THE MORPHOLOGICAL DEVELOPMENT OF A WOOD BURL SHADER

A Thesis

by

ROBERT SIMMS MOYER

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

December 2003

Major Subject: Visualization Sciences

THE MORPHOLOGICAL DEVELOPMENT OF A WOOD BURL SHADER

A Thesis

by

ROBERT SIMMS MOYER

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Approved as to style and content by:

Donald House (Chair of Committee)	Rodney Hill (Member)
Zhiyong Cai (Member)	Phillip Tabb (Head of Department)

December 2003

Major Subject: Visualization Sciences

ABSTRACT

The Morphological Development of a Wood Burl Shader. (December 2003)

Robert Simms Moyer, B.A., University of Maryland

Chair of Advisory Committee: Dr. Donald House

In the field of computer graphics, shaders provide an interface between lights and surfaces, giving the appearance of metal, plastic, wood, etc. As the field progresses, more and more shaders are required to simulate a wider and wider variety of materials. We present a new shader for the simulation of wood burl, a complex material used in furniture, art, car interiors, and a host of other luxury items. This shader was developed through a morphological approach – a study of the original material, its structure, and growth. Consequently, research began with a thorough look at wood burl, polished and unpolished, in an assortment of different species. We discovered the appearance can be broken into three sub-appearances – knots, curl, and a subtle undergrain. These three sub-appearances interact to create the characteristic swirls and whorls of burl. For the subtle undergrain, we used a common oak shader, added noise, and faded it into the background. We then developed a system of randomly placing points through the material to act as knots. Since the knots grow and distort the surrounding grain, we used distance-scaled forces to push the surface coordinates around and between all the knots. When the oak shader is applied, it appears to swirl and curl around the knots, much like a stream between rocks. This created the first level of curl or swirly grained wood, but one level alone appeared flat. To solve this, we procedurally blended levels of curl to give a look of increased depth. Finally, we added reflection, gloss, and other surface properties to give a look of warmth and polish. All of these properties are controlled by a set of parameters in the shader's interface. By adjusting these parameters, the user can emulate a variety of different burl types.

Dedicated to my father,

Robert Simms Moyer

"Life is far too important a thing ever to talk seriously about."
Oscar Wilde

ACKNOWLEDGMENTS

I would like to thank my thesis committee, including Dr. Donald House, Prof. Rodney Hill, and Dr. Zhiyong Cai. Their guidance was indispensable to this project. I would also like to thank the rest of the Visualization Laboratory faculty and staff for their instruction and aid.

I would like to thank my friends and fellow students in the Visualization Sciences Program. Matthew Roach, Gavin McMillan, Brian and Charu Clark, Karthik Swaminathan, William Telford, and Stephen Parker for their suggestions, critiques, and humor during our time at Texas A&M.

I would like to finally thank my mother, Laureen H. Moyer, my sister, Teresa Moyer, and my grandmother, Mary Moyer. They provided a welcome home to grow from and occasionally retreat back to. Most of all, I would like to thank my wonderful girlfriend, Krista Borman. Her support, love, and sporadic, well-meant kicks in the rear produced this thesis more than anything else.

TABLE OF CONTENTS

Page

LIST OF FIGURES

FIGURE Page

FIGURE Page

INTRODUCTION

The creation of the final image is an important issue for the science and industry of computer graphics. We render that image from a scene description, combining object surfaces, lights, and materials. The object surface is the model itself, the computer equivalent of a sculpture. Lights illuminate the scene, offering time of day, mood, weather, and other scenery information. The scene is still not complete, though, since all of our illuminated object surfaces are still flat gray. The real world has a nearly infinite number of materials (metal, ceramic, cloth, wood, glass, concrete, etc) and an even greater number of variations on each one (rusty iron, new copper, galvanized aluminum, hard-water stained steel, dented tin, etc).

Shaders are used to approximate the surface properties of real world materials. They distinguish a metal pot from a clay one and wooden table from a glass one. They tell the story of an object's surface without any words. What is it made of? How old is it? What kind of environment does it exist in? Is it treated gently or hard? All of these questions are answered simply with a shader. A shader is any program that tells a surface its color, opacity, specularity, roughness, and any number of other attributes. They can be based off images (texture maps) or from programmed procedures.

Three-dimensional procedural shaders, i.e. 3D functions, are commonly used to give a surface the look of a carved object. By assigning color values through an entire 3-dimensional space, rather than just along the surface, a 3D procedural shader can ensure continuity through corners, holes, and protrusions. These functions have long been used to imitate materials such as wood and marble. However, as the industry matures and its audience grows more experienced, we require more and more complex surfaces to achieve artistic vision and scene realism. A simple wood shader is no longer adequate; we need shaders that cover the full range of wood surfaces, from basic oak to highly

---

This thesis follows the style and format of *IEEE Transactions on Visualization and Computer Graphics*.

figured woods like wood burl. With that in mind, the goal of this research is to develop a procedural shader for wood burl.

Relatively rare, wood burl is used for a host of luxury and high-dollar items, such as fine furniture, turned vessels, and expensive automotive trim. It is a highly complicated surface, filled with twisted grain and subtle color shifts. The hypothesis underlying this thesis is that a structurally or morphologically based approach to creating a wood burl shader will help capture these unique properties.

A morphologically based approach assists shader design by breaking the problem into easily identifiable sub-structures. Upon investigation of real wood burl, we noted its look comes from a combination of three structures. First is a subtle under-grain that runs through the entire surface. Seen in figure 1, this grain largely appears in the specularity, faded behind the other two sub-structures. Second, there is a more prominent "curly" or swirly grain as seen in figure 2. Finally, there are the clusters of knots or buds shown in figure 3. None of these sub-structures exist independently – knots cause additional curl, curl obscures under-grain, under-grain eddies around knots. By understanding these substructures and their contribution to the appearance of wood burl, we designed a shader that approximates the same process.

In nature, wood burl is essentially a tumor on the tree; a piece of tree that has "forgotten" how to grow properly. This explosive growth process begins with scattered knots or buds that develop within the wood. The buds grow and push against each other, creating grains that often resemble an eddied stream. With this in mind, the shader package begins with a set of knot locations. Much like burl growing from buds, these knots create a set of forces that push the surface coordinates outwards. When a basic wood shader is run through the manipulated coordinates, curl develops. Finally, more basic wood grain is blended in to simulate the subtle under-grain sub-structure. The interaction of these sub-structures approximates wood burl's overall appearance.

Finally, the shader includes a variety of noise functions, color options, and surface properties to imitate specific types of wood and finishes.

Figure 1: The subtle undergrain sub-structure.

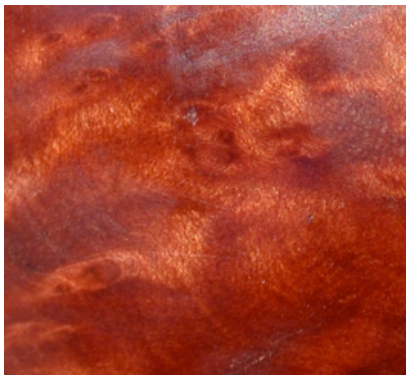


Figure 2: The curl sub-structure.

Figure 3: The knot or bud sub-structure.

BACKGROUND

The term “highly figured” encompasses wood such as curl, burl, spalt, knot, bird's eye, and many, many others. Fancy names aside, highly figured wood is any wood that goes beyond the classic appearance of wood, i.e. grain, texture, ring color contrast. The “figure” in highly figured wood refers to additional patterns that appear on the radial and tangential faces of a board. Since figure appears as a quality to the shine or color, it often "has to do with the light-reflecting properties of the wood." Figure may appear in colorful swirls, delicate ripples, or stormy patches. The end appearance is endlessly varied, even within a single tree. [18]

With this in mind, the choice of “where to carve” becomes vitally important. The end purpose of the burl decides how the figure is sliced. Much like a diamond cutter, the woodcutter working with highly figured wood must take time to understand what is inside the raw wood. [6] [11] The elegance and variety is what makes the material so attractive to carvers of expensive furniture, turned vessels, and other fine workings. While raw figured wood is attractive in its own right, it does not really jump out until finished. Thus, finishing becomes a necessary aspect of the “look” of figured wood. [7]

For this thesis, we specifically investigated wood burl, an irregular growth or tumor on a tree. Almost any kind of trauma can cause burls -- fire, frost, physical impact, even fungal or bacterial irritation. This stress causes riotous cell division, creating irregular pockets of included bark, heartwood, or sapwood within the burl. [5] Burl patterns fall into three main classifications: annual layering, end-grain budding, and a combination of the two. Annual layering grows much the same way as the rest of the tree, only faster and more erratically. This causes the swirl and curl distinctive of highly figured wood. End-grain budding patterns develop "through an explosion of early bud development that never quite makes it through the bark." This causes the knotted patterns characteristic of classic burl. The most highly sought after patterns include a

combination of both. [12] However, the very complexity that makes burl appealing also makes it difficult to imitate and add to a computer graphics scene.

One relatively simple method of applying wood burl or other material appearances to a virtual surface is called texture mapping. [3] Texture mapping essentially takes a painting and wraps it around a 3-dimensional mesh. However, the wrapping phase quickly proves difficult as meshes increase in complexity. Thus, surface parameterization is a fundamental issue with texture-mapping. The geometry must be carefully assigned coordinates that correspond with the image. Unlike the original painting process, this assignment is time-consuming and complicated. Moreover, once the parameterization is done, it will only work for that specific object. Each new object requires a different parameterization – and in many cases, a whole new texture image. Finally, a simple texture map will not remain 3-dimensionally consistent, losing the appearance of a carved object – making it inadequate for materials like wood and marble.

Procedural shaders, programs that determine the coloration and material properties of a surface, conquer these problems for many cases. Shaders act as an interface between the surface meshes and lights, detailing how the object should render. When the scene camera takes a picture of the scene, a certain sequence must occur to ensure our image renders correctly. This is called the rendering pipeline – and shaders fill a vital role in its completion. First, the scene geometry is determined by calculating visibility of all the objects. Each object has a shader with a bi-directional reflectance distribution function (BRDF). This function describes how lights interact with the object, factoring in lights we have in the scene, and returning a final color and opacity for that surface point. Shaders can also change the geometry itself through displacement, by moving the location of the actual surface points. [1] Practically speaking, they exist as set of functions approximating real-world physics – in other words, a simulation.

This simulation approach can also extend to other properties of the wood burl shader. Wood burl is a natural material, and follows a certain growth process or morphology. In nature, one often finds such materials exhibit a "form of design," or regularity and order. This order or structure develops naturally from a set of functions or laws that govern its behavior. [9] Pusinkiewicz, Lindenmeyer and Hanan similarly propose that to simulate any specific form, we must first analyze and describe the processes that led to the form. These processes also offer a natural mechanism for simulation control. [17][16] The more knowledge the simulator has about the plant and its environment, the more accurate the result will be. Similarly, the closer we can model our shader to real world burl morphology, the closer the end appearance to real world burl.

Many shaders use this model or simulation concept to improve accuracy. Rather than simply painting a texture map, equations can grow a texture directly onto the surface. By imitating these growth functions, we can imitate the final appearance. For example, while not strictly natural, a stone wall grows according to specific rules. Miyata creates random block cells and bricks row-by-row, following the general metaphor of real masonry. Once the cells are established, he fills in each with a stone texture. [13] By imitating the morphology of a real wall, Miyata improves the accuracy of his final result.

Others take this simulation approach to a cellular level, using low-level simulation to create high-level patterns. Reaction diffusion is a set of natural processes wherein cells exchange and adapt chemicals and information. Using a mathematical approximation of how pigments diffuse and react through cells, we can control the growth and appearance of a texture across an arbitrary surface. Since this is how cellular growth actually occurs, a wide variety of stripes, spots, and scales can be achieved. This provides similar surface independence as 3-dimensional texturing, but with a 2-dimensional non-carved look. [8] Similarly, one could cover a surface with a cellular function, but control the growth with biologically-motivated cellular development simulation. This allows the

texture to respond even more closely to the geometry -- creating small scales in high detail areas and large scales in wide-open flat areas. [2] However, these methods, while producing excellent results, also tend to be slow.

If speed is our concern, we can move to the other end of the spectrum – procedural basis functions. Basis functions render extremely fast because they do not simulate on the cellular level. Instead, they follow the principle that a set of spots or noisy patches essentially looks like any other set of spots or noisy patches. Consequently, it is unnecessary to simulate perfectly if the overall appearance is correct. For example, Perlin introduces the concept of a solid texture (what is often called a 3d texture), a function whose domain is 3-dimensional. [15] Solid texturing, or texturing surface points by their 3-d space coordinates rather than surface coordinates, allows for easy shading of "carved" objects. As the surface cuts through 3-D space, different points on the surface will access different parts in the function. [14] Perlin introduces this concept with noise and turbulence functions, algorithms that simulate nature's irregularities. They result in a natural appearance, without each surface point interacting with every other surface point in a complicated simulation.

However, while very versatile, Perlin's noise functions do not extend to cellular looks such as scales, slate floors, and leather. Worley provides a basis function for cellular solid texturing that is fast and flexible. Based on a Voronoi-style tessellation, it creates a jittered set of points that can be used to draw a set of scale-like cells. Like Perlin noise, it renders extremely fast and is 3-dimensionally consistent. However, also like Perlin noise, its speed comes from sacrificing "total surface awareness"…i.e., each shading point knows nothing about the surface properties of the others. [20] This eliminates many of the benefits of simulation, such as control and flexibility.

These disparate approaches are not completely exclusive, however. When used with a shade-tree type modular language, these functions easily become part of a set of blocks

to build larger, more complicated shaders. Cook and Perlin proposed shading-tree languages that are modular and flexible. Rather than developing an entirely new shading model for each surface, more general nodes and functions can be combined to achieve appearances. By combining these nodes and functions, we can assemble our rules for morphological simulation. [4,15] Hanrahan and Lawson formalize this with their introduction of SL, a shading language subset of C. By focusing the C language towards graphics, they give a robust, flexible implementation of Cook's shading tree concept. [4] Using SL to blend the accuracy of simulation and the speed of basis functions, it is possible to approximate the complexity of wood burl.

METHODOLOGY

Due to the dual scientific and artistic nature of computer graphics, the research methodology did not easily fall into one category. Since we tried to imitate a real world surface, our overall goal was a simulation. However, the evaluation process only asked the question, "does this look like wood burl?" This was, by nature, a qualitative process – a visual and subjective analysis. Consequently, we worked with aspects of both simulation and qualitative research techniques.

First and most importantly, we observed our source or reference material. Then we approximated our observations with programmed shader functions, adapted existing ones, or stacked different functions together. Figure 4 arranges our research methods as a flowchart. Note the loop back to step one labeled "compare to reference material." With each change or addition, we rendered a demonstration image of our shader on an example surface. Through personal and outside observation, we compared this simulated surface with our reference material. Depending on results, we went back and programmed / adapted / stacked again, until we achieved the desired appearance. This iterative process was necessary to steer closer and closer to an imitation of our reference material.

A critical first step in shader development is seeing the subject material. This goes beyond simply noting whether the material is brown or if it's shiny. It is important to understand how the material works -- its uses, its background, its story. For example, with wood burl, we investigated its growing processes and its applications. Is it a normal form of wood? No, it comes from a trauma or tumor on the tree. Is it a strong structural material? No, it is used almost exclusively in luxury or art pieces. Is it left raw? No, it is cut and polished to emphasize its beauty, much like a gemstone. We sought to capture the nature of wood burl, its qualities that set it apart from other materials.

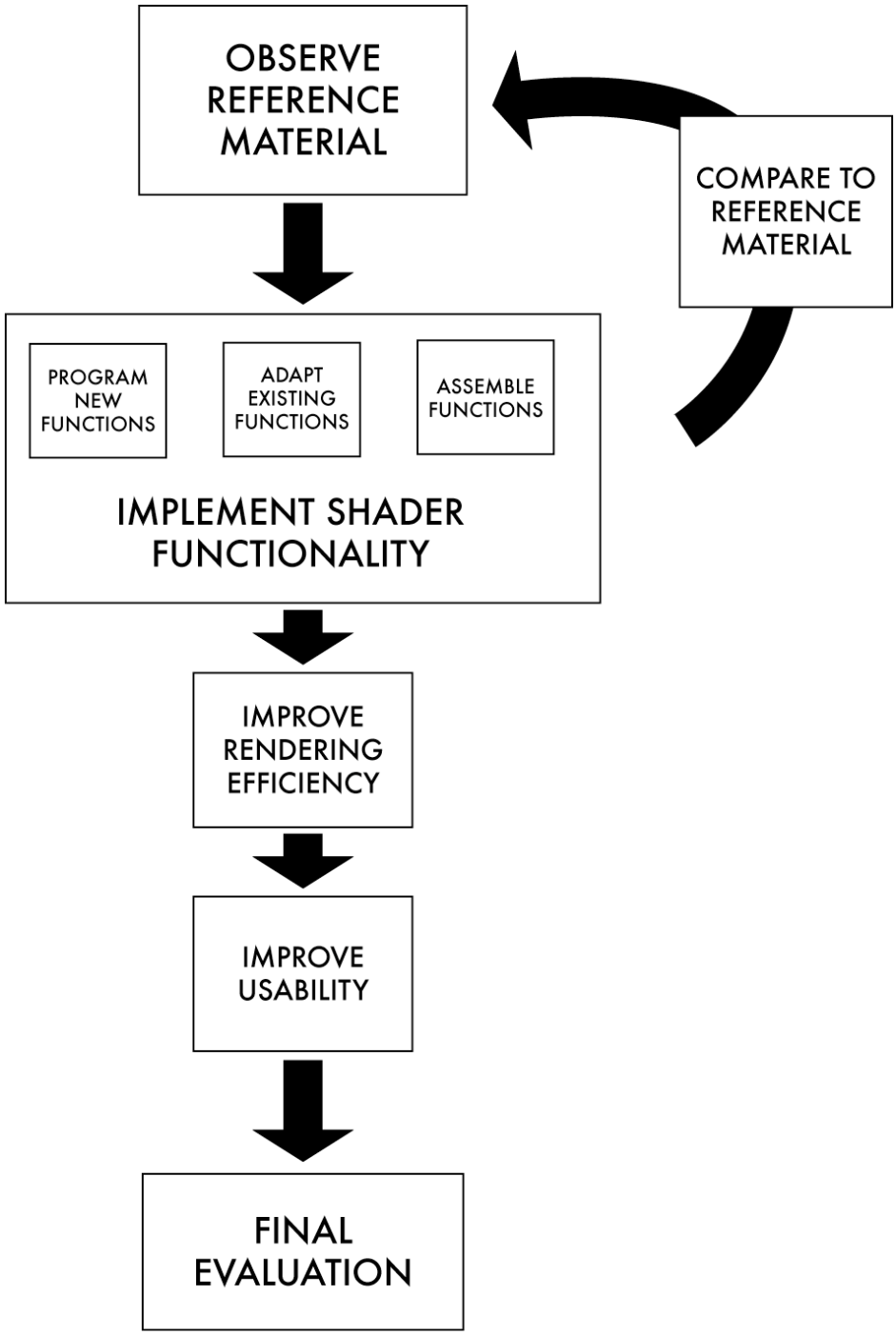


Figure 4: A flowchart diagram of the shader development pipeline.

Our first task was to collect a great deal of reference material. Though our final shader emphasized the polished appearance, raw wood burl proved excellent for noting subtle details in the morphology. It was often helpful to understand the original texture of the wood, as this plays a part in how the material ages, breaks, or otherwise interacts with its environment. We also chose a selection of finished items, to better understand the intended final look. We took a collection of detailed photographs as well; to help note small details the eye did not easily catch. A survey of related literature was also vital to understanding the morphological processes behind its appearance. We wanted to know how burl grows, how burl is harvested, how burl is worked, and even how burl is appreciated.

From our reference material, we discovered that knots, curl, and undergrain grow together to create wood burl's final appearance. As discussed earlier, many natural forms can be described as a set of rules -- simulate these rules and one may simulate the form. We used our sub-structures as a system of simulation rules, since they define how the wood burl grows. These sub-structures also gave us a logical plan of attack, allowing us to isolate our variables and break the complex problem into manageable pieces. Moreover, each of these sub-structure solutions could be used in a future shader, promoting code-reuse and maintainability.

At this point, we began to develop the shader. We started with the oak shader presented in Gritz and Apodaca's Advanced Renderman [1]. By pushing the shader's settings and appearance, we could see where it excelled and where it came up short. As seen in figure 5, it proved to be very adept at creating a basic curled grain, sufficient for the under-grain sub-structure. When pushing it farther to imitate knots and curl, the grain seemed random compared to the reference material. In the reference material, the knots seem almost like stones in a stream as the curl's grain and rings swirl and pool around them. The stock oak shader, on the other hand, had none of that evident morphology.

The shader also appeared very flat, having none of the signature depth or complexity of wood burl.

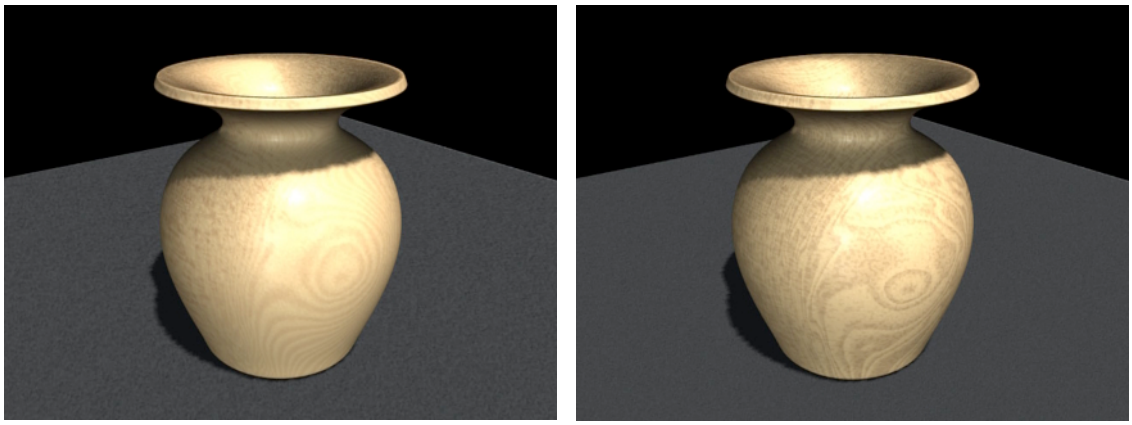
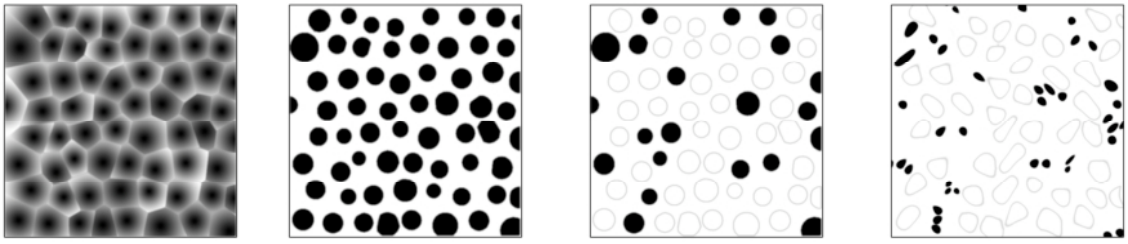


Figure 5: Different settings using the basic oak shader.

We decided a different method would be necessary for the other two sub-structures, knots and curl. From observation of the reference material, we determined that the knots were the seeds of the curl effects. It was not enough to just have a random swirling pattern; to look correct, the curl needed to appear caused by another process. Consequently, of the two, knots were the first effect to tackle. Since the knots were essentially just spots, we first tried a cellular basis function as described in [21]. This used an array of regular cells, whose center points are jittered to produce a random effect. By coloring according to distance from the closest point, we get an organic appearance, much like plant cells. While this function can be purely three-dimensional, we used a two-dimensional version accessed three-dimensionally to get streaks and long knots traveling across the surface. This method turns extrudes the circles of the two-dimensional knots into three-dimensional cylinders. Figure 6a shows a cross-section of such a function.



a. b. c. d.

Figure 6: Knots created with a Worley cellular basis function.

Next, we created smooth spots nearly ideal for knots by thresholding the cell distances. (figure 6b) However, we did not want the entire surface covered, so we randomly turned the visibility off for certain cells. (figure 6c) Since the knots tend to appear in clusters, we created sub-cells within each cell, thus building groups of small cells. Finally, to give the knots a less uniform appearance, we added noise to the surface coordinates. (figure 6d) This method of knot production created a believable spread of small knot-like spots. Figure 7 shows vases with different numbers of cellular-based knots. In these images, one can see how knots create streaks along the model's sides. These streaks are an accurate feature of the carving process.

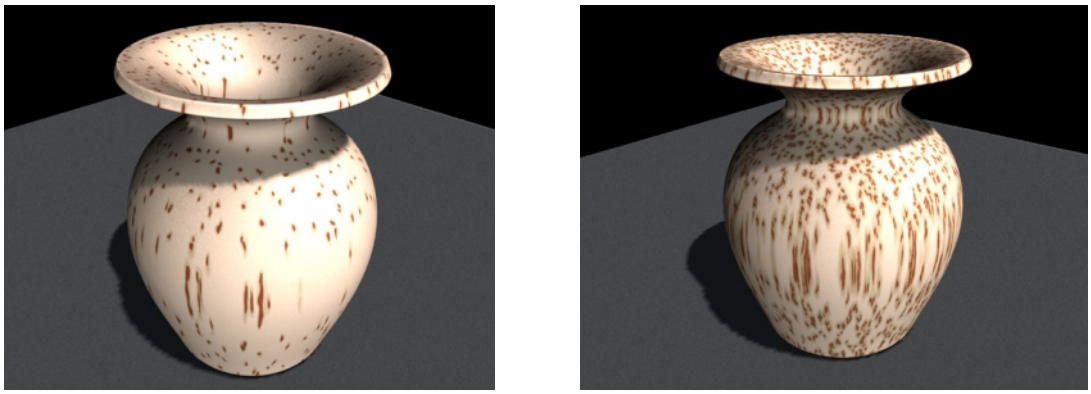
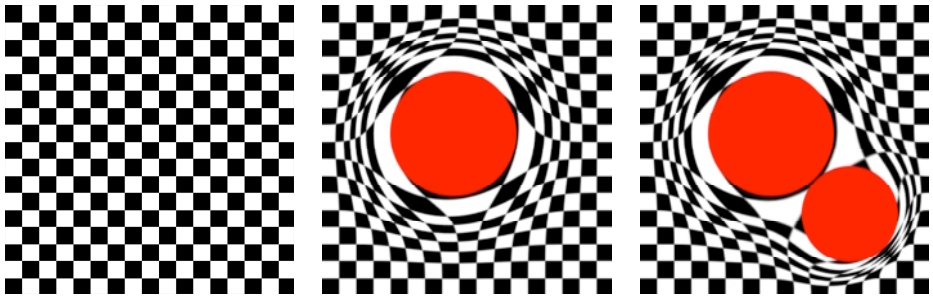


Figure 7: Shader demonstrating different amounts of cellular-based knots.

At second glance, though, the cellular basis function seemed to cause more problems than it fixed. It made great little clusters of knots, but controlling the rest of the grain proved difficult. First, the basis function's speed and efficiency comes from only knowing about the nine closest cells to the current surface point. However, for the grain to be pushed around correctly, it needed to know the entire knot pattern. Second, since the "empty" or invisible cells were in fact still there, it became difficult to exclude them from the "closest knot" algorithm. Third, we wanted to use the vectors to the knots and their distances as forces to push around the surface coordinates of the wood grain, thus giving a swirly, eddying effect.



a. b. c.

Figure 8: Demonstration of knot-pushing algorithm.

The first and second problems were fixed by simply dropping the cellular basis function and using a large array of randomized points instead. While computationally expensive (all the knots had to be calculated for every surface point), it insured that the shader knew about all the knots for every surface point. This allowed us to implement the morphological effect of the knots affecting the grain. Figure 8a-c show this idea. Starting with a basic checkerboard (figure 8a), we add one knot. To get the "pushing" effect, we wanted every surface point to sample from another location – a point getting pushed outward from the knot.

To this end, every surface point calculates its distance and direction to every knot, so that each knot acts as a point force on that point. The calculated distance then scales that force. The sum of these forces is the final influencing vector on the current surface coordinate. The surface point then uses that force vector to sample the wood function. Each "force" actually originates at the surface coordinate and ends at the pushed point. (figure 8b) Add lots of knots, and we begin to see the distorted grain of wood burl. (figure 8c)

In order to accomplish this, we used the Renderman Shading Language (RSL) to develop the surface coordinate pushing algorithm shown in figure 9. The function kpush is passed the current surface point (x,y,z), a pair of float arrays containing the knot coordinates, a level parameter (to be described later), and an output float location for the distance to the closest knot. The function loops through every knot to determine its effect on the current surface point. This effect is calculated as if each knot were a point (or cylinder in 3D) force. As all the forces are summed and applied to the original surface point, they create an eddying effect around each knot. This left largely garbage inside the knot, which could be covered with the actual knot spot coloration. Figure 10 demonstrates how this pulling method makes the normal wood texture appear squeezed between all the knots.

	Type	Definition
KNUM	constant float	the number of knots defined
Ps	point	current surface point
Level	float	a scalar to control different levels of burl
ks[]	float (array)	knot surface coordinate s values
kt[]	float (array)	knot surface coordinate t values
mindist	output float	distance to the closest knot
push	vector	sum of force directions pushing on surface point
curr	vector	direction from surface point to current knot
i	float	loop iterator
dist	float	distance from surface point to current knot

```
#define KNUM 50
point knot_push (point Ps; float ks[], kt[], level;
                 output float mindist;) {
    vector push = 0, curr = 0;
    float i, dist=0;

    //Loop for all knot locations
    for (i=0; i<KNUM; i+=1) {

        //Determine dist. and dir. To curr. knot from Ps.
        //Scale push force by distance and level.
        curr = point (ks[i], ycomp(Ps), kt[i]) - Ps;
        dist = length(curr);
        push += normalize(curr) * (1/(dist*(level-1)));

        //Store closest knot.
        if (dist<mindist) mindist = dist;

    }
    //Returned adjusted surface coordinate
    return (Ps + push);
}
```

Figure 9: The knot-pushing function. The wood curl function uses knot locations to push around surface points.

Figure 10: Shader demonstrating knot pushing function.

Of our three substructures -- undergrain, knots, and curling – we had accomplished the first two and had a solid start on the third. However, while the surface coordinate push function achieved basic curling, it still looked flat, with none of the depth associated with polished wood burl. By referring back to our original material, we discovered that wood burl combines several different grains, tricking the eye into believing it is seeing depth. As shown in figure 11, we initially tried mixing two wood functions together. While getting the balance was difficult, the grain did appear richer and deeper. Figure 10 also includes an environment-mapped reflection to give a better look of gloss.

Figure 11: Mix of two different grains, adding depth and complexity. The large black spots are knots that had their surface coordinates incorrectly scaled.

To increase depth and allow more user control, we added a system to procedurally mix

multiple levels of wood burl. (figure 13) Instead of hard coding in two or three levels of curl, the user can specify how many levels, and can add grains with decreasing amounts of curl. The level variable included in figure 13 performs this function. It controls the calculated distance's effect on the force, decreasing the force with increasing levels. Increasing or decreasing this level factor chooses how many levels of curl are mixed together. In figure 12, we see the procedural level system creating a sense of depth not evident in earlier renders. The green spots are the knots, shaded green to make clear where they stop and start. As shown in the results section, we later filled in the knots with coloration similar to the surrounding wood.

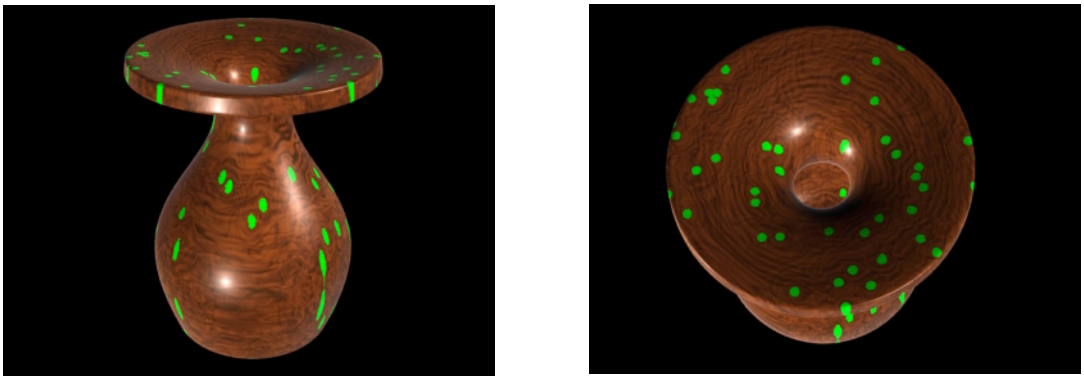


Figure 12: Level system for layering different amounts of curl. The green spots are just the knots, shaded green to stand out.

With the coloration largely established, we next worked on surface properties. We added the warmth, glow, and polish of burl to the shader. A very wide, wood-colored anisotropic specularity was given to the surface to improve warmth and glow. This warming of the surface increased contrast and helped pull out the depth in the burl. Since we wanted a glossy, high-polished surface, we began with the basic gloss shader in [1], but instead of just using a flat color, we multiplied by the light color and an environment map. This gave extra complexity to the gloss highlight, making it seem the result of a wide area light source. We finally had a three-dimensional shader with the appearance of highly polished wood burl.

Value	Type	Definition
PP	point	current surface point
Pcurl	float	the “pushed” surface point
knots	float (array)	knot surface coordinate s values
knott	float (array)	knot surface coordinate t values
min_dist	output float	distance to the closest knot
wood	color	sum of different wood levels
ringf	float	wood ring frequency
I	float	loop iterator
levels	float	number of mixing levels
Vnoise	float	additional vector noise
balance	float	mixing factor for wood levels

```
 1  Pcurl = knot_push(PP*0.25, 0, knots, knott, min_dist);
 2  wood = oaktexture ( /*parameters*/ );
 3
 4  //loop through levels
 5  float ringf = ringfreq;
 6  for (i=1; i<levels; i+=1) {
 7      //add noise to the surface point
 8      Vnoise = vfBm(0.4 * Pcurl,0.1,4,2,0.5);
 9      PP = Pshad + 0.5 * Vnoise;
10
11      //push surface point from knot locations
12      Pcurl = knot_push(PP*0.25, knots, knott, I, min_dist);
13
14      //mix previous and current levels together
15      wood = mix(oaktexture (Pcurl, ringf * 1.25,
16                  /*parameters*/) ,wood,balance);
17  }
```

Figure 13: The procedural mixing function. This code adds together different levels of wood.

RESULTS

Our results consist of a series of renders demonstrating the shader's capabilities. We can compare these renders with real wood burl to get a basic feel of our success. However, we do not expect a perfect match. Instead, we look for a general impression of wood burl that can be tweaked and improved.

For example, figures 14 and 15 offer a comparison of real wood redwood burl to shader burl. We notice that the general color and form appear correct, including many characteristics of the original. The imitation wood has depth and complexity from the different levels and grains mixing together. We see a similar blend in the reference material. Likewise, both our imitation and real burls have curl and wood that swirls around the clusters of knots. Our imitation also offers much of the warmth and gloss of the original. The glossy highlights are complex and seem to reflect an area light. This is an improvement over the standard gloss shader which leaves a solid flat highlight. The subtler wide, wood-colored anisotropic highlight also brightens the wood, making it interact more with the light.

Comparing figures 16 and 17, we also see the shader works well at imitating lighter burls. This imitates a maple burl or other unstained wood. While such wood might not have the warmth or glow of the redwood example, many other signature features become apparent. The knot-caused curling is very evident, as is the streaking along the side. This streaking comes from carving through the curl, since the knots work as cylinders.

Missing from our results are features like bark, holes, and other patterns so distorted that they do not obey usual ring and grain patterns. These currently are not included in the shader. These real-world features can especially be seen in the detail photographs of 14 and 16. Artifacts of the carving process, like the tiny lathe grooves on the sides that cause anisotropic highlights, are also missing. Similarly, age and environmental effects

like wear and tear are visible only on the reference. Both of these issues were considered outside the scope of this shader.

Also included in our results are a few more complex examples. Figure 18 shows the shader applied to a subdivision model of a hookah-smoking caterpillar. Note the complexity and variation in the grain as it travels along the character's arm and stomach. Since the shading function is three-dimensional, the character appears carved from one solid chunk of burl. By comparison, achieving the same effect through image-based two-dimensional texturing would be very difficult. Moreover, the same shader can easily be applied to a series of models, regardless of surface complexity.

Often wood carvers combine different types of burl to add contrast and emphasis. Figure 19 shows the burl shader used with different settings on different portions of the same model. The robot's hand has a light wood on the "skin" pieces, and a darker wood on the internals. This creates a pleasing blend of wood types, highlighting the different pieces in the model.

Perhaps most importantly from these examples, we gain a feel of the aesthetics of burl. This is a quality not easy to define, but comes from a holistic sense of what burl should look like. The results, while not a direct pixel-to-pixel simulation of our reference, impart the same sense of luxury and beauty as the original. It is this quality that is our most important goal.

Figure 14: A collection of photographs of a redwood burl vase. Note the warmth and gloss of the highly polished wood.

Figure 15: Renders of a computer-modeled vase with the wood burl shader. The knots swirl and push the wood grain around. The different levels of wood are also visible. The wood has much of the glow and gloss of the original. Render time: approx. 8 minutes on a Pentium 3 800Mhz.

Figure 16: A collection of photographs of a maple burl vase. Note how the spots at the top of the vase become long streaks on the sides. There are also curly lines of bark swirled into the burl.

Figure 17: More renders of a computer-modeled vase with the wood burl shader. The colors have been adjusted to approximate maple burl. Note how the spots become streaks and swirls along the side. However, the curly bark lines and rough edges are missing. Render time: approx. 8 minutes on a Pentium 3 800Mhz.

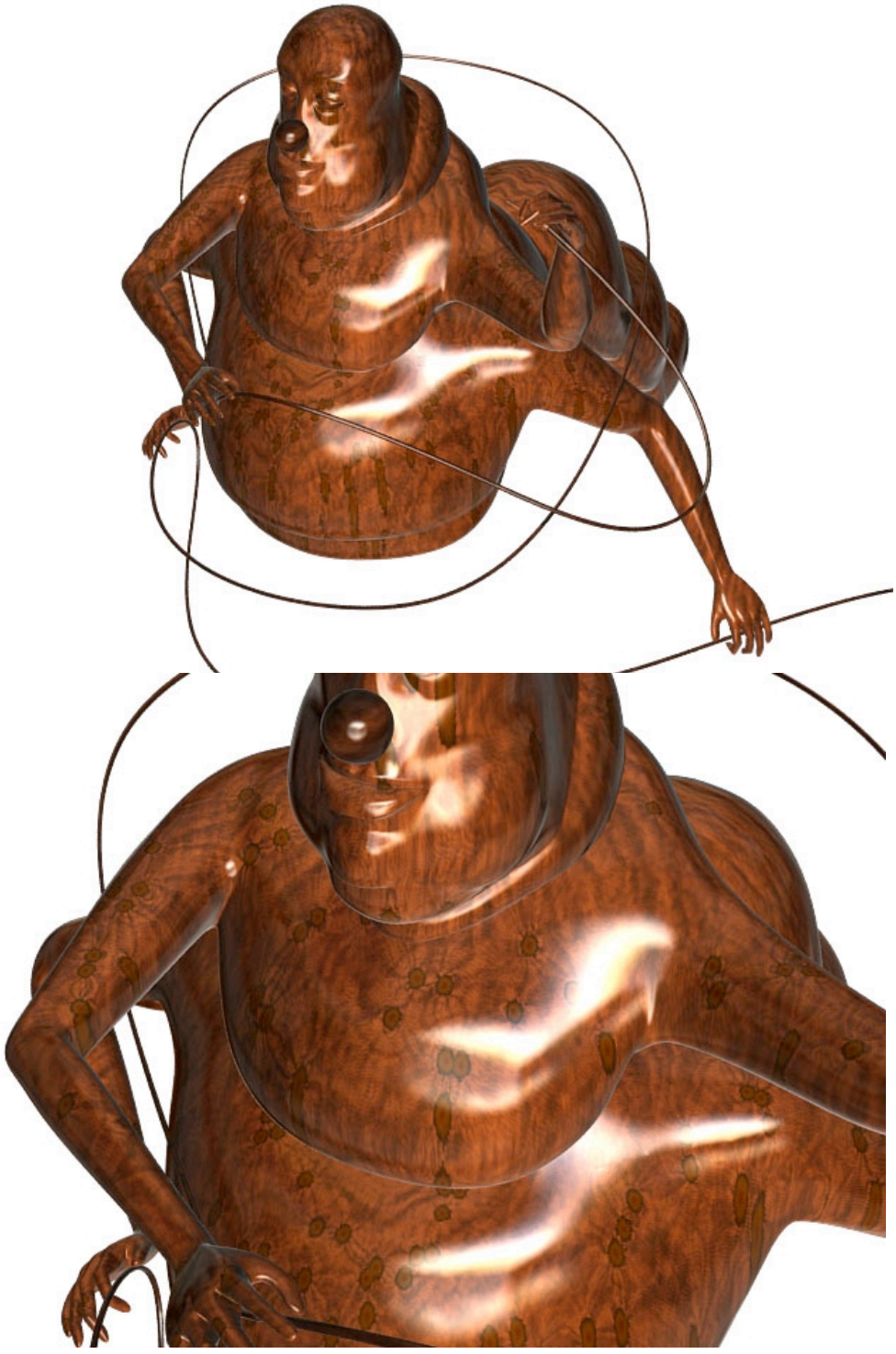


Figure 18: Shader as “redwood burl” applied to a more complex model. Render time: approx. 10 minutes on a Pentium 3 800Mhz.

Figure 19: Redwood and maple burl used together for a more complicated effect. Render time: approx. 10 minutes on a Pentium 3 800Mhz.

CONCLUSIONS

The shader was successful in its central goals. It gives computer graphics models an overall impression of wood burl -- matching color, form, and material properties. The shader also achieves the less concrete goal of aesthetics, including glow, warmth, and attractiveness. Finally, it is easy to use, simple to apply to any computer model – regardless of complexity. Although simple, it still provides the look of a carved material.

However, the shader has room for improvement. First and foremost, its render time is unacceptably high. For even simple models, render times were 8 minutes or more. This is due to the method of handling the knots and forces. Since all the knots and forces have to be re-calculated for each shading point, a great amount of render time is wasted. This same process takes control away from the shading artist, since they are unable to choose where precisely the knots appear. This severely limits artist interaction. Another issue is occasional aliasing during animation. While this can be fixed by adjusting the renderer's shading rate, this increases an already long render time.

In the future, we would like to tackle several of these issues. We believe both render time and artist interaction can be solved with the addition of a textural/procedural hybrid system. In this approach, an artist could draw an image of points where they would like knots to appear. A conversion program could turn that image into two 32-bit float images of "growth" forces – one for the X channel, one for the Y channel. During render, the shader would only need then to access these pre-calculated forces rather than calculating and recalculating for each surface point. This allows the artist to control how the burl forms, while also greatly reducing render time. We might also include a displacement system to imitate holes, sections, or edges left as rough wood. Integration with Pixar's SLIM shading network tool might also be helpful to future users.

By investigating the morphology of wood burl through both written documentation and visual reference material, we wrote a procedural shader that makes three-dimensional computer models appear carved from solid wood burl. The shader includes settings for coloration, material surface properties, and morphology (such as ring width, grain density, etc). Following the morphological process of natural burl, the wood's curl moves through a system of knot-caused "forces" simulating growth. We layer different levels of curl together to imitate the depth and complexity of burl. Finally, the shader blends in a subtle undergrain and gives the entire surface a warm, polished appearance. The resulting shader gives the overall impression of wood burl. We hope that this modularly based shader can be easily adapted for future research applications.

REFERENCES

1. A. Apodaca and L. Gritz, *Advanced Renderman: Creating CGI for Motion Pictures*, Academic Press, 2000.

2. A. Barr, B. Currin, K. Fleischer, and D. Laidlaw, "Cellular Texture Generation," *Proc. SIGGRAPH*, Vol. 29, No. 3, pp. 239-248, 1995.

3. E. Catmull, *A Subdivision Algorithm for Computer Display of Curved Surfaces*, Ph.D. thesis, Dept. of Computer Science, University of Utah, 1974.

4. R. Cook, "Shade Trees," *Proc. SIGGRAPH*, Vol. 18, No. 3, pp. 223-231,1984.

5. P. Hanrahan and J. Lawson, "A Language for Shading and Lighting Calculations," *Proc. SIGGRAPH*, Vol. 24, No. 4, pp. 289-298, 1990.

6. B. Hoadley, "Q&A: The Growth and Anatomy of a Burl," *Fine Woodworking*, No. 96, pp. 28-30, September/October 1992.

7. D. Holzapfel, "Q&A: Slicing a Burl," *Fine Woodworking*, No. 43, p.16, November/December 1983.

8. J. Jewitt, "Pop the Curl in Curly Maple," *Fine Woodworking*, No. 135, March/April 1999, pp. 38-40.

9. M. Kass and A. Witkin., "Reaction-Diffusion Textures," *Proc. SIGGRAPH*, Vol. 25, No. 4, pp. 299-308, 1991.

10. Y. Kawaguchi, "A Morphological Study of the Form of Nature," *Proc. SIGGRAPH*, Vol. 16, No. 3, pp. 223-232, 1982.

11. J. Lewis, "Algorithms for Solid Noise Synthesis," *Proc. SIGGRAPH*, Vol. 23, No. 3, pp. 263-270, 1989.

12. M. Lindquist, "Spalted Wood: Rare Jewels from Death and Decay," *Fine Woodworking*, No. 7, pp. 50-53, Summer 1977.

13. M. Lindquist, "Harvesting Burls: Strange Formations Are a Turners Delight," *Fine Woodworking*, No. 47, pp. 67-69, August 1984.

14. K. Miyata, "A Method of Generating Stone Wall Patterns," *Proc. SIGGRAPH*, Vol. 24, No. 3, pp. 387-394,1990.

15. D. Peachey, "Solid Texturing of Complex Surfaces," *Proc. SIGGRAPH*, Vol. 19, No. 3, pp. 279-286, 1985.

16. K. Perlin, “An Image Synthesizer,” *Proc. SIGGRAPH*, Vol. 19, No. 3, pp. 287-296, July 1985.

17. P. Prusinkiewicz, A. Lindenmayer, and J. Hanan, “Developmental Models of Herbaceous Plants for Computer Imagery Purposes,” *Proc. SIGGRAPH*, Vol. 22, No. 4, pp. 141-150, 1988.

18. P. Reffye, C. Edelin, J. Fran□on, M. Jaeger, and C. Puech., “Plant Models Faithful to Botanical Structure and Development,” *Proc. SIGGRAPH*, Vol. 22, No. 4, pp. 151-158, 1988.

19. P. Tischler, “Working Highly Figured Wood,” *Fine Woodworking*, No. 105, pp. 44-49, March/April 1994.

20. G. Turk, “Generating Textures on Arbitrary Surfaces Using Reaction-Diffusion,” *Proc. SIGGRAPH*, Vol. 25, No. 4, pp. 289-298, 1991.

21. S. Worley, “A Cellular Texture Basis Function,” *Proc. SIGGRAPH*, Vol. 30, No. 4, pp. 291-294, 1996.

VITA

Robert Simms Moyer

B.A. Computer Graphics, University of Maryland, May 2000
M.S. Visualization Sciences, Texas A&M University, December 2003

Rt 3 Box 711
Harpers Ferry, WV 25425