4.4 Bread

The bread was an example of a special case project that fell outside the general approach, and required a new technique. It still used scattering to create a soft look, but took it one step further into visualizing this softness through the formation of air bubbles inside the density of the bread.

To create this airy effect we used a multi-step approach. The bread interior was a volumetric hypertexture (Perlin and Hoffert, 1989) defined using a procedural density function. The density function was made up of several layers of procedural voronoi noise.



Volumetric hypertexture and final bread shader

In addition, the important features such as the position of the crust and the position of the torn off face were encoded in point clouds which the shader used to modify the density function.

The entire density function was also deformed using a texture map. The texture map was applied to the density field using a planar projection. Before evaluating the density function, the x and y positions of the current point were replaced by the value obtained from the texture map. This allowed us to compress air bubbles which were closer to the crust, and to give a slight twist to the pattern of air bubbles, as is often seen in real bread.

5 Summary

We see and eat food everyday, and the challenge was to find this emotional impact of positive associations. Creating shaders in a realistic way, but still fitting into our film stylization helped us do this. By hitting the key subtelities of appealing illumination - saturation, softness, and reflection, we achieved our goals for shading.

6 Acknowledgments

A very large thanks to Daniel McCoy and Byron Bashforth who were major contributors to this work. Thanks as well to the entire talented Rat Shadepaint group wrangled by Lourdes Alba. Additional thanks to Harley Jessup, Sharon Calahan and Belinda Van Valkenburg for thier wonderful art direction. Special thanks to Manuel Kraemer, Sarah Fowler-Deluna, Junyi Ling and George Nguyen who contributed to images contained in this course.

7 References

Henrik Wann Jensen and Juan Buhler. A rapid hierarchical rendering technique for translucent materials. ACM Transactions on Graphics, 21(3):576–581, July 2002.

Henrik Wann Jensen, Stephen R. Marschner, Marc Levoy, and Pat Hanrahan. A practical model for subsurface light transport. In Proceedings of ACM SIGGRAPH 2001, Computer Graphics Proceedings, Annual Conference Series, pages 511–518, August 2001.

Perlin, K., and Hoffert, E. M. (1989) Hypertexture. In J. Lane (ed) Computer Graphics (SIGGRAPH '19 Proceedings), Vol 23, pp 253-262.

Chapter 3
Lighting Food

Stefan Gronsky
Pixar Animation Studios

1. Introduction

A passion for food is a central element of *Ratatouille*. The audience must identify with the main characters' love of good food in order to understand the characters' motivations. When it comes to lighting the film, this means that a guiding principle was to make food look delicious so that the audience would be able to imagine the smell and taste of it.

What makes food look delicious? And how does this relate to the film as a whole? Well, food is most appealing when it is appetizing and fresh. That description is too vague, though. More specifically, to look appetizing and fresh:
- Food needs to be colorful as this indicates ripeness or freshness.
- The surface should appear wet, dry, waxy, or soft as appropriate. These things are visual cues of its taste, texture and feel.
- A fleshy or translucent appearance is essential for certain foods. It lets you know how ripe or juicy the food is.

To help convey these qualities we studied food photography (the kind you see in food magazines or cook books). This food photography strikes a balance between realism and pure beauty by presenting an idealized version of the food. Light falls across cuts of meat to reveal their perfectly-cooked texture, a soft backlight hints that fruits and vegetables are plump and juicy, and highlights glisten on sauces. The lighting helps to emphasize the qualities that make that food look like *good* food, and in the process it makes the whole image look more appealing.

These goals tied into the film's central visual style, which was to give everything on screen the same visual appeal that good food has. So while this course focuses on food, the techniques used to convey the richness, vibrancy, and texture of food were also used to light the characters and sets.

2. Quality of Light

We now outline the basic lighting approach and how it tied to our visual goals. How do we use our lighting tools to capture this look? How do we light the food (as well as the characters and set) with a maximum amount of control in order to meet the exact needs of each shot? And since we must light approximately 1,700 shots that depict often highly complex scenes, how do we do so in a way that minimizes long render times?

A restaurant's kitchen has bright overhead lights, but that light then bounces around from all of the reflective surfaces in the room and provides a more even illumination. As a chef is preparing a dish under those bright lights, additional light spills in from adjacent lights and bounces up from the tabletop or stovetop. We created this look by paying careful attention to how we balanced our lighting, by making our fill and bounce lights bright enough to bring out detail in the food while at the same time minimizing harshness or clipping from over-range values.

We let light smoothly wrap around surfaces instead of being sharply terminated by using area light sources especially for fill and bounce lights. Even if we had a strong light source we wanted it to feel soft and inviting rather than harsh. We always made sure that light fell on food at an

angle that best revealed the broken texture of bread or the smooth sheen of sliced tomatoes, even if that meant diverging somewhat from what the rest of the set lights are doing.

We considered global illumination technology such as irradiance and caustics, but we found that they were not really practical for the complex scenes that we had to deal with in this film. It's more important to have control over lights that provide more immediate feedback. If we wanted a red tomato to bounce light onto its surroundings, we could add a red point light with distance falloff. If we wanted a knife or a metal container to reflect a caustic pattern, we could fake that with a slide projector light instead (see the caustic pattern on the rear wall in figure 8b).

The soft look also informed our approach to shadows. We often wanted a feeling of indirect lighting. This could be from light bouncing around a kitchen, from the soft glow of chandeliers in a dining room, or from skylight streaming in through a window. Deep shadow maps (LOK00) were used and they had several advantages that were well suited to our needs. Since they are anti-aliased we could render our maps at a more reasonable resolution, and regardless of resolution we could capture fine details and semi-transparent surfaces. Very soft deep shadows also have significantly less noise than other shadow methods.

3. Translucency

Capturing light transmission and diffusion inside food was critical in creating a translucent look. We used two different techniques to achieve this.

3.1 Subsurface Scattering

Many food shaders used subsurface scattering (which we will also call simply “scatter”) to create the fleshy appearance of meat, fish, cheese, bread, and vegetables. Subsurface scattering is more in the domain of shading than lighting and it is covered in more detail in chapter 2, section 3 of this course. It was a key component of lighting food, though, so we discuss how scatter affected our lighting approach:

It was critical to avoid over-lighting. Sometimes we would turn down the fill lights (or turn down their scatter contribution) to let the scatter from the key light bleed farther through the surface for a softer quality. Don’t crank up the key light just to make scatter shoot through every crevice and let scatter naturally extend the light.

Pay special attention to scatter distance, as it conveys the density, size, and overall feeling of the food. This sounds obvious but it's important! Let the scatter bleed all the way through the object and it feels very light and jelly-like (too much and it can feel waxy or even flat). Reduce the scatter distance and the object feels dense. Sometimes we need to cheat this. If we have a large piece of cheese next to a few little slices and hit everything with one light, the light should penetrate further through the small thin pieces. We might want to reduce the scatter distance on those small pieces though to get a nice strong scattering effect without having them go totally flat. Similarly, I might increase the scatter distance on the larger piece to get a nice falloff that is strong enough to make the cheese feel nice and soft but subtle enough to convey its density.

Differentiate between back lighting and rim lighting effects. One of the most striking effects of subsurface scattering is the way that it enhances back lighting. In this case we use scatter to really show the interior color and density of an object. A rim light serves a different purpose, which is to silhouette an object or highlight its edges, and so we often turn down the scatter in our rim lights.

3.2 Gummi

Subsurface scattering provided a very rich look but it had some issues from a lighting perspective:

- Scatter emulates light scattering outwards through the surface but we often wanted something that emulated a more directional transmission of light straight through an object.

- The look of scatter is developed and fine-tuned within each surface shader so that each object has its own appropriate scatter response. Sometimes, though, we needed to adjust translucency more uniformly on a large group of objects.

- Scatter requires a pre-pass to calculate the scatter data, and this pre-pass rendering time increases with the number of on-screen objects. The render time became prohibitive in shots that had dozens or even hundreds of food items on-screen. We needed something that could approximate the look of scatter on background objects and that could render faster.

We implemented a new light type based on the translucent fishes from *Finding Nemo* that emulates this directional light transmission. We called this light a Gummi light. The core behavior of the Gummi light is that different wavelengths of light are absorbed at different rates as the light travels through a material. We needed an inexpensive way to calculate how much material the light passes through. PRMan deep shadow maps provide a "deepprevdisttotal" mode that approximates this[1]. A traditional deep shadow map stores the accumulated opacity over distance, but in this mode the map instead stores the accumulated distance traveled inside objects. Figure 1 shows how this is calculated (the spheres in this diagram represent the grapes seen in figure 2). Values start accumulating from the shadow camera's near clipping plane. At point P1 the value is 0 because no surfaces have yet been encountered, at P2 the value is (P2 – P1), at P3 the value is still (P2 – P1), at P4 the value is ((P2 – P1) + (P4 – P3)), etc.

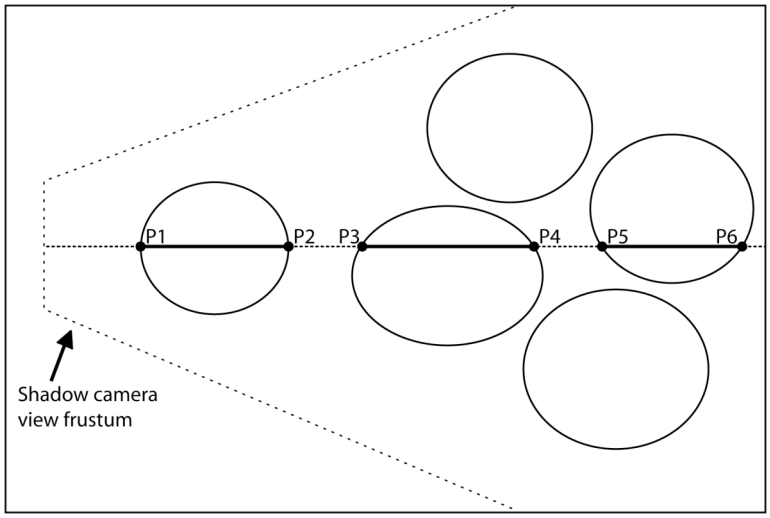


Figure 1

We called these deepprevdisttotal deep shadow maps thickness maps because they approximated the interior thickness of objects. To generate the thickness Map we used the following lines in the deep shadow map RIB file:

[1] The deepprevdisttotal display mode is an undocumented feature of PRMan. Its behavior is not guaranteed to work and it may not be supported in future releases, so please use it at your own risk.

```
PixelFilter “box” 1 1
Display “ThicknessMap.sm” “deepshad” “deepprevdisttotal”
Sides 2
```

The deepprevdisttotal requires that all surfaces are closed and backfaces are visible (hence the need for “Sides 2”). Otherwise it is no different from generating a standard deep shadow map. The Gummi light shader then uses the thickness map to attenuate the light. Below is a light shader that includes the basic Gummi function:

```
light Gummi(
    /* Note that this light has no position or direction.
     * All directional information comes from the Thickness map. */
    color lightColor = (1,1,1);

    /* Thickness Map. Map should be a deep shadow map created
     * using the deepprevdisttotal display mode. */
    string mapName = "";
    float mapBlur = 0;

    /* gummiFalloff is type color but it is really independent
     * red, green, blue distance falloff controls. Higher values
     * produce a steeper falloff. */
    color gummiFalloff = (1,1,1);
)
{
    /* The shadow() function internally does a “1-value” transform
     * before returning its value. We need to do our own 1-value
     * to undo that and get the original value. */
    float thickness = 1 - shadow(mapName, Ps, "blur", mapBlur);

    /* distance-based attenuation of each color channel,
     * which represent different wavelengths. */
    color gummiAmount = color(
        clamp(exp(-comp(gummiFalloff,0) * thickness), 0, 1),
        clamp(exp(-comp(gummiFalloff,1) * thickness), 0, 1),
        clamp(exp(-comp(gummiFalloff,2) * thickness), 0, 1));

    solar() {
        Cl = lightColor * gummiAmount;
    }
}
```

Gummi has several advantages:

- Gummi is a more directional effect that emulates light transmission straight through an object, so it complements subsurface scattering.

- Gummi appears to surface shaders as a standard diffuse light so we could use it on any surface. This was especially helpful for large groups of objects or objects that were not initially written to use scatter. We could use it on shelves full of cheese and vegetables (figure 6b), cloth (figure 2c), and even liquids like wine (Figure 3).

- Gummi is fast because the pre-pass is simply a deep shadow map and the illumination loop is not much more computationally expensive than a standard diffuse illumination model.

Figure 2a: a scene with no scatter or Gummi

Figure 2b: with scatter

Figure 2c: with scatter and Gummi

Figure 3: Gummi adds translucency to wine

Subsurface scattering and Gummi really complemented each other, and most food shots used both techniques. Figure 2 shows an example scene. In figure 2a we removed the scatter and Gummi. The cheese and bread are very dark because we reduced the fill light to give the scatter and Gummi room to spread without blowing everything out. The grapes are also quite dark and they feel solid, almost metallic. The key light from above the frame adds a bright rim to the grapes and cheese but without any translucency it cuts off abruptly. When we turn on scatter in figure 2b things get much better. The key light scatter adds a nice ramp that extends halfway through the top cheese wedge, it softens the bread, and it makes the grapes feel plump. We then add Gummi in figure 2c. The Gummi light uses a thickness map that is aimed in the same direction as the key light. Notice how the Gummi light complements the scatter. It extends the neutral-colored ramp on the cheese wedge into a deeper white-yellow-orange ramp, and it adds color to the larger piece of cheese below it. On the bread and grapes it acts like a fill light, adding shape to the areas that are shadowed by the key light and again adding color complexity with the more neutral light of the scatter transitioning into a more colorful Gummi ramp. Together subsurface scattering and Gummi create a two-stage translucency effect and we can balance the color and intensity of the two relative to each other to get a very rich look.

4. Surface Highlights

Many foods have some amount of surface wetness or sheen. Look at any meat or fish, a soup, a sauce, or even many fruits and vegetables, and notice how important the moisture content is in conveying the look. So how do we create this look? The main visual cues are the highlights that reflect off the outer layer of water or oil (figure 4).

Figure 4: The image on the left has no supplemental highlights. When highlights are added the scallops and caviar look fresh and moist and a lot more appealing.

It seems pretty clear that we want to use specular light or reflections to create these highlights in CG, but how do we truly capture this particular look? Let's think about these highlights and break them down into their basic characteristics:

Color: realize that the highlight is reflecting off the outer transparent liquid layer, not the underlying food. Transparent liquids generally reflect light without modifying its color (as opposed to a metallic surface in which the reflections take on the surface color). So even if the food is a deep red or green you will still get a neutral highlight from the liquid.

Intensity: If you have a really bright light then you would expect to see an equally bright highlight on the food, but not every highlight needs to be so strong. In addition to the main highlight you will see secondary highlights and reflections of the room. These subtle highlights really add shape to the food (especially in motion because they are view dependent) and they are equally important in creating a convincing wet look. See the subtle highlights on the sides of the scallops in figure 4.

Shape: notice the shape of the highlights on food. The highlights are broad and broken-up, letting you see fine ridges and shape changes in the surface. The first inclination might be to use specular light to create these highlights, but a traditional specular highlight is often inadequate for creating wet highlights. The only real control you have over the shape of a specular highlight is the roughness parameter. Low roughness produces a sharp, focused highlight that is too small (figure 5b). To make the highlight larger you must increase the roughness. But high roughness produces a broad smooth highlight that does not adequately break up to show off surface detail; it creates a semi-shiny plastic look or waxy look but not a wet look (figure 5c).

Let's think about why those highlights look so good in food photography. The food was probably photographed in a studio with large studio lights and reflectors or photographed in a room with a big window nearby. The highlights come from these large shaped objects reflecting on the food surface. So let's just use reflections instead of specular light! This lets us directly control highlight size, shape, and softness.

Size: a small reflection texture will produce an effect similar to a tight specular reflection. A large texture with sharp edges produces a studio lighting effect or a window highlight with broad highlights that abruptly break up based on the surface angle, which is usually the best starting point for creating a wet look.

Softness: We can control the size of the reflection texture vs. the width of the falloff at its edges, and this is perhaps the most important element in controlling the overall feeling of the highlight (in contrast, a traditional specular highlight always falls off from its center – it is not possible to define a broad center to it). Very soft highlights produce a shiny, waxy appearance that is appropriate for things like an apple or the outer layer of a red onion. You can produce this using a texture with very soft edges; the effect resembles a specular reflection with high roughness. Broad, sharp reflections produce a wet look (figure 5d). By slightly blurring the edges of the reflection texture you can soften the overall look while still maintaining shape in the reflections (figure 5e).

Shape: A rectangle mimics a window, a large hemisphere mimics a broad sky. A broken up pattern is a very direct way to break up the highlight in cases where the actual surface is too even or where you want to add more visual interest (figure 5f). This approach is very flexible because changing the highlight shape is as easy as painting a new texture map.

Figure 5a

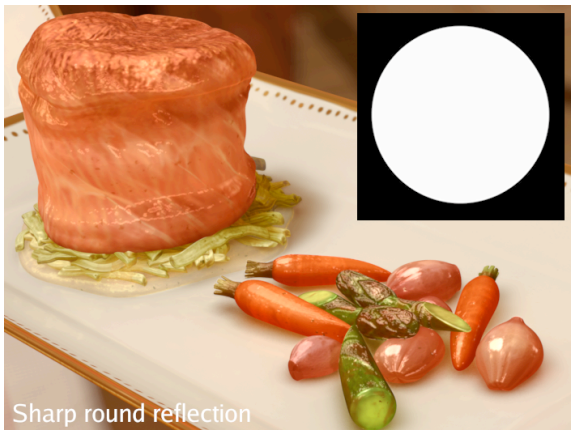


Figure 5d

Figure 5b

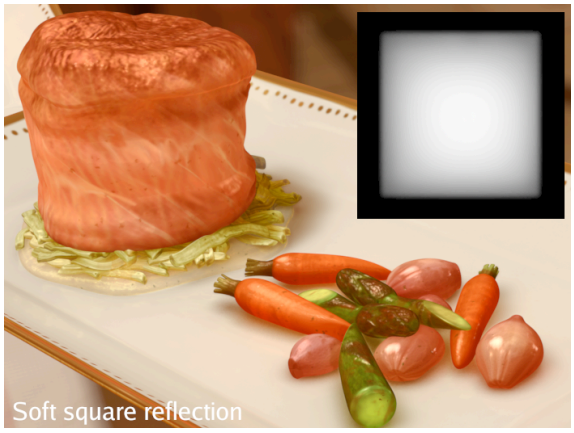


Figure 5e

Figure 5c

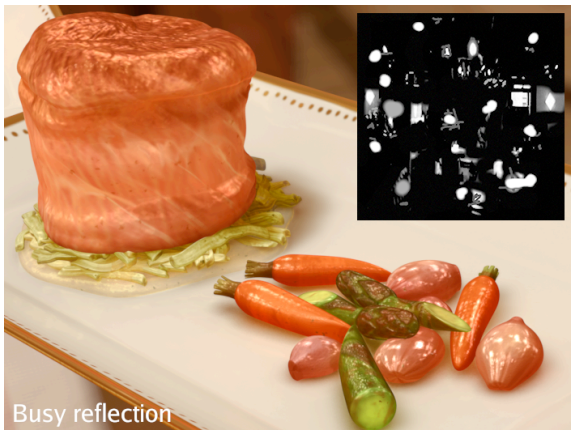


Figure 5f

Figure 5: comparing different types of highlights. The soft square reflection highlight in figure 5e was ultimately used for this shot in the film.

5. Color

Translucency and highlights address aspects of certain types of food. A more general visual cue that you will see in *all* good food is color--it looks vibrant. The challenge is to capture color hue and saturation without going overboard by making things too strong and to maintain proper color balance throughout the image.

Leave some room in the dynamic range for highlights to sparkle. This ties in somewhat with the quality of light that we were talking about earlier. We don't want to light things so that they always go close to 100% intensity in the color channels—it leaves less room before highlights clip, but it also makes it difficult to maintain subtlety in the image.

Get good color in the dark areas. Even a bright yellow lemon can feel dull and lifeless in the shadows. Pay attention to those dark areas in the frame and get some extra color in them by using colored bounce lights, for example (see the shadowed lemons at the bottom of the metal basket in figure 6a still has a strong golden color). Sometimes you need to get extra light on something that might otherwise fall into the dark shadows. In figure 6b the cheese wheels used a supplemental Gummi light to add color and prevent them from going too dull as they fall out of the key light, and the oranges and other fruit in the top right corner of the frame have a fill light that maintains some shape and color.

Figure 6a

Figure 6b

Don't turn it up, turn other things down. To help make the food look more vibrant without having to push it out to full 100% pure colors, the sets used a more neutral and restrained color palette so that the food (and characters) could stand out without feeling too strong. The scallops in figure 4 are a fairly restrained color, but because the surrounding plate and tabletop are even more subdued the scallops feel colorful.

Figure 7

A shot of the film's signature dish demonstrates how all of these contribute to the image (figure 7). The lighting is very soft and subdued, allowing highlights on the ratatouille, the sauce, and the edge of the plate to be visible without blowing out or looking overpowering. Enough fill light is used to maintain color in the crevices between the vegetables. And since the color in the set is so subdued (white plates and tablecloths, deep reds and browns) the food really pops out without feeling too strong.

6. Good Food vs. Bad Food

So far we've been talking about using soft lighting, translucency, reflections, and vibrant color to make food look appetizing. Let's look at a few direct examples of how reducing or removing these features makes the food look less appealing.

Figure 8a

Figure 8b

Rotting food vs. fresh food: Figure 8a shows some food that rats found in the garbage. By drastically reducing the highlights and making them feel more waxy rather than wet the food looks like it has dried up. Reducing translucency signals that the internal moisture has evaporated and the food has wilted; less translucency also means that color does not extend into the dark areas of the food so much. The lighting emphasizes the dimpled, pitted appearance that hints at a dry, rough texture. The base color was also shifted so that the strawberries are a more brick red and the green vegetables look yellowish or pale, and in general the color palette is more uniform so that the various food types blend together and are somewhat indistinct. Compare this to the images of fresh food in figure 8b (or to any of the images earlier in this chapter).

Figure 9a

Figure 9b

Bad soup vs. good soup: at one point in the film a pot of really awful soup (figure 9a) is transformed by some expert cooking skills into something much better (figure 9b). Let's look at the bad soup first. The base color is a very unappetizing flat brown. The most prominent feature is the uneven bubbling foamy layer on the surface. High-contrast lighting (notice the hot spots on the copper handles and on the rim of the pot) adds a harsh tone to the image, as does the high contrast of the burnt layer inside the pot. Compare this to the good soup that starts with a softer, warmer cast to the lighting. The soup is now a much more appealing creamy color. Very subtle color variation makes it feel delicate. The vegetables add another touch of color. Translucency is a key part of the potato pieces. A subtle, broad highlight falls across the surface.

The central challenge of lighting food in Ratatouille was not to make the food look purely photorealistic. Realism was certainly part of it; just as in photography of real food, the food must look natural to be appealing. The real challenge was to capture and emphasize the most appealing details of good food (vibrant, translucent, delicate) while deemphasizing less pleasing details (dull, dry, rough surface texture, burned), to simplify things to highlight those important details without looking overly complex or busy. We wanted the lighting to capture the best aspects of real food as a way to bring audiences into the stylized world of the film.

Acknowledgments

The lighting work in *Ratatouille* was all done in service of the story and to meet the artistic goals of the film's creators, so thanks to Sharon Calahan for her vision and leadership; the tools and techniques described here were developed in order to put that vision on the screen. Thanks to Justin Ritter for the code examples provided here and to the entire Ratatouille lighting technology group for designing and implementing an amazing set of new lighting tools. And thanks to Kim White and the Ratatouille lighting team for taking those tools and making a truly beautiful film, some examples of which are presented in this chapter. The lighting in these images were produced by Brian Boyd (master lighting for figure 2), Luke Martorelli (shot lighting for figure 2), Michael Sparber (figure 3, 8a), Lloyd Bernberg (figure 4), Jonathan Pytko (figure 5, 8b, 9a, master lighting for 9b), Vandana Sahrawat (figure 6a), Mitch Kopelman (figure 6b), Airton Dittz, Jr. (master lighting for figure 7), Afonso Salcedo (shot lighting for figure 7), and Jeremy Vickery (shot lighting for figure 9b).

References

LOK00 – Tom Lokovic, Eric Veach, Deep shadow maps, SIGGRAPH 2000

Chapter 4

Cooking Effects

Apurva Shah
Pixar Animation Studios

1. Introduction

Creating visual effects for feature animated films poses some interesting challenges. This is especially true if they are tightly coupled to the story and performances, as is the case with the cooking in Ratatouille. We will describe what some of these challenges were and how we addressed them. In the process we will also describe some of the technical underpinnings of our effects pipeline and shot specific techniques.

2. Animation Interaction

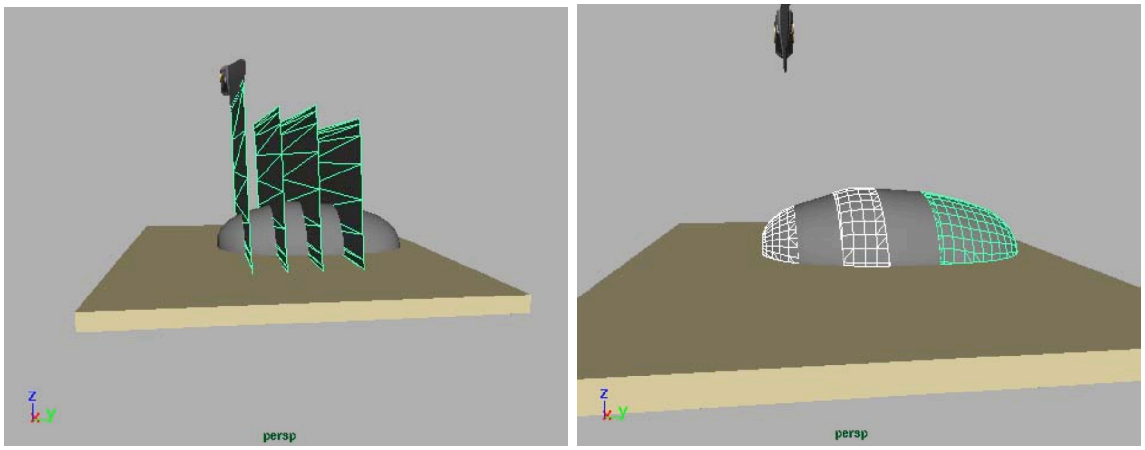
Rat's Problem: The chef needs to chop a carrot but where should the knife land so that the carrot will slice properly? OR The chef is supposed to roll dough but there is just a ball on the table.
Underlying Challenge: Iterating with animation – it's expensive and frustrating to go back and forth.
General Solution: Let animation go first with performance – miming.

Our first example is the Chopping System that was used in food prep for shots that required slicing, dicing or peeling of vegetables. Before the shots came to the effects department the animators would work with solid objects for basic position and size. After the animation was completed the main steps of the chopping workflow were:

1. Established cutting planes and pre-dicing the object.
2. Ran rigid body simulations on the pieces.
3. Transferred shading from the solid object to the diced pieces.

Three points were selected on a knife. Their movement swept out the cutting plane. The planes did not correspond exactly to the knife's motion but compensated for side

to side motion or jitter. For additional control the cutting planes could be added to, deleted or modified by the effects artist. Maya boolean operators for polygons were then used to divide the original object into the cut pieces. For a more detailed survey on constructive solid geometry (CSG) please refer to [LAI86]. Figure 1 shows the cutting planes and resulting pieces for a half potato lying on a cutting board.



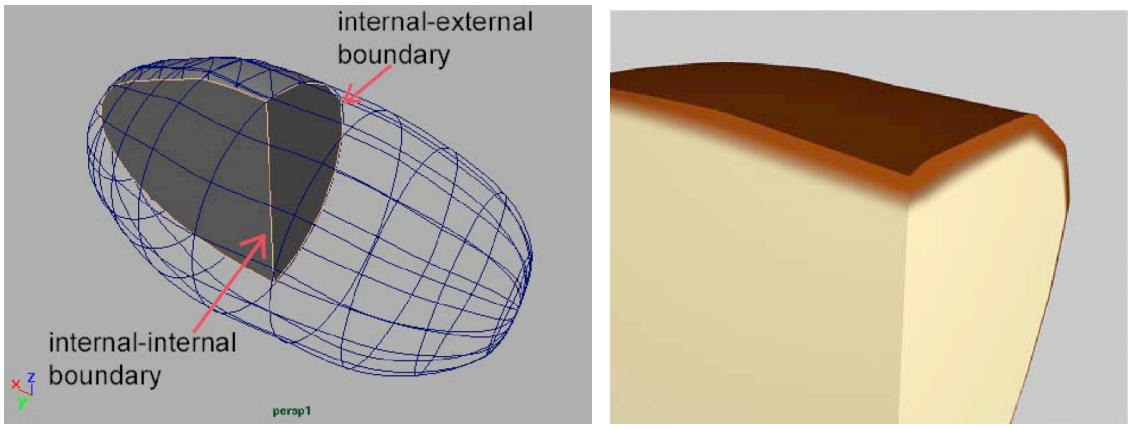
(a) Cutting Planes (b) Pre-Diced Pieces

Figure 1: A half potato solid model about to be sliced lying on a cutting boards. (a) Shows the cutting planes swept out by the knife motion. (b) Shows the pre-diced geometry.

The rigid body simulations were done using the Open Dynamics Engine (ODE) [SMI 06]. A Maya mel interface was used to define simulation properties across the pieces and to pass the data to the ODE process. The resulting transformations were read back into Maya as curve data. Initially we used the knife itself as a collision object but this often lead to stability issues requiring very small step sizes. Using impulse forces made the simulations a lot more stable even with large steps. Along the same lines, pieces were switched from passive to active only if they were on the right side of the active cutting plane. Also, when the motion of the pieces fell below a threshold for n frames they were once again made passive. In this way pieces not actually being chopped remained rock solid. Since the rigid body simulations are essentially baked into the pieces we could keyframe animate them for added control.

All shading on objects that needed to be chopped avoided parametric textures relying on procedural or projected textures instead. Transferring the outside shading

was trivial since we used the portability of slim palettes [PIX07] to transfer the shading network from the original model to the diced pieces. Primitive variables (vertex varying data) [APO02] were used to distinguish outside from inside surfaces when they were first created in the chopping step. Using primitive variables in this way allowed us to soften the edge from outside to inside when necessary. Figure 2 shows an example of this on the sliced potato.



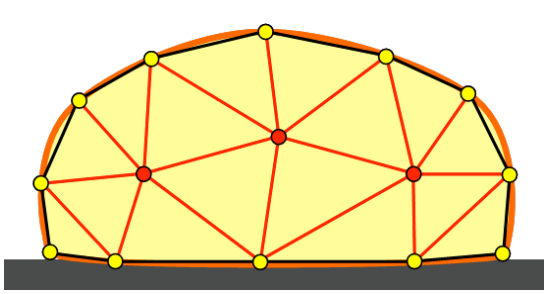
(a) Dicing Edges (b) Softened Transition

Figure 2: Primitive variables are used to identify the internal-external dicing edges during the chopping step. In the ensemble shader the primitive variable is used to soften the transition between the external and internal shading.

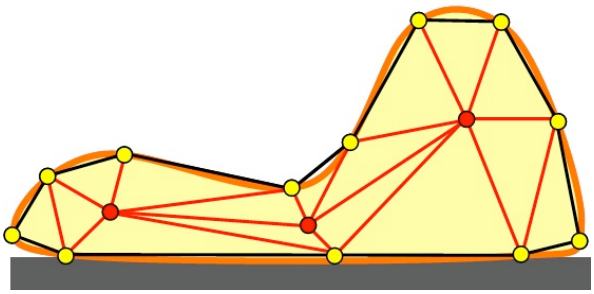
Another more shot specific example of animation interaction was one of the chefs rolling dough. The animation called for very specific shapes based on the force and direction of how the hand strikes the dough ball. The dynamics for the dough was created using a classic spring-mass system [TER87] composed of a mesh of surface springs augmented with additional internal springs to maintain volume. Trying to drive the shape of this mesh simply based on the input animation would have required numerous iterations with animation and tweaking spring parameters.

Instead the animator simply worked with a rigid shape used as a guide for position and size. Based on the animation we built key blend shapes. However, rather than doing simple shape blending we used these key shapes to come up with new rest configurations for the internal nodes at keyframed moments. The internal nodes in

turn would pull on the internal springs causing a dynamic change to the object's shape. In this way we were able to combine very specific control to match the animation and still allow for dynamic collisions and history one would expect in soft dough. This was also a good way to achieve plasticity with classic soft bodies in that they don't always return to the same rest shape.



(a) Default springs



(b) Posed springs

(c) Default rendered dough

(d) Posed rendered dough

Figure 3: (a) and (b) show the internal (in red) and external (in black) springs of the dough ball in default and posed positions respectively. (c) and (d) show the corresponding rendered dough from the shot.

There were some exceptions when we did have to change the animation to make the simulation work. These generally involved a container (ladle, glass, spoon) or a collision object (spatula) that had such a fundamental impact on the simulation in terms of shape or more frequently timing that it became necessary to change the animation in order to get the right simulation behavior. Even in these situations, to minimize iterations, we would first get the simulations working simply by re-animating the problematic object before asking animation to re-do the rest of the character animation.

3. Directed Simulations

Rat's Problem: The viscosity of the sauce is feeling really good but can it come out of the pot sooner? OR We want nice snappy animation as this pot fall to the floor. Can we do something about the fact that the soup inside wants to either be left behind or completely explode?

Underlying Challenge: Physically believable versus physically accurate simulation with the ability to iterate quickly and with the right creative knobs.

General Solutions: Cheat like crazy! This includes: splitting up the simulation into small manageable pieces; animating physical properties that would typically be fixed; augmenting with particles or other more controllable schemes and massaging the final shape with deformers.

The small body liquids for sauces, soups and wines were some of our hardest simulations. In one of the shots the chef pours out a thick marmalade sauce with raspberry bits suspended inside onto a plate. The sauce was simulated using the physBAM fluid libraries [BRI06]. A Maya interface was used to setup the initial sauce fluid; pot and plate collisions and simulation properties. After several wedge tests we settled on a viscosity that matched our reference. However, it took too long for the sauce to pour out in the duration of the shot. So we animated the gravity both in direction and intensity to force the sauce out more quickly. Figure 4(a) and 4(b) compare the directionality of the sauce when it's first pushed out and later in the pour once the gravity is a more normal downward vector.

(a) Initial Pour (b) Main Pour (c) Breakup

Figure 4: When the pour starts (a) the gravity is stronger and pushes the sauce sideways to force it out of the pot. (b) shows a more conventional downward vector for the main pour. (c) shows how the pour breaks up with augmented Maya particles towards the end.

Another problem we had with the initial simulation because of the high viscosity was that the sauce would snap at some point creating a sharp discontinuity in the pour. To bridge this gap we simulated some maya particles from the lip of the pot sauce to create some breaking shapes. Figure 4(c) shows one of the later frames in the pour where the input particles to the mesher were a combination of marker particles from physBAM and simple maya particles.

The meshing of fluid surfaces was done with p2mesh, which applied a moving least square implicit function [SHE04] on the input particles with extensions for preserving temporal coherence. Figure 5 shows a block diagram of our basic fluid pipeline. Note that in addition to the voxel based physBAM we also used a particle solver that was originally developed for Nemo called splasht [BAR92]. The choice of which simulator to use completely depended on the specific needs of a particular shot. From the lighting and rendering standpoint the simulator was immaterial since the mesh was generated in a consistent way.

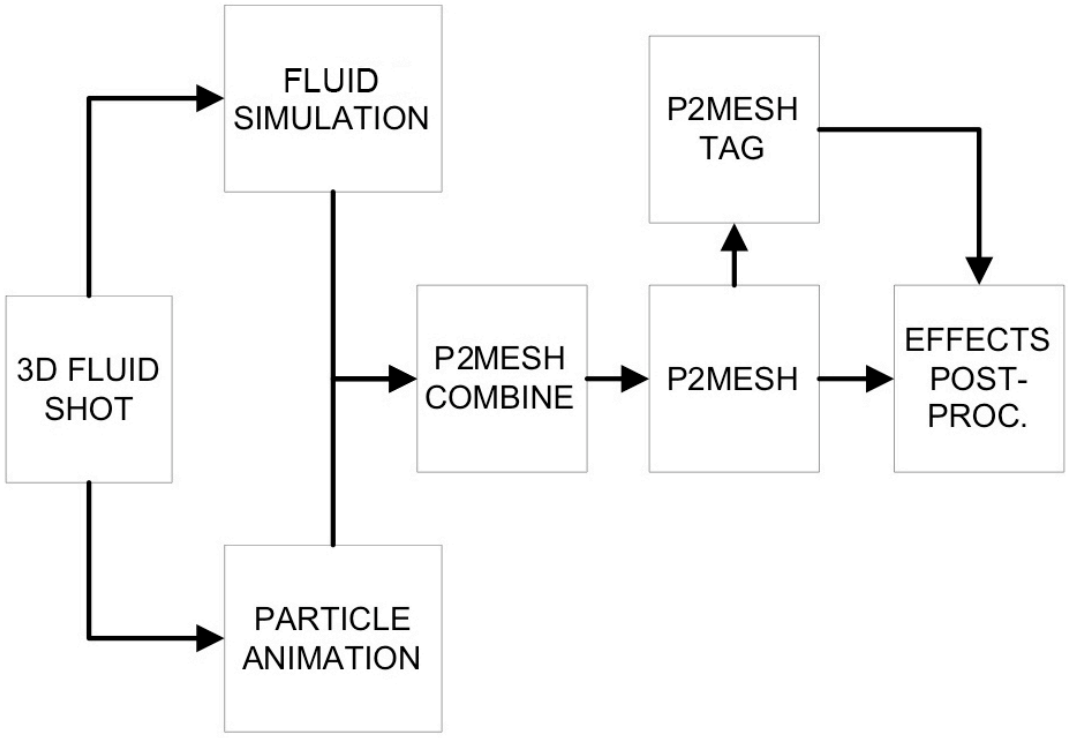


Figure 5: Fluid Pipeline

It is worth calling attention to a few other aspects of the fluid pipeline. As we mentioned before, we could add maya particles to augment the marker particles or fluid particles produced by the simulators. Since maya particle dynamics are quick to iterate, easy to control and mel scriptable it gave us a very direct way to bridge problem areas in the simulation. In some cases we would even cull the fluid particles out and replace them with particles from a different run of the simulation or with maya particles. In spite of this patchwork of inputs the resulting mesh would be smooth because p2mesh combined all the inputs before generating the implicit function.

Another production friendly aspect of the pipeline was the ability to deform the fluid mesh with lattices. For example if our sauce pour felt too thick due to resolution constraints we could thin it out with a lattice. Finally p2meshTag made it possible to attach properties to the input particles that were carried over into the fluid mesh as vertex varying data. This allowed us among other things to attach a parametric shading space that evolved with the fluid flow.

The raspberry bits inside the sauce were simulated in a second pass by advecting particles through the velocity of the fluid simulation and instancing geometry at each particle. Even though the solid bits should have some influence on the sauce fluid we made a conscious production decision to break this dependency thus allowing us to break the more complex and expensive fluid simulation into two simpler, more manageable passes.

Figure 6 shows frames from another shot where a chef knocks over a pot of soup that splatters all over the floor. We had a couple of interesting simulation challenges in this shot. The pot propelled down to the floor at such a high velocity that the soup inside was unable to physically keep up. It either got left behind or would become unstable due to the extremely high forces needed to keep it inside the pot. We overcame this problem by changing the object frame of the simulation relative to the world. In particular we removed all translations from the frame until the pot hits the ground but maintained the rotations of the pot's tilt. This made the simulation more

stable. We then tracked the soup mesh back to the flying pot position before it hits the floor.

(a) Soup Spill (b) Splattering on floor

Figure 6: The soup spill in (a) was done by fixing the pot's translation frame but allowing rotation. (b) shows the use of an accumulated displacement map for the thin splatter on the floor.

Another problem had to do with the resolution. We wanted the soup to spread out in a nice thin layer on the floor. Unfortunately, we had a fairly large space to cover and there just wasn't enough voxel resolution to make this happen. As a result the soup would break up into islands as the liquid thinned out on the floor. In order to get around this we rendered the soup on the floor with a projection camera. Displacement maps [COO87] were created by accumulating the rendered images over time. The soup on the floor was shaded using the displacement map rather than using the fluid mesh.

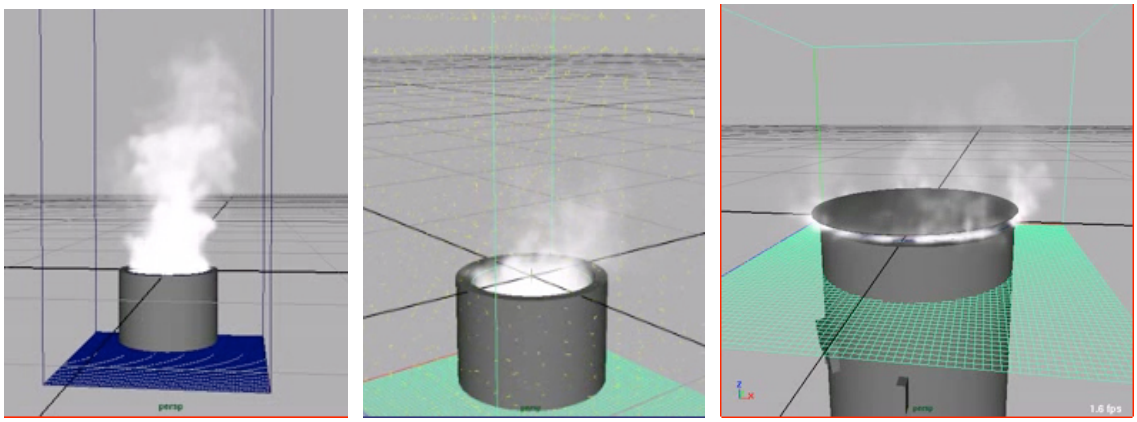
4. Active Dressing

Rat's Problem: The kitchen needs to feel busy. It would be nice to see some flames under the pots and pans and some steam wafting up from them. Oops! We are going to need steam in half the movie.

Underlying Challenge: Continuity Issues with environmental effects.

General Solution: Active dressing/Catalog based Effects. Put the extra effort into optimization.

Over half of Ratatouille takes place in a kitchen. To give the impression that the chefs are cooking and the kitchen is busy almost every single shot required steam to waft up from the pots and pans and flames underneath. At the same time, the set dressing department put a lot of time and energy into figuring out what dressing was needed at each station based on story needs, the time of day and the chefs at work. Also, it took a lot of diligence on their part to ensure dressing continuity across each sequence and the movie as a whole. It would have taken a lot of back and forth to figure out what type of steam and flames would be appropriate to the rest of the dressing and to maintain continuity as well.



(a) Boiling (b) Simmer (c) With lid

(d) Steam and flames dressed

Figure 7: (a), (b) and (c) show three different steam variants from the Vapor catalog. (d) shows an example of steams and flames dressed in one of the shots.

So during pre-production we made a decision to setup a catalog based system called the Vapor model. The set dressing artist could then pick the appropriate steams and flames from the catalog and dress it in with the rest of the kitchen. Size and rotation variation as well as random frame offsets were used to create added diversity on top of what was available in the catalog. Figure 7 shows some examples of catalog steam and the resulting dressed shot.

Whenever the characters interacted with the steam we simply replaced the catalog variant with a hero simulation. These simulations were done with Maya Fluids [STA99]. In these situations the catalog variants served as starting points, which meant less parameter tweaking to get a particular look for the steam.

5. Scene Integration

Rat's Problem: The batter is flying all over the place but the counter is spick and span and there is not a stain on the chef's apron.
Underlying Challenge: Environmental implications of an effect on the overall scene.
General Solution: Breakdowns should be about more than just spotting effects. Consider the fallout on other departments as well as sequence continuity issues.

The important thing is to identify integration issues early on. When spotting effects for a sequence its important to look at:

- The impact of that effect on the rest of the image in terms of cloth dynamics, groom for hair and fur, dressing, shading, lighting and animation.
- Continuity over time.

When the chefs were cooking in the kitchen sets and shading ensured that their stations were messy and their clothes stained. Even the best realized cooking effects would have seemed out of place if the environment was clean and the clothes didn't have wear on them.

Figure 8: Messy cooking station and stained clothes

In situations where integration issues slipped through the cracks, fixing things later proved to be expensive and frustrating for other departments. We found that there was a ripple tax on correcting the integration issues on a shot by shot basis.

6. Conclusion – making soup from the ingredients

Creating the cooking effects for Ratatouille posed some unique technical and process challenges that we have discussed. Ultimately, however, the success of effects in an animated film has to be judged by how well they are able to respond to story, performance and look needs.

A prime example of this in Ratatouille was the soup, which brings together a lot of the key concepts we have touched on. The soup goes through several transformations almost like distinct personalities that are shown in Figure 9. In order to have maximum flexibility we broke down the technical challenge of the soup into two obvious parts: simulation and shading.

(a) Burnt Soup (b) Broth in preparation

(c) Finished Cream Soup

Figure 9: Soup at various stages in the movie. The same shader was adapted for different looks.

The simulation for the soup ranged from full on, voxel based fluid dynamics, which was needed for the soup spill, to a simple height field for the finished cream soup simmering in a pot. In some cases, as with the broth, we used fluid dynamics even if the surface was relatively simple in order to get the velocity field needed to advect vegetables or spices being added to the soup.

The soup shader had different controls for color, depth tinting, floating oil and bubbles. By balancing these attributes it was possible to create a wide range of looks. For clearer soups the depth tinting was critical to get a sense of translucency and volume. For the thicker, creamier soups we used a combination of floating oil, shaped reflections and scatter to create the right visual texture. Also, note the shader dressing on the pots in Figure 9 to blend the soup surface with its container.

Another interesting example of story driven effects were the distinctive cooking styles for all the chefs ranging from precise to musical to mysterious to down right clumsy. In our opinion, the choice of letting animation go first with the performances, the flexibility of the chopping system and other tricks to make simulations more directable were all critical in getting the range of cooking effects needed in the film.

Acknowledgments

A very special thanks to the entire Rat *fx* team whose work is represented here. In particular Patrick Coleman and Eric Fromeling for the chopping system; Tolga Goktekin for the sauce pour and dough examples; Chen Shen and Gary Bruins for the soup spill example; Jon Reisch, Darwyn Peachey and Chen Shen for the very versatile fluid pipeline; Eric Froemling for the vapor model; Jason Johnston, Mach Kobayashi and Tom Nixon for the soup; Seth Holladay, Chris King, Trent Crow, Darwyn Peachey, Enrique Villa, Ben Anderson and Juan Buhler for stunning cooking effects through out the film; Kevin Gordon the fx manager and Nick Berry our coordinator.

References

BAR92 – David Baraff, Andrew Witkin, Dynamic simulation of non-penetrating flexible bodies, SIGGRAPH 1992

BRI06 – Robert Bridson, Ronald Fedkiw, Matthias Muller-Fischer, Fluid Simulation: Course Notes, SIGGRAPH 2006

COO87 – Robert Cook, Loren Carpenter, Edwin Catmull, The Reyes image rendering architecture, SIGGRAPH 1987

LAI86 – Laidlaw H. David, Trumbore W. Benjamine, Huges F. John, Constructive Solid Geometry for Polyhedral Objects, SIGGRAPH 1986

PIX07 – Pixar Marketing Doc, Slim – A Complete Shader Management System, https://renderman.pixar.com/products/tools/slim.html, 2007

SHE04 – Chen Shen, James F. O'Brien, Jonathan R. Shewchuk, Interpolating and approximating implicit surfaces from polygon soup, SIGGRAPH 2004

SMI06 – Russell Smith, Open Dynamics Engine Users Guide, www.ode.org, 2006

STA99 – Jos Stam, Stable Fluids, SIGGRAPH 1999

TER87 – Demetri Terzopoulos, John Platt, Alan Barr, Kurt Fleisher, Elastically Deformable Models, SIGGRAPH 1987