Modeling and 3D Printing Sea Shells
Final Report

Edward Ye || 100972832

2019/04/22

Abstract

Sea shells are an intricate natural object that have been studied and modeled mathematically and computationally. With the advent of consumer 3-D printing, some recent work has gone into create realistic printed versions. This project has attempted to extend techniques of modeling sea shell protrusions, through reaction-diffusion textures, and to ultimately use 3-D printers to fabricate the concrete from the abstract. The techniques involved in generating models and textures are outlined.

Contents

1 Some Background and Motivation

Sea shells have interesting patterns which appear to be readily described by mathematics and computation. Work has already been done to describe aspects of sea shells, from the spiral shape to the color patterns to the protrusions found on the exterior [3][4][8]. Recently some work has also gone into the 3D printing of sea shell model [2][1].

The motivation of this project will be to extend the methods of generating the exterior protrusions to be able to mimic a wider variety of shells. Prusinkiewicz and Fowler have modeled periodic ridge and bump patterns and they have also combined multiple generating curves to imitate more intricate shells [4]. Galbraith et al. have used constructive solid geometry (CSG) to compose different modules to generate a complete Murex Cabritii model. They have proposed the use of reaction-diffusion (RD) to place protrusions algorithmically [3]. Intuitively this appears to be a reasonable idea since the protrusion placement of certain shells are something like the placement of spots upon a leopard. Such patterns have already been described as textures using RD [7].

1.1 Main Objective

An interesting candidate for such a method would be:

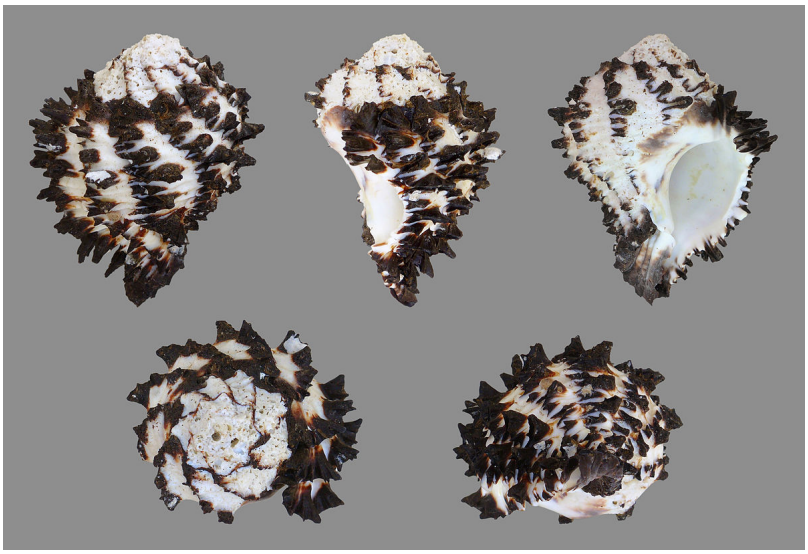

Figure 1: Hexaplex radix [11]

The main objective would be to create a model that closely resembles the shell and to 3D print it.

2 Implicit Surface

The initial idea was to use a raytracer, such as POV-Ray, to create an implicit surface (isosurface in POV-Ray terminology) that would represent the shell model. The software allows for textures to also act as functions, so that the sum of the implicit equation and the texture function would generate the desired surface. POV-Ray could then generate a point cloud that would form a mesh to be 3-D printed.

2.1 Logarithmic Spiral

Shells are modeled by logarithmic spirals. In polar coordinates the logarithmic spiral equation is:

$$r = Ae^{\alpha\theta} \tag{1}$$

Where A shifts the position of the spiral inward or outward, and α multiples the growth rate ($\frac{dr}{d\theta} = \alpha Ae^{\alpha\theta} = \alpha r$).

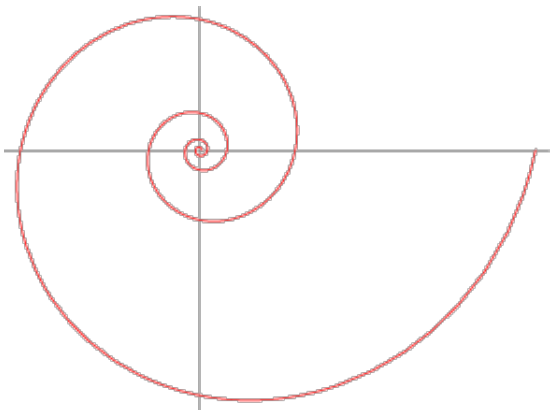

Figure 2: 2-D Logarithmic Spiral using equation (1). [9]

In 3-D Cartesian coordinates (1) can be represented as:

$$r = \sqrt{x^2 + y^2 + z^2}$$
$$\theta = tan^{-1}(\frac{y}{x})$$

$$\sqrt{x^2 + y^2 + z^2} = Ae^{\alpha \cdot tan^{-1}(\frac{y}{x})} \tag{2}$$

Squaring both sides of (2) then gives the implicit equation:

$$s(x, y, z) = x^2 + y^2 + z^2 - Ae^{2\alpha \cdot tan^{-1}(\frac{y}{x})} = 0 \tag{3}$$

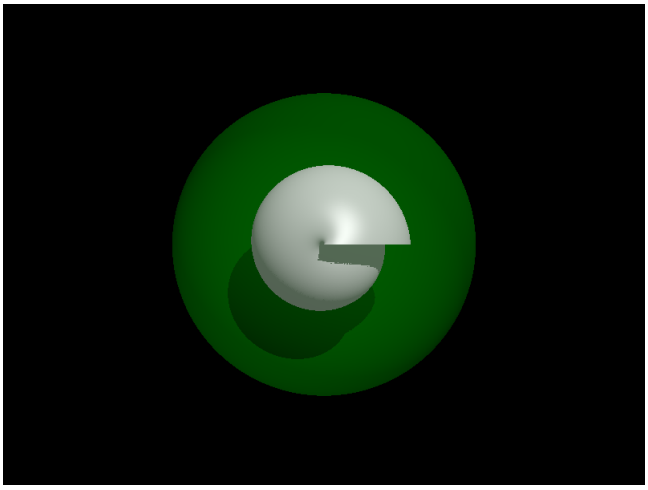

Figure 3: 3-D log spiral generated by *logspiral.pov* using an implementation of equation (3).

If I have some other function, such as $f_{noise}(x, y, z)$, I can add it to equation (3) to get another implicit equation:

$$s(x, y, z) + f_{noise}(x, y, z) = 0 \tag{4}$$

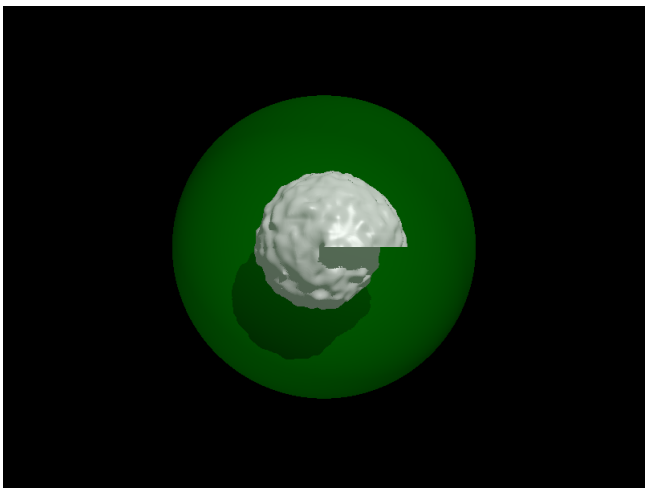

Figure 4: Equation (4) gives something pretty ugly.

After fiddling around with this for a while it became apparent that the existing parametric equations describing shells could not be easily turned into implicit equations.

3 Parametric Equations

Two kinds of parametric equations will be discussed: the first is modified torus used in [12], and the other is a logarithmic spiral with a generating curve used in [6].

3.1 Modified Torus

The parametric equation for a torus is:

$$\begin{cases} x = (c + a \cdot cos(v))cos(u) \\ y = (c + a \cdot cos(v))sin(u) \\ z = a \cdot sin(v) \end{cases} \tag{5}$$

for $u, v \in [0, 2\pi)$.

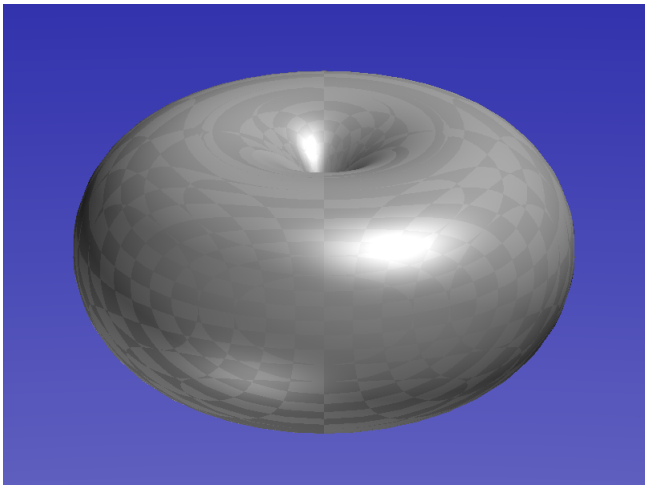

Figure 5: Raytraced torus using equation (5).

A simple way to create models resembling sea shells is to form a linear spiral that grows downward as described in [12]. The basic equation used is:

$$\begin{cases} w(u) = \frac{u}{2\pi} \\ x = w(u) \cdot [c + a \cdot cos(v)]cos(N \cdot u) \\ y = w(u) \cdot [c + a \cdot cos(v)]sin(N \cdot u) \\ z = w(u) \cdot a \cdot sin(v) + H \cdot [w(u)]^2 \end{cases} \tag{6}$$

Where H is the height and N is the number of turns of the spiral. $w(u)$ ensures that the spiral grows linearly from 0 to 1, and is used quadratically in the z parameter, so that the height doesn't grow too quickly downward.

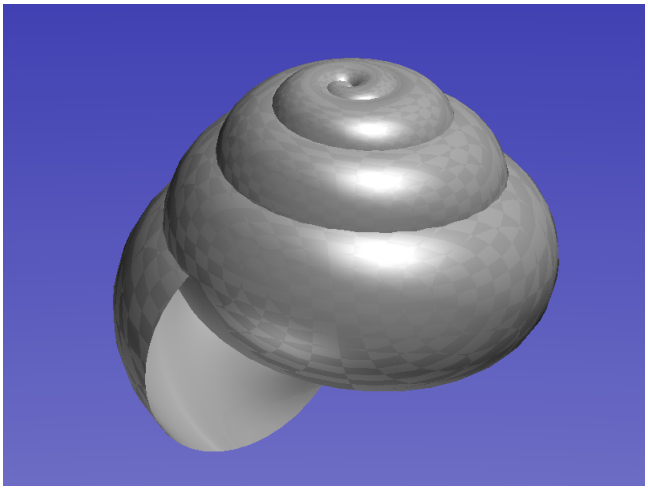

Figure 6: Periwinkle shell using equation 6, where $a = 1$, $c = 1$, $N = 4.6$ and $H = 2$. [12]

In blender using a XYZ Math Surface (parametric surface) and the modified torus equation, this model was created with the Solidify modifier:

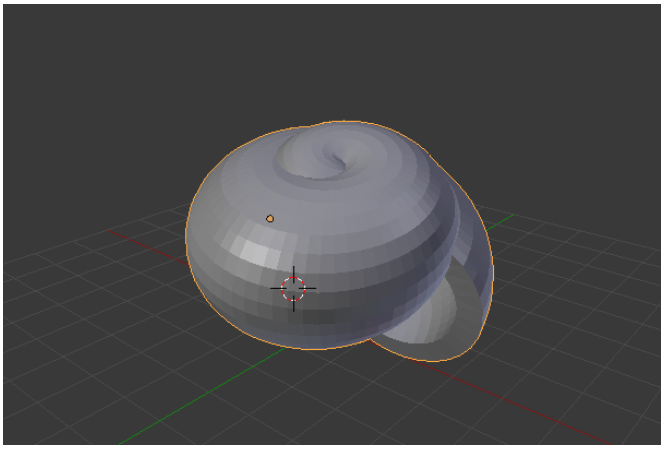

Figure 7: Solidified XYZ Math Surface in blender. Exported as *modified_ torus.stl*.

Then it was 3-D printed.

Figure 8: Printed from *modified_ torus.stl*.

3.2 Generating Curve

The method used in this project is described in detail here [6].

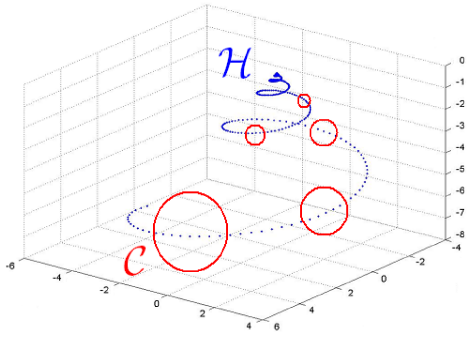

Figure 9: H represents the spiral, C the generating curve, which is an ellipse. [6]

H is a logarithmic spiral, which takes the polar form:

$$H(\theta) = Ae^{\theta \cot \alpha} \tag{7}$$

We get $H(u, v)$ by representing equation (7) in 3-D parametric form:

$$H(u,v) = \begin{cases} x = Asin(v)cos(u)e^{u \cot \alpha} \\ y = Asin(v)cos(u)e^{u \cot \alpha} \\ z = -Acos(u)e^{u \cot \alpha} \end{cases} \tag{8}$$

The generating curve $C(u, v, s)$ is an ellipse, with semimajor axis a, semiminor axis b, and angle s.

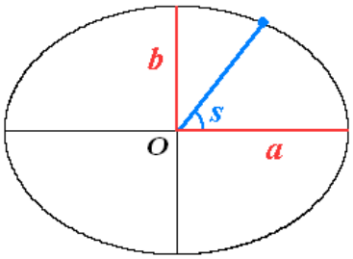

Figure 10: Elliptical generating curve. [6]

The parametric elliptical curve equation is:

$$r_c(s) = \frac{ab}{[(a \cdot sin(s))^2 + (b \cdot cos(s))^2]^{0.5}}, 0 \leq s \leq 2\pi \tag{9}$$

The width of the curve C is increase as it spirals along H. It is assumed that this rate of growth is the same as H. So, we multiply equation (9) by the growth rate, in polar form:

$$R_c(u, s) = r_c(s)e^{u \cot \alpha} \tag{10}$$

In Cartesian form:

$$C(u, s) = \begin{cases} x(u, s) = cos(s)cos(u) \cdot r_c(s) \cdot e^{u \cot \alpha} \\ y(u, s) = cos(s)sin(u) \cdot r_c(s) \cdot e^{u \cot \alpha} \\ z(u, s) = sin(s) \cdot r_c(s) \cdot e^{u \cot \alpha} \end{cases} \tag{11}$$

To get the shell equation we add equations (8) and (11) together:

$$\mathbf{S}(u, v, s) = H(u, v) + C(u, s) = \begin{cases} x = (Asin(v)cos(u) + cos(s)cos(u) \cdot r_c(s))e^{u \cot \alpha} \\ y = (Asin(v)sin(u) + cos(s)sin(u) \cdot r_c(s))e^{u \cot \alpha} \\ z = (-Acos(v) + sin(s) \cdot r_c(s))e^{u \cot \alpha} \end{cases} \tag{12}$$

Then we also add the 3 axes of rotation for C.

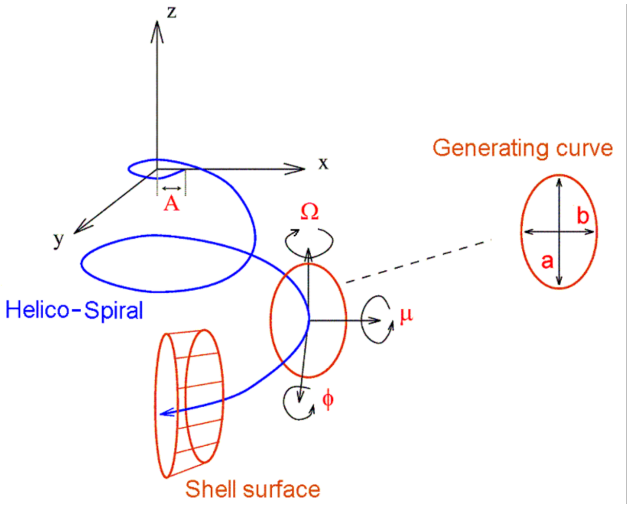

Figure 11: 3 axes of rotation. [6]

To get a large final equation:

$$\mathbf{S}'(u,v,s) = \begin{cases} x = D[Asin(v)cos(u) + cos(s+\phi)cos(u+\Omega)\cdot r_c(s) - sin(\mu)sin(s+\phi)sin(u+\Omega)]e^{u\cot\alpha} \\ y = [Asin(v)sin(u) + cos(s+\phi)sin(u+\Omega)\cdot r_c(s) + sin(\mu)sin(s+\phi)cos(u+\Omega)\cdot r_c(s)]e^{u\cot\alpha} \\ z = [-Acos(v) + cos(\mu)sin(s+\phi)\cdot r_c(s)]e^{u\cot\alpha} \end{cases} \tag{13}$$

D takes a value of 1 or -1, if the direction of coiling is either dextral or sinistral, respectively. The paper also presents some additional parameters for protrusions. Though they have been implemented in this project, they remained unused since the goal is to use RD textures for such purposes.

3.2.1 Processing

Some time was spent trying to get an existing implementation of this equation to work. An existing project used Processing and was abandoned in 2012. It required porting to a new version of Processing and, though an attempt was made to improve it, it remained buggy. It was useful for gaining an intuition of the effect of different variables, but an implementation in Blender could replace all its features.

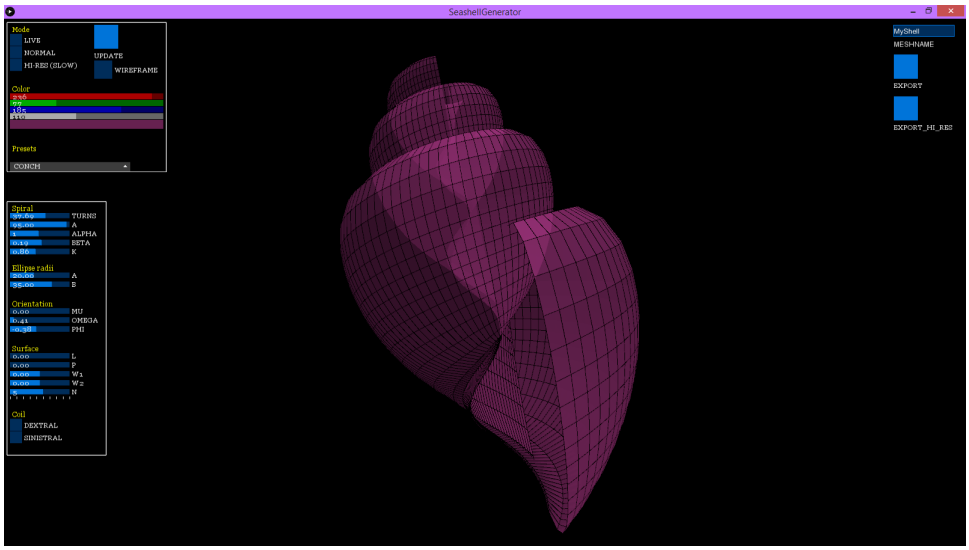

Figure 12: Processing based sea shell generator.

3.2.2 Blender

A python script, *blender_seashell.py*, was created to be used in Blender, which implemented equation (13). After fiddling with the parameters a shell was chosen that resembles hexaplex radix.

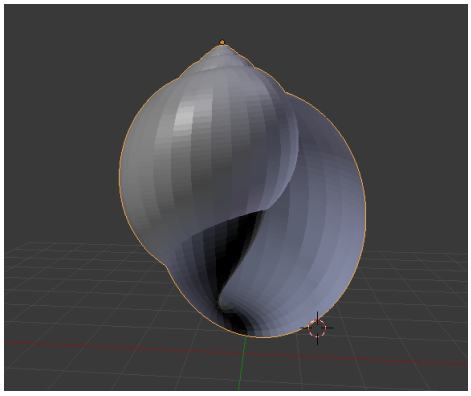

Figure 13: Generated by *blender_seashell.py*.

Then it was 3-D printed.

Figure 14: Printed from *radix_ shell_ only.stl*.

4 Reaction Diffusion Shaders

In order to model the protrusion of hexaplex radix an RD texture was created. Notice from Figure 15 that on hexaplex radix the pattern of the protrusions seems to originate from some kind of sinusoidal shape.

Figure 15: Sinusoidal pattern on hexaplex radix.

The RD equation used is based on equation 2.4 from [4], which uses an activator-inhibitor model.

$$\frac{\partial a}{\partial t} = \sigma b a^{*2} - r_a a + D_a \frac{\partial^2 a}{\partial x^2} \tag{14}$$

$$\frac{\partial b}{\partial t} = b_b - \sigma b a^{*2} + D_b \frac{\partial^2 b}{\partial x^2} \tag{15}$$

$$a^{*2} = \frac{a^2}{1 + s_a a^2} + b_a$$

$$\sigma = \rho + r_a[\tau + \nu \sum_{i=1}^{\eta} sin(x \cdot rand())]$$

a represents the concentration of activator and b represents the concentration of inhibitor. $\sigma b a^{*2}$ is the production rate for a, and b_b is the production rate for b. $-r_a a$ is the depletion of a, and $-\sigma b a^{*2}$ is the depletion rate for b. $D_a \frac{\partial^2 a}{\partial x^2}$ is the exchange by diffusion. σ is mainly a superposition

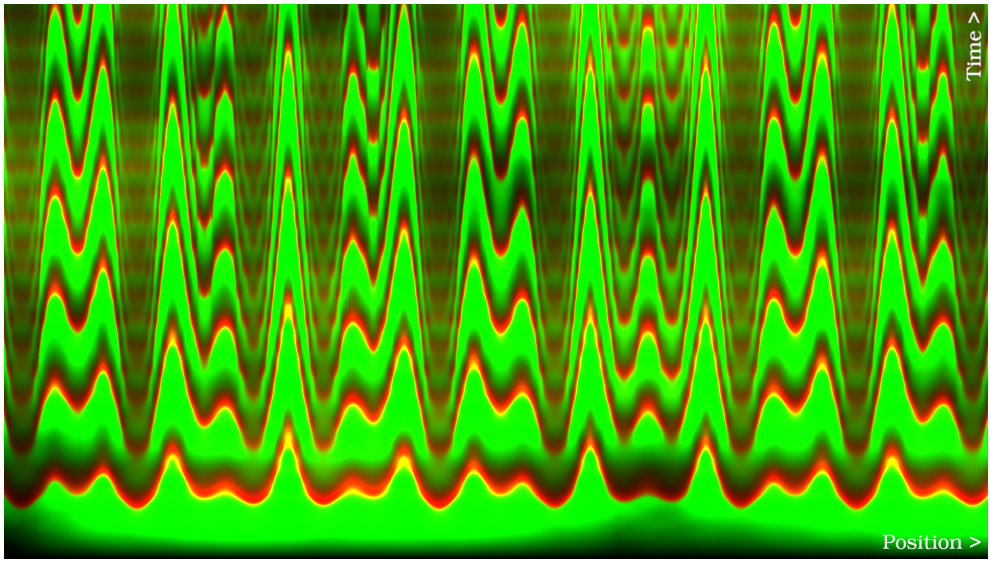

Figure 16: Image created from equations (7) and (8). Green value is inhibitor, red is activator. Each value is multiplied by the Fractal Brownian Motion shader. [13]

of *sin* waves to give the pattern a chaotic appearance. This produces a 1-D cellular automata, where the x-axis represents the position and the y-axis represents time.

Because the protrusion of hexaplex radix have a mountainous appearance, it seemed appropriate to try adding some fractal noise to the final image. An additional shader was created which multiplies the values of the activator-inhibitor shader by the values of a fractal brownian motion (FBM) texture described in [5]. It is difficult to notice in Figure 16, because the RGB values are not clamped to 0 to 1 in the RD shader. Perhaps this should be changed in later attempts.

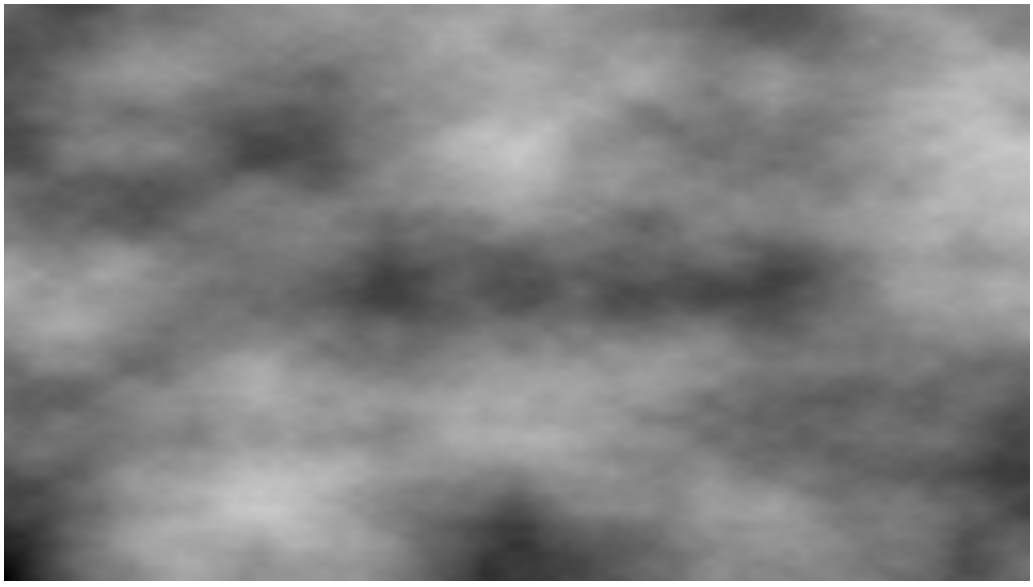

Figure 17: Fractal Brownian motion. [14]

Then the activator concentration, red, was converted to the uniform RGB color of the height map texture.

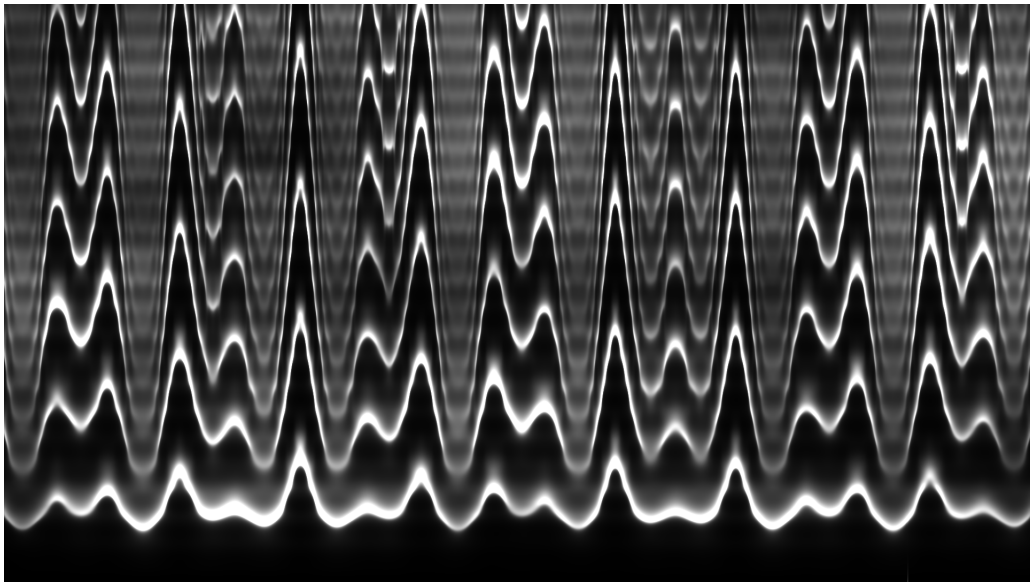

Figure 18: Actual height map texture. Red activator value becomes the white color.

5 Putting It All Together

Everything was put together in Blender. The original shell model, seen in Figure 13, was modified so that all non-visible vertices and faces were removed. Then the height map RD texture was UV mapped to the shell.

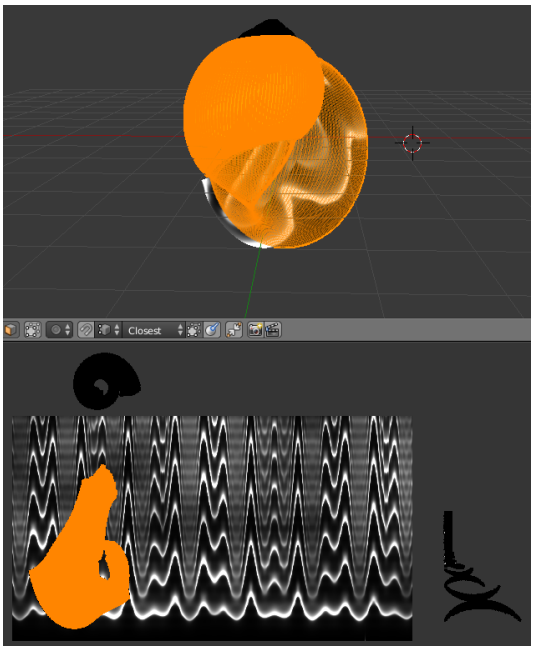

Figure 19: UV mapping of the texture onto the shell model–.

Here is the final model in Blender, with the Solidify modifier applied:

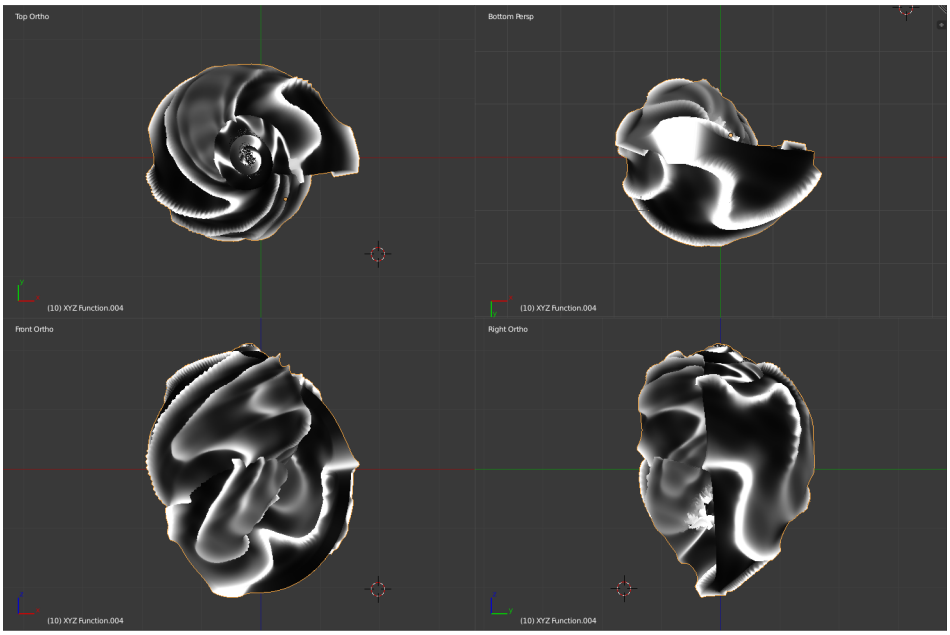

Figure 20: Final shell model. *radix_shell_textured.stl* contains the geometry of the shell.

I think it looks better with the height map inverted.

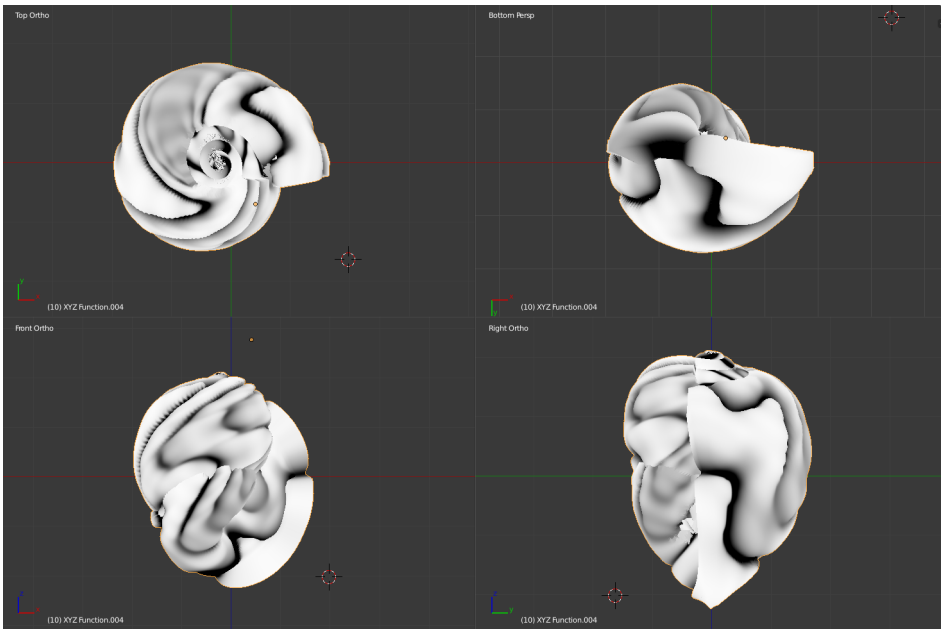

Figure 21: Inverted height map texture applied.

Unfortunately I didn't get to printing the model.

6 Conclusion

The striking feature of this project was the change in 3-D modeling technology from the late 1990's. Graphics were likely mostly done using the CPU, so raytracing was dominated rather than meshes, and thus the early project related to creating sea shell models used raytracing techniques. In contrast, application such as Blender can do real-time modification to meshes, which allows for far quicker iteration on equation parameters. Modern tools are also quite general and widely

available. It was probably quite a feat for Galbraith et al. to write a CSG library almost 2 decades ago, but modern applications comes with that functionality built in. The trickiest thing about this project hasn't been writing correct software, but rather finding the correct tools and learning how to use them.

As can clearly be seen, there is quite a discrepancy between the intended goal in Figure 1, and the final model in Figure 20. Some small tweaks that may make the resemblance more alike would be to change equations (14) and (15), such that the *sin* values of σ have a higher frequency and smaller amplitude. Doing that in combination with improving the UV mapping such that it fit more correctly would likely improve the similarity between the model and the goal. RD equations can be somewhat unruly and I lack the mathematical dexterity to create new equations that might better model the actual behavior. The UV mapping is also kind of tricky and might require an artistic touch. Also mentioned earlier, it might be a good idea to try to clamp the RD texture's values to a maximum of 1, such that the FBM shader will be more apparent.

Another interesting idea to model the protrusions is to use iterative fractal methods, which could be generalized to produce something like this:

Figure 22: Chicoreus palmarosae [10]

References

[1] David Bachman. Modelling seashells. http://mathartblog.com/?p=257.

[2] Francesco De Comité. Modelling seashells shapes and pigmentation patterns: Experiments with 3d printing. *Bridges 2017: Mathematics, Art, Music, Architecture, Education, Culture*, 2017.

[3] Callum Galbraith, Przemyslaw Prusinkiewicz, and Brian Wyvill. Modeling murex cabritii sea shell with a structured implicit modeler. 2000.

[4] Hans Meinhardt. *The Algorithmic Beauty of Sea Shells*. Springer-Verlag, 2009.

[5] Jen Lowe Patricio Ganzalez Vivo. Fractal brownian motion. https://thebookofshaders.com/13/.

[6] Jorge Picado. Seashells: the plainness and beauty of their mathematical description. http://www.mat.uc.pt/ picado/conchas/eng/article.pdf.

[7] Greg Turk. Generating textures on arbitrary surfaces using reaction-diffusion. *SIGGRAPH Comput. Graph.*, 25(4):289–298, July 1991.

[8] Aadjan van der Helm, Peter Ebell, and Willem F. Bronsvoort. Modelling mollusc shells with generalized cylinders. *Computers and Graphics*, 22(4):505 – 513, 1998.

[9] Eric Weisstein. Logarithmic spiral. http://mathworld.wolfram.com/LogarithmicSpiral.html.

[10] Wikipedia, the free encyclopedia. Chicoreus palmarosae. https://upload.wikimedia.org/wikipedia/commons/thumb/1/13/Chicoreus_-palmarosae.jpg/266px-Chicoreus_palmarosae.jpg.

[11] Wikipedia, the free encyclopedia. Hexaplex radix. https://upload.wikimedia.org/wikipedia/commons/thumb/f/f1/radix_01.jpg/1200px-Hexaple_radix_01.jpg.

[12] Mike Williams. http://www.econym.demon.co.uk/isotut/shells.htm.

[13] Edward Ye. Reaction diffusion + fbm. https://www.shadertoy.com/view/tsjSz3.

[14] Edward Ye. Simple fbm. https://www.shadertoy.com/view/wdSXD3.