Neon Waves: A Model for Computer Generated Simulation of Bioluminescent Ocean Waves

Natalie O'Leary

Advisors: Professor Theodore Kim and Professor Adam Finkelstein

Submitted in partial fulfillment

of the requirements for the degree of

Bachelor of Arts

Department of Computer Science

Princeton University

May 2021

Abstract

Under very specific conditions, certain types of marine bacteria have the ability to emit a blue-green glow that is visible to the human eye. This glowing phenomenon is called bioluminescence and it is a striking and rare occurrence in nature. In this thesis I propose a model for recreating this phenomenon as it occurs in ocean waves using the effects software, Houdini. The bacteria in the water are stimulated by the wave forces, which results in a luciferase reaction that produces light. This effect can be accurately reproduced using a FLIP fluid simulation and treating a portion of the FLIP particles as bacteria. The forces on the particles can be directly translated to emission in Houdini, producing an effect that is visually convincing and rooted in the real natural phenomenon.

Acknowledgements

First and foremost, I would like to thank my advisor Professor Ted Kim. I am so grateful you responded to my cold email and agreed to help a random student with little to no graphics experience to pull together a thesis. You let me see what was possible in the world of graphics, and you guided me through every step of the way. You are such a kind and thoughtful person as well as an amazing professor and I am so lucky to have completed this project with you. Thanks for helping me figure out my life as well as my project and for always looking out for me.

Thank you also to Professor Finkelstein for acting as my Princeton advisor so I could pursue a subject I was really passionate about. Thanks for helping me out even when you already had too many advisees.

Thank you Mom and Dad for loving and supporting all my weird interests and for going to the ends of the Earth whenever I needed you. To my whole wonderful family, I love you guys. And thank you Haimie.

Thank you to my wonderful roommates Erica and Myrha for picking me up off the floor and giving me food. Thank you Sarah for being my mom. Matthew, thanks for being you and for believing in me. To Zyanne, thank you for teaching me how to code when I had absolutely no idea what was going on. Thanks Ariel and Lucy for always being there for me despite being very far away.

Contents

Abstract iii

Acknowledgements iv

List of Figures vii

1 Introduction **1**

2 Background and Related Work **4**

2.1 Bioluminescence 4

2.2 Wave Physics 8

3 Approach **11**

3.1 FLIP Simulations 11

3.2 Set Up 12

3.2.1 Wave Control 13

3.3 Configuring the Glow 13

4 Implementation **15**

4.1 Representing the Wave 15

4.1.1 Large Ocean Representation 15

4.1.2 Wave Node Topology 18

4.2 Calculating Glow 24

4.2.1 Determining Acceleration 25

4.3 Shading 27

4.3.1 Attribute Ramp and Falloff Equation 27

4.4 Tone Mapping 29

4.5 Whitewater 29

5 Results **31**

5.1 Visual Comparison to Real Phenomenon 31

5.2 Emulation of Underlying Physical Behavior 35

5.2.1 Concentration of Bacteria 35

5.2.2 Bacterial Behavior 38

6 Conclusions and Future Work **39**

6.1 Future Work 39

6.1.1 Whitewater 40

6.1.2 Quorum Sensing and Attenuation 40

6.1.3 Distribution of Particles 41

List of Figures

2.1 Different bacterial bioluminescence to evaluate quorum sensing. [5] . 5
2.2 p(t) # of photons emitted per second [13] 7
2.3 Bioluminescence Signal Decomposition [13] 8
2.4 Banzai Pipeline [12] 9

3.1 Initial Node Network for the Flat Tank 12

4.1 Node Graph for Large Ocean Simulation 16
4.2 Reference Image with Flatter Parts Circled [3] 17
4.3 One Frame of My Simulation, with Similar Flattened Portions 17
4.4 Object Level Topology of Wave Simulation 18
4.5 Wave Force Topology 19
4.6 VolumeVop Node Inner Topology 21
4.7 X ramp and Z ramp at frame 29 22
4.8 Inside the AutoDopNetwork node 23
4.9 Wave Acceleration Colored in XYZ Directions 26
4.10 Ramp Controlling How Glow Attribute is Mapped to Color 27
4.11 Different falloff equations for percentage of current and previous force 28
4.12 Without selective illumination on the foam 30
4.13 With a light that only illuminates foam 30

5.1 Surfer on Bioluminescent Wave, Photo by Michael Latz [9] 32

5.2 A Similar Frame From My Simulation 32
5.3 Bioluminescent Pipe Wave, Photo by Rob Wessels [16] 33
5.4 Simulated Recreation of Figure 5.3 33
5.5 Wave Rolls into the Beach, Photo by Andrew Loera [10] 34
5.6 Frothy Frame of My Simulation 34
5.7 Wave With 2/3 of particles selected to glow 36
5.8 Also 2/3 of Particles Glowing, Selected Differently 36

Chapter 1

Introduction

The field of natural world simulation has been revolutionized in recent years and has grown exponentially in the past decade. Sophisticated algorithms have been developed to simulate everything from hair to rubber bands to fluids. However, the field is still relatively young, and there are many natural phenomena that have not yet perfected or even attempted in animation. Once algorithms have been perfected to replicate a specific phenomenon like fire or fluid, these tools can be used by animators for years to come to create compelling and realistic depictions of the real world. In fact, the technology and models available to animators often determine the direction of animated games and films, and thus greatly influence the general population.

Take for example, the first completely computer animated major motion picture, *Toy Story*, released in 1995. This film was revolutionary in the field of both film and animation, however the reason the movie was centered around toys was simply because plastic was the easiest surface to render at the time. They didn't have the capabilities to make realistic looking humans (evidenced in the appearance of eerie Andy in the film, and in the fact that his parents faces were not even attempted), so they made a movie about plastic characters instead [14]. Moreover, the influence of computer animation extends far beyond the realm of children's movies. With their paper on "Wavelet Turbulence for Fluid Simulation," Kim et al. created a new standard for fluid simulation in CGI, which can be used to simulate both fire and smoke in animated and live action movies alike [8]. Beyond the realm of film, the ability to accurately model and recreate natural phenomena with CGI has practical uses. In fact, just this year the snow model created for the movie *Frozen* was used to provide an explanation to the decades long investigation of the deaths of 9 hikers in the Dyatlov Pass [1].

The goal of my project is to study the patterns of bioluminescent bacteria in ocean waves, and to recreate that phenomenon in the effects software, Houdini. While this phenomenon involves previously modeled aspects such as waves and luminescent objects, it has never been formally and specifically modeled for an academic paper. The interactions between the force and acceleration of ocean waves and bioluminescent bacteria are very specific and thus cannot be modeled with pre-existing tools. The challenging aspect of this undertaking comes from the nature of biological versus graphical study. While there is an abundance of research regarding bioluminescent bacteria, the majority of this work focuses on their biological makeup, rather than their aesthetic appearance. As such, there are quantifications of the light emitted by these bacteria in many studies, however they tend to be taken in isolated, unstimulated lab settings, and provide information on underlying behavior, rather than visual appearance. This project rectifies that discord by providing a model for replicating the intensity and fade of individual bacterium from the time of stimulus until the light has been completely quenched. This equation is derived from existing biological studies of the behavior of the bacteria, as well as visual studies and iterative models of their behavior in ocean waves. Bioluminescence is a very rare phenomenon, and it is notoriously difficult to capture, both on film and in lab environments. It is difficult to determine when and where it will even occur, so replicating it in order to study has proved difficult to scientists for years. By deriving the equations to model the luminescence of the bacteria, as well as providing the groundwork to recreate compelling simulations of this phenomenon I hope to make it more accessible to those who are unable to travel to see it. In turn, these clearer visualizations may lead to scientific breakthroughs in this long elusive subject, much like the breakthrough in the Dyatlov Pass.

Chapter 2

Background and Related Work

2.1 Bioluminescence

The first important paper which informs my work is “In Situ Measurement and Correlation of Cell Density and Light Emission of Bioluminescent Bacteri”, published in 2018 by Brodl et al. [5]. This paper emphasizes the difficulty of isolating and studying bioluminescence due to the wide variety of bacteria that can display this phenomenon. Bioluminescence in protists is caused by the *lux* gene which encodes luciferaces luxA-F, which catalyze a reaction that produces long chain acids (CH3(CH2)nCOOH), flavin mononucleotide (FMN), and water [5]. The energy released in this reaction causes FMN-4a-hydroxide to enter an excited state which in turn emits the light that we see as bioluminescence [5]. However, this gene is very common in many different types of bacteria, all of which demonstrate different patterns of light intensity and bioluminescence regulation. Each type of bacteria also has a different method of ”quorum sensing,” which is a property that allows individual bacterium to perceive and respond to the behavior and population density of other bacteria in the colony through gene regulation. This makes it especially difficult to measure bioluminescence *in Situ*, as the perceived light emission in a section of ocean is likely the result of several different species of bacteria, all of which have different patterns for emission, and different methods of controlling emission through quorum sensing. It is also difficult to transfer these marine bacteria to a lab setting, so this study instead utilized the lab ready E. Coli bacteria, and mutated them by introducing the *lux* gene, so that it could be studied in a controlled lab setting. They then use this data and apply it to *in Sito* recordings, to isolate the sources and behaviors of specific bacterial emissions. They found that they were able to replicate the behavior of the bioluminescent *P. mandapamensis* bacteria which was studied in an *in Situ* setting. Results from this study are shown in in Figure 2.1. This study provided the basis of my knowledge on how bioluminescence works, and why it is so difficult to study, but it does not delve into the visual appearance of the illuminated bacteria.

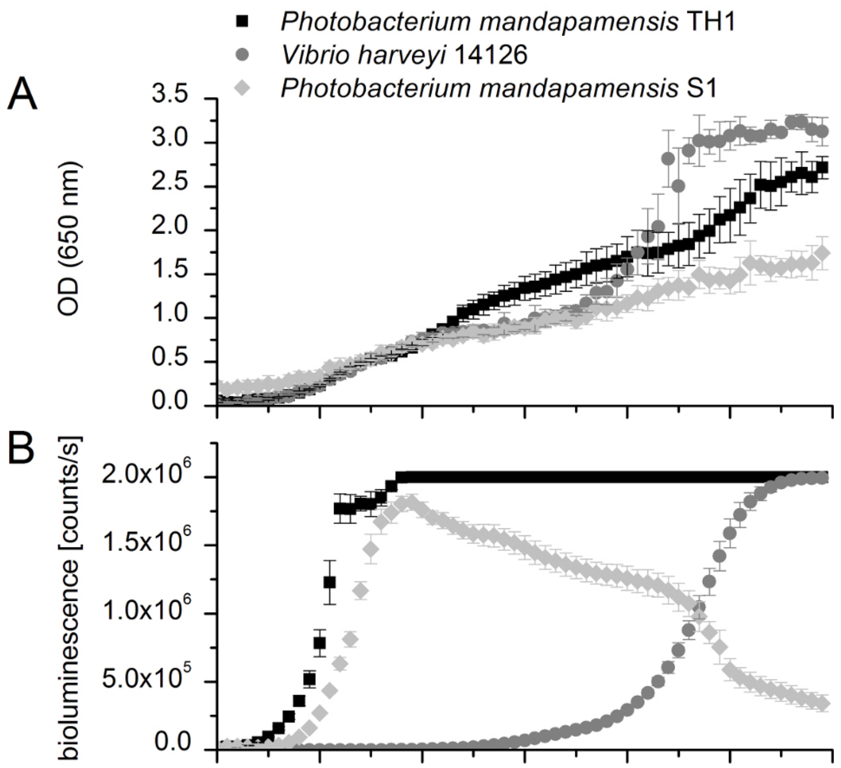


Figure 2.1: Different bacterial bioluminescence to evaluate quorum sensing. [5]

While Brodl et al. transferred the luciferase gene to E. Coli, Valiadi et al. studied the bacteria by tagging the gene itself in bacterial colonies. This process is detailed in the 2014 paper, “Molecular detection of bioluminescent dinoflagellates in surface waters of the Patagonian shelf during early austral summer 2008” which deals with the density and dispersal of dinoflagellates in the water of the Patagonian Shelf [15]. This study also attempts to isolate specific sources of bioluminescence among many light emitting bacteria, however they chose to focus on the most common light emitter, the dinoflagellate. In addition to measuring the bioluminescence levels using a bathyphotometer, the team in this study also tagged the luciferase gene found in dinoflagellates with a PCR primer to more accurately measure the population of bacteria [15]. They found that optical measurements of bioluminescence often vastly underestimate the amount of bacteria present. This is in part due to the fact that bioluminescence may only be perceptible at high densities, and only occurs as night. Additionally, intensity and fade patterns vary widely across different bacteria and across different strains of the same bacteria. This is helpful to my research, as it means that attempting to directly recreate a visual simulation based on recorded bacterial populations will likely not produce the desired result. There is a discord between the phenomenon perceived by the eye and the underlying biological occurrences in the scene. Modeling the bacteria in a controlled, computer environment could help bridge this gap and provide more information about the phenomenon.

The paper entitled “Bacterial bioluminescence onset and quenching: a dynamical model for a quorum sensing-mediated property” provides a basis on which to recreate a specific amount of light emission from bacteria in a lab setting. This paper focuses on quantitatively recreating bioluminescence, by providing a model to determine light intensity at any point based on population size and density. Furthermore, this paper provides a figure for “quenching time” [13] which describes the time elapsed from the activation to the end of emission for bioluminescent particles. A photomultiplier tube was used in this study, in a completely dark room containing the bacteria to measure the bioluminescence over time. This study had a particular focus on quorum sensing, and determined that the bioluminescent signal is controlled primarily by both:

1. The growth of the bacterial colony
2. "The increase in the number of bioluminescent cells due to quorum activation" [13]

This accounts for the non-linear behavior of bioluminescent activation. Quorum sensing was actually *discovered* in the study of bioluminescent bacteria, as it was observed that upon quorum activation, the ratio of bioluminescent bacteria among the population grew. The bacteria are able to secrete signalling molecules called "autoinducers,"[13] and each bacterium in turn has receptors to detect these autoinducers. When the concentration of autoinducers that direct the population to glow is high enough, it will trigger quorum activation, which "triggers an intracellular signalling cascade that results in a phenotypic switch"[13]. In this case, the phenotypic switch results from a higher abundance of the *lux* gene, which allows the cells to glow. The non-linear model this paper proposes provides precise equations for photons emitted by the colony at any given time depending on colony size, and percentage of quorum active bacteria, as shown in Figure 2.2. Models fitted to the

$$p(t) = \kappa_e c_e a(t) = \kappa_e c_e [f(t)n(t) - f(t_0)n(t_0)].$$

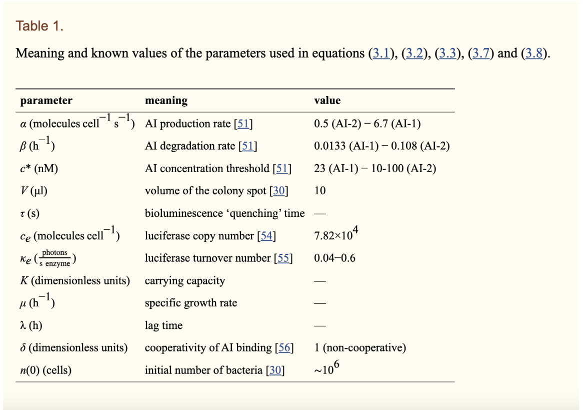


Table 1.

Meaning and known values of the parameters used in equations (3.1), (3.2), (3.3), (3.7) and (3.8).

parameter	meaning	value
α (molecules cell^{-1} s^{-1})	AI production rate [51]	0.5 (AI-2) – 6.7 (AI-1)
β (h^{-1})	AI degradation rate [51]	0.0133 (AI-1) – 0.108 (AI-2)
c^* (nM)	AI concentration threshold [51]	23 (AI-1) – 10-100 (AI-2)
V (μl)	volume of the colony spot [30]	10
τ (s)	bioluminescence 'quenching' time	—
c_e (molecules cell^{-1})	luciferase copy number [54]	7.82×10^4
κ_e ($\frac{\text{photons}}{\text{s enzyme}}$)	luciferase turnover number [55]	0.04–0.6
K (dimensionless units)	carrying capacity	—
μ (h^{-1})	specific growth rate	—
λ (h)	lag time	—
δ (dimensionless units)	cooperativity of AI binding [56]	1 (non-cooperative)
$n(0)$ (cells)	initial number of bacteria [30]	~10^6

Figure 2.2: p(t) # of photons emitted per second [13]

behavior of bioluminescent colonies have been produced before, however prior to this study, a predictive model for light emission had never been successful in its predictions. This is essential to my research, as my work needs to be able to adapt a model to consistently replicate the phenomenon from scratch. The researchers in this study were even able to separate the proportion of the luminescence due to quorum sensing and colony size into discrete parts shown in Figure 2.3

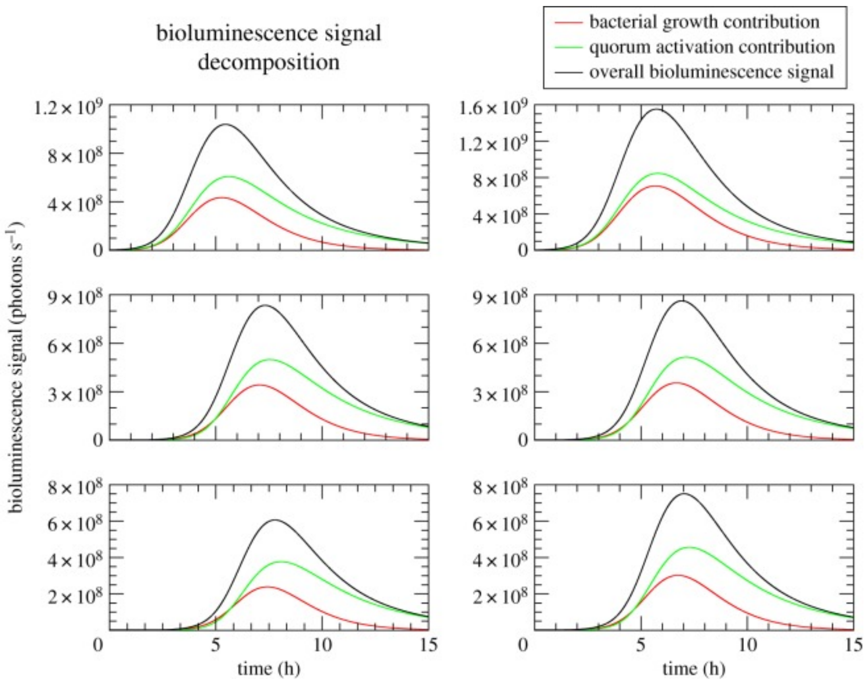


Figure 2.3: Bioluminescence Signal Decomposition [13]

2.2 Wave Physics

Peter Janssen's book, "The Interaction of Ocean and Wind," published in 2004, goes into great detail about the physics of wave formation and wave patterns in the ocean. There are many kinds of ocean waves, from small swells to huge waves breaking near the shore. The wave pattern that is most relevant to my research is what is called a "pipe wave" which is a barrel-like wave often conducive for surfer. A famous example of this kind of wave is the Banzai Pipeline which occurs off the coast of O'ahu in Hawaii, shown in figure 2.4. Ocean waves follow the pattern of orbital progressive

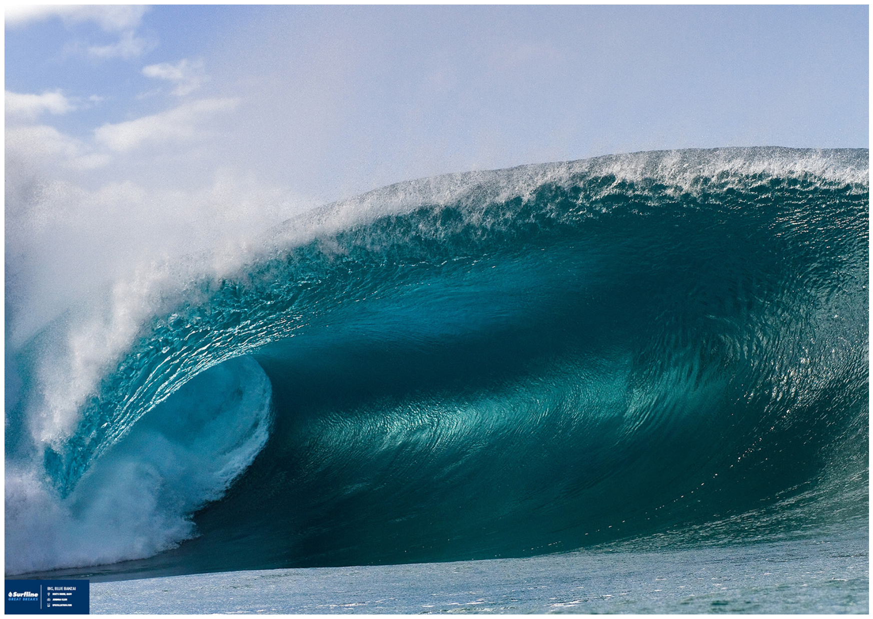


Figure 2.4: Banzai Pipeline [12]

waves [7], meaning the particles in the wave travel in circular motions, up the crest of the wave, and then down the trough to under the surface of the water. The first step in wave formation is wind, which blows on the surface of the water forming white, frothy peaks. These peaks then increase the surface area available to the wind, which allows the wind to further push the water, and form larger, taller swells in the ocean's surface. When the wind is blowing consistently in one direction, these swells begin to form a pattern, where they are all moving away from the wind. Through constructive interference, some of these swells actually combine to create larger waves [7]. As the waves get closer to the shore, the ground beneath them begins to slope upward toward the beach, and the water gets shallower and shallower. With less water to go up through the orbital swell, the waves closer to the shore move forward with less velocity, and the waves following them begin to get close in behind them. The resulting effect is that by the time the water particles reach the top or front of the wave, they're slower and begin to fall more sharply, rather than in a circular path, while the back of the wave still retains the circular shape. The encroaching ground beneath the waves pushes the waves higher up as the waves near the shore, and eventually they will "break." Breaking is actually where the circular back of the wave essentially overtakes the slower front of the wave, causing the water to spill over and form the barrel that we are looking for. The shape and duration of the barrel depends largely on the slope of the ground below the wave approaching the shore. A gradual slope will cause the wave to break in a more collapsed fashion, causing flatter, foamy waves to travel toward the shore. The pipe wave is included in a category called "plunging breakers" in which the wave breaks at a point where there is either a large jump or very steep slope in the ground beneath the wave that causes the back of the wave to go much faster than the front and spill up over the top creating a barrel [2]. The final component for the pipe wave is that the direction of the wave is not parallel to the shoreline, or the slope of the ground beneath. This means that the whole wave will not break at once, since the wave does not reach the breaking point all at once. The wave will break all the way across slowly, which causes the sustained barrel that surfers can ride on. This sustained wave is also the most interesting pattern in which to study bioluminescent light patterns, as the phenomenon can be viewed in a continuous wave, however the particles themselves are always being cycled through the wave, so different bacteria will be present in the wave as it progresses and the glow can be sustained longer than it would in stagnant water.

Chapter 3

Approach

3.1 FLIP Simulations

The foundation of my simulation is in the FLIP fluid simulator, which was created in 1986 by Brackbill et al. [4]. FLIP stands for Fluid Implicit Particle, and it is a hybrid between volumetric and particle based fluid simulations. In this case, this means that the fluid information is stored in the particles themselves, which persist frame to frame. In Houdini, this includes position and velocity as well as color. At each frame a volume is dynamically produced on an adaptively zoned grid. The fluid flow is calculated on a Lagrangian mesh, however since the particles store all of the information related to direction and velocity, the mesh is not strictly necessary, which is why it can be adaptively zoned. In addition to velocity, the particles also have a momentum property, to prevent particles from grouping and all moving in the same direction and velocity. Particles in the same location in a FLIP simulation collide according to their momentum values. The flow of the fluid is calculated on the Lagrangian Mesh, and then the solutions are transferred back to the particles.

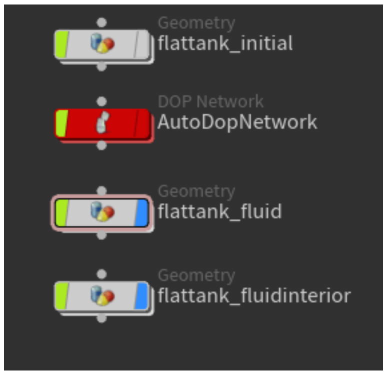


Figure 3.1: Initial Node Network for the Flat Tank

3.2 Set Up

The FLIP simulator is the ideal Houdini tool to use for simulation of the wave, since the shape and volume can be controlled at a high level, but the individual particles are easily mutable as well. This is essential for being able to simulate the glowing bacteria in the waves, as in this case they will be represented by glowing FLIP particles. The control that a FLIP simulation lends is especially useful in choosing a subset of the particles to glow, as opposed to an entire section of the simulation. Each particle is represented in the simulation with its own unique ID, velocity, position and color. Additionally, attribute expressions can be configured into the Houdini node network to add properties to the particles such as acceleration, which is necessary in order to calculate the individual force on each particle which will be the catalyst for the glow operation. The basis for the simulation therefore is the "Flat Tank" tool from the Houdini shelf tools. This tool automatically creates four nodes at the object level of Houdini: flattank_initial, flattank_interior, flattank_fluid, and AutoDopNetwork, as shown in figure 3.1. The flattank_initial node controls the initial configuration of the tank, so this node controls the size, shape and position of the tank at the start of the simulation. The AutoDopNetwork node is what controls the majority of the simulation. The FLIP solver network is within the AutoDopNetwork node. The flattank_fluid node combines flattank_initial and the AutoDopNetwork and renders the simulation to output. Surfacing can also be controlled from the flattank_fluid node. Finally we have the flattank_interior node which controls the volume of the simulation, and the interior particle properties like opacity and murkiness.

3.2.1 Wave Control

Control and formation of the wave is controlled by a velocity grid that provides the source for the FLIP simulation, rather than simulating the influence of wind on the ocean. Because we have full control over the particles themselves, we can achieve a much more precisely controlled wave by directly controlling the velocities of the particles to achieve the orbital progressive wave, rather than relying on the perfect storm of ground slope and wind speed. The wave form tool in Houdini can be used to create the swells in the water, and then the velocities can be hard coded such that the velocity in the back of the wave overtakes that of the front of the wave creating the pipe wave. Additionally, the wave has to be at about a 10 degree angle from parallel to the bounding box of the flat tank, so that the wave will break from right to left rather than all at once. The geometric mesh of the wave animation is stored in the file cache located in the flattank_fluid node, and it is this geometry that will be used in the next step.

3.3 Configuring the Glow

The exported geometry is then imported to a new Houdini network. The particles of the FLIP simulation are retained in this new representation, so they can still be accessed and manipulated individually. To create the effect of glowing bacteria, a new expression has to be written such that each particle stores the velocity at the previous frame, which is used to calculate the acceleration of the particle. The acceleration can then be used to calculate the force on the particle at that point, which will in turn trigger the glow reaction. The glow itself is controlled by a linear ramp, such that very low amounts of force do not cause any glow, but above a certain threshold, the particles will glow brightly. This ramp is shown in figure 4.10.

Further, an exponential falloff is used to control the fade of the particles after the initial excitation, so that the particles will eventually stop glowing, though they continue to be pushed by the acceleration of the wave. This is because way that bioluminescent bacteria have a refractory period in which they cannot immediately glow again, nor can they sustain glow for long. This will all be calculated directly on the particles and translated into a color distinct from the base color of the mesh. The actual emission quality of the particles has to be controlled by the material surfacing of the mesh. The Principled Shader in Houdini has an emission property, through which you can control the intensity and color of the emission at any point on or in the mesh. We can use the color we assigned using the force and falloff equations in the mesh to control the emission intensity, because we can manually assign the emission color and the base color of the mesh using the shader. These colors will override the indicator colors we used to feed into the emission intensity, and the resulting mesh will glow at all of the places where the unshaded mesh was colored. The intensity of the glow corresponds to the intensity of the color on the linear glow ramp.

Chapter 4

Implementation

4.1 Representing the Wave

4.1.1 Large Ocean Representation

Before moving on to the smaller FLIP simulation for greater control of the environment, I created a larger ocean simulation, with the intent of representing the phenomenon in its natural context. To do this I used the Large Ocean shelf tool in Houdini, which creates a starter topology as seen in figure 4.1. This simulation is very large, and even rendering a single frame takes a very long time, so you can see that it has both a render_grid and a preview_grid. The render grid is 8000 by 8000, so the preview grid is simply a copy of the render grid that can be shrunk down to a reasonable size to preview the result. The ocean_preview node is where the preview grid is rendered, and the save_spectra node is where the whole simulation is rendered. Shaders applied in these nodes transform the simulation from wavelike deformations on a grid, to the waterlike surface that you see in figure 4.3. The foam nodes are optional and are not relevant to this simulation. The oceanspectrum nodes are what control the bulk of the simulation, as they control the loop on which the simulation runs, which controls the wind and wave patterns. The wind attribute controls

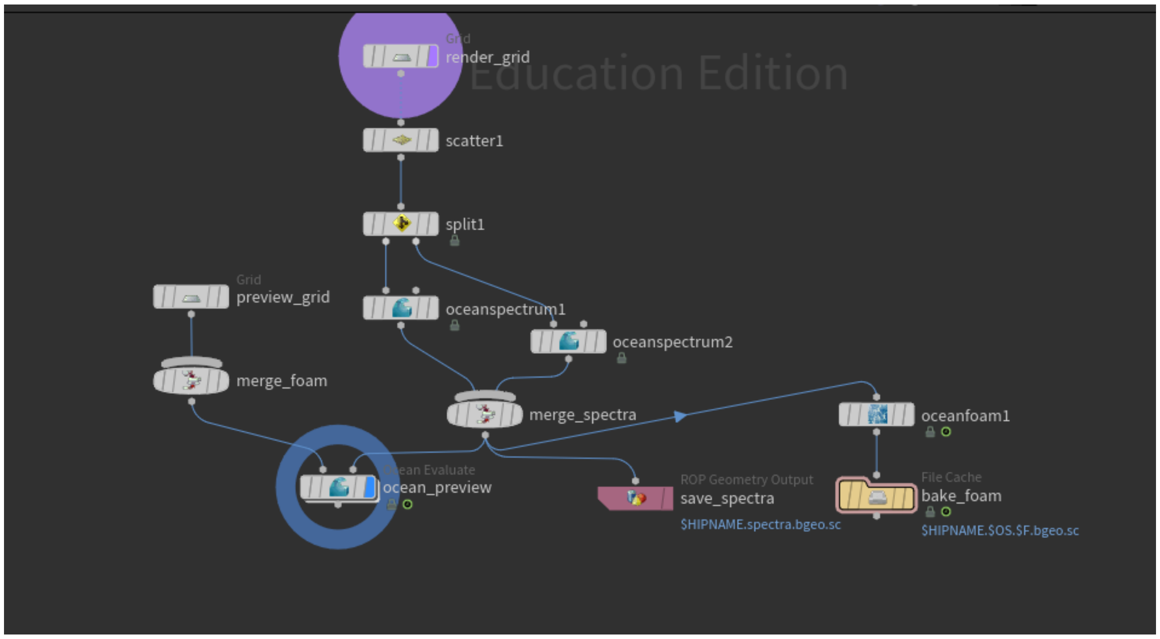


Figure 4.1: Node Graph for Large Ocean Simulation

whether there are waves at all in the simulation, and from here one can manipulate the speed, direction and variety of the wind. Additionally the oceanspectrum node enables controls of the wind amplitude, which controls the radius of the wind force, and the pattern in which it repeats. Once wind has been enabled, the waves can then be directly modified as if accounting for the subsurface environment that we are not directly simulating (e.g. the effects of ocean floor, plant life etc.). The Wave Instancing tab can be used to override the effects of the generated wind to add randomness to the created waves in selected or random locations throughout the surface of the simulation. There are two ocean spectrum nodes in this node topology to further randomize the patterns of the ocean, so the simulation looks more natural, and irregular. My implementation of the large ocean simulation actually had four oceanspectrum nodes to further naturalize the effect. One of the spectrum nodes that I included implemented the mask tool to suppress waves in certain patches of the ocean, to mimic the circled patches in figure 4.2. One frame of my simulation can be seen in figure 4.3. This type of fluid simulation is completely surface based, and

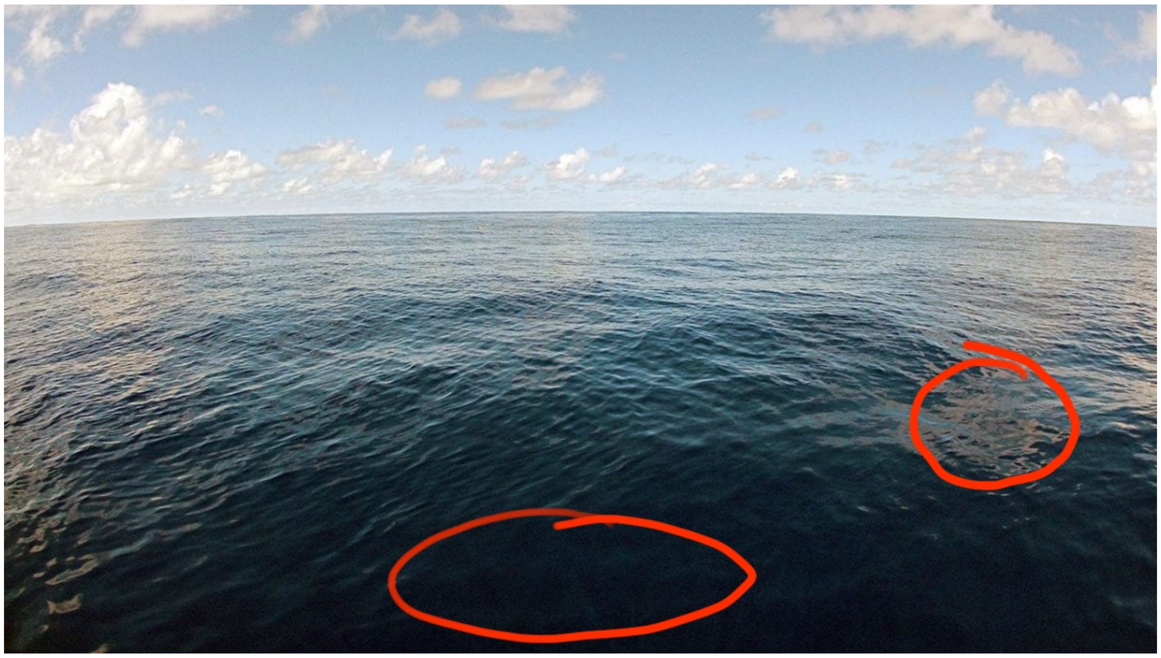


Figure 4.2: Reference Image with Flatter Parts Circled [3]

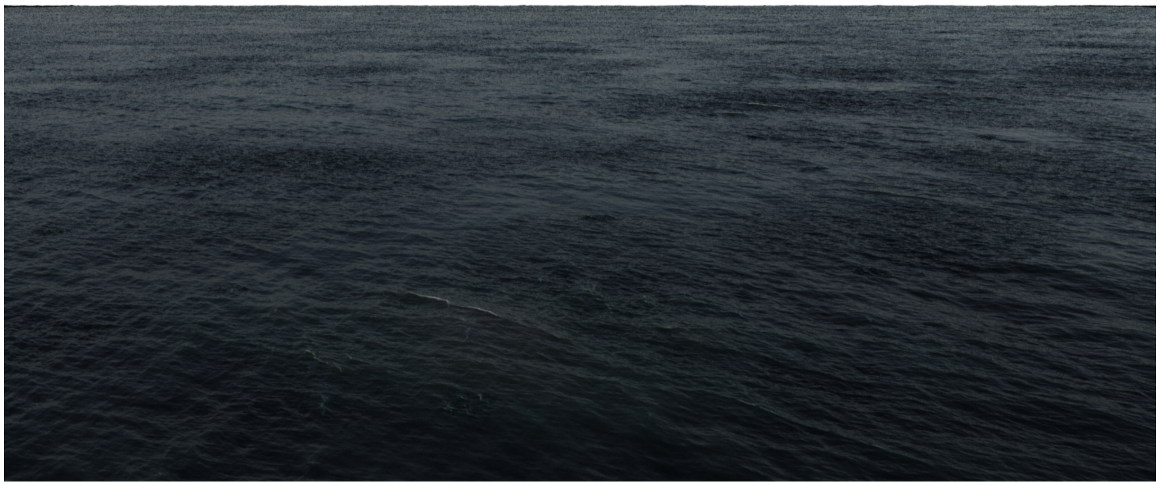


Figure 4.3: One Frame of My Simulation, with Similar Flattened Portions

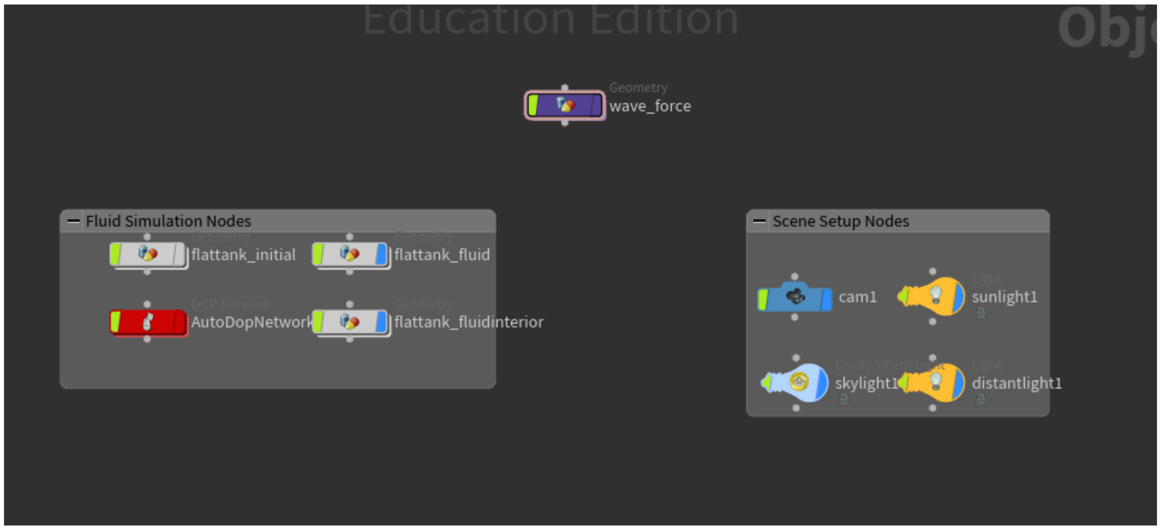


Figure 4.4: Object Level Topology of Wave Simulation

carries no information about individual particles in the simulation. Thus, there is no way for underwater bioluminescence to be represented in this form. Additionally, I need to be able to represent the bacteria as individual particles within the simulation that have their own values for emission and velocity. This mode is ideal for creating smaller waves that occur in the body of the ocean, however it is difficult to manipulate the wind in the oceanspectrum to perfectly emulate the wind conditions that would create a pipe wave, especially without the ability to control the actual ground beneath the waves, which play a large part in shaping the wave. Therefore, this was not a conducive environment to create my simulation in.

4.1.2 Wave Node Topology

After trying and failing to implement my simulation using a grid based ocean surface, I discovered FLIP simulations, which provide the basis of my final simulation. I have already discussed the foundational node topology of a simple flat tank flip simulation in Houdini, as shown in Figure 3.1 which is explained in chapter 3. The actual topology of the nodes used in my simulation can be seen in figure 4.4. The essential nodes are the same as the setup described in the Approach section, however you

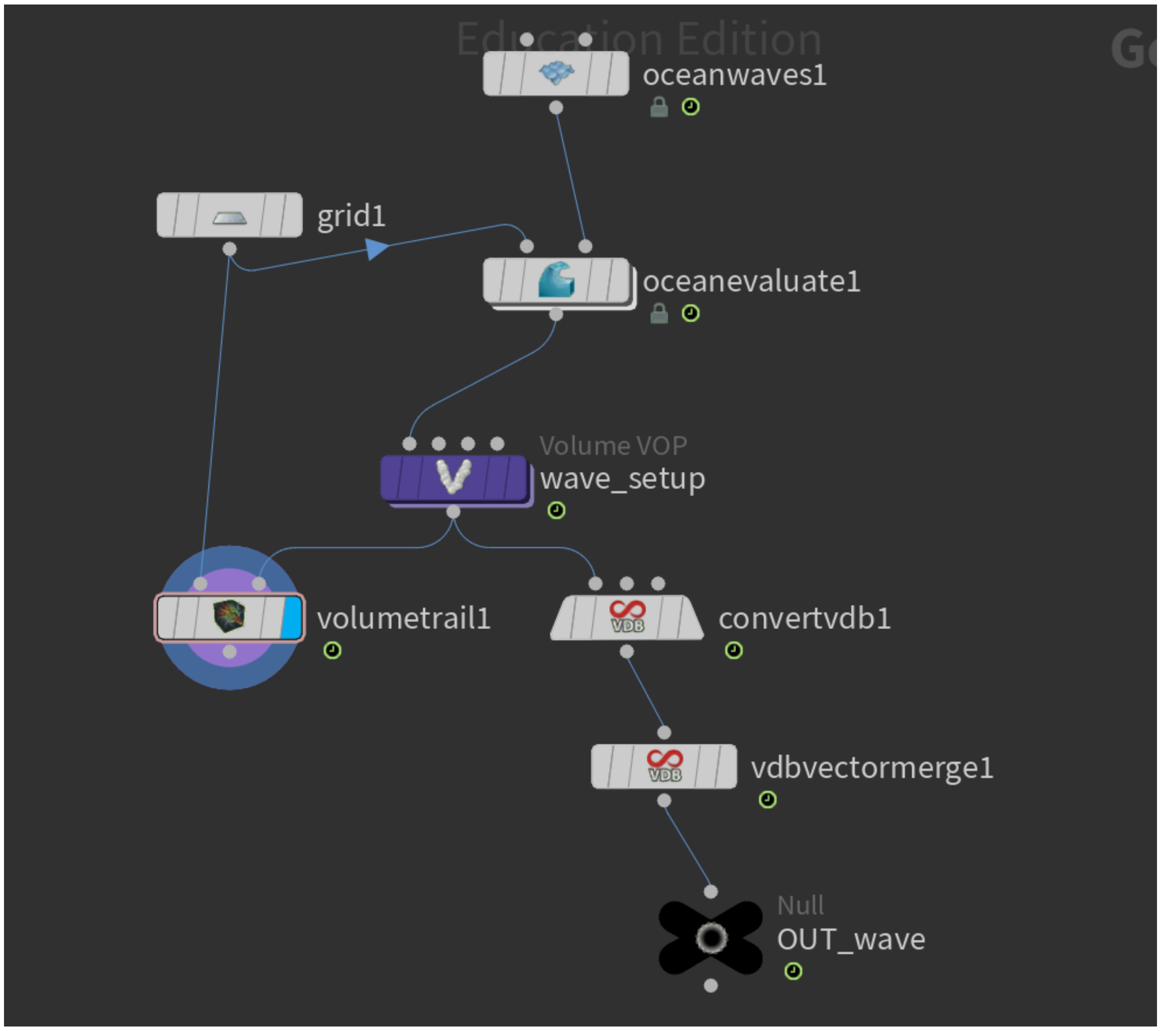


Figure 4.5: Wave Force Topology

can see that there are lighting and camera nodes that are necessary for rendering the scene. Additionally, the most important node that differs from the foundational example is the wave force node. To understand what it does, we have to dive into this node's topology, which is shown in Figure 4.5.

I adapted this topology from a tutorial by Carlos Parmentier on YouTube [11]. The topology is relatively similar to that of the large ocean. You can see that the velocity is structured on a grid node, which is visualized through the volumetrail node, similar to the setup in the large scale ocean simulation. The oceanwaves node directly controls waves on the grid rather than influencing their shape with wind.

With this tool, you can also choose exactly how many waves to make, their amplitude, direction, crest width speed and radius among other tools. This lends much greater control than the oceanspectrum node, which creates a spectrum of waves, rather than one single wave. For this simulation, I created just one wave using the oceanwaves node, with direction 10, meaning a 10 degree rotation from parallel to the box. I set the offset of the wave to -45, because an offset of 0 will have the wave start in the center of the flat tank box. The boxes dimensions are 55x15x40, so an offset of 45 will have the wave begin just outside the box. To have the wave rise and break and fall exactly inside the box, I set the speed to 6 and the crest width to 50. The oceanwaves node provides the input to oceanevaluate in this case along with the grid, rather than an oceanspectrum node. The oceanspectrum node creates three volumes as input to oceanevaluate: phase, frequency and amplitude. Because oceanwaves does not provide any volumes, just waveforms, we need to use the oceanevaluate velocity tool to create a velocity volume from the waveform we created in oceanwaves. The oceanevaluate node is then wired into a volumevop node for more precise control over the exact velocities of the wave. The volumevop node can constrain and bind the flow of a fluid using scripts input by the user. The node network that I created inside the volumevop node explains how this is done. This network is shown in figure 4.6. The volumeglobal node simply represents all possible attributes to be manipulated with this node. The selected attribute, BB, represents position inside the bounding box of the volume, which in this case is our velocity volume. The vecttofloat node that this is wired into takes the position vector and gets the x, y and z values. The x value is wired into ramp1 and the z value is wired into ramp2. These ramps are spline ramps, which map the input position to a value on the ramp, which I shaped manually. The first bind node is used to bind the volume's velocity to this expression, so that it is completely controlled by the volumevop node. This bind is then connected to the multiply node along with the x and y ramps so that the wave's velocity is multiplied

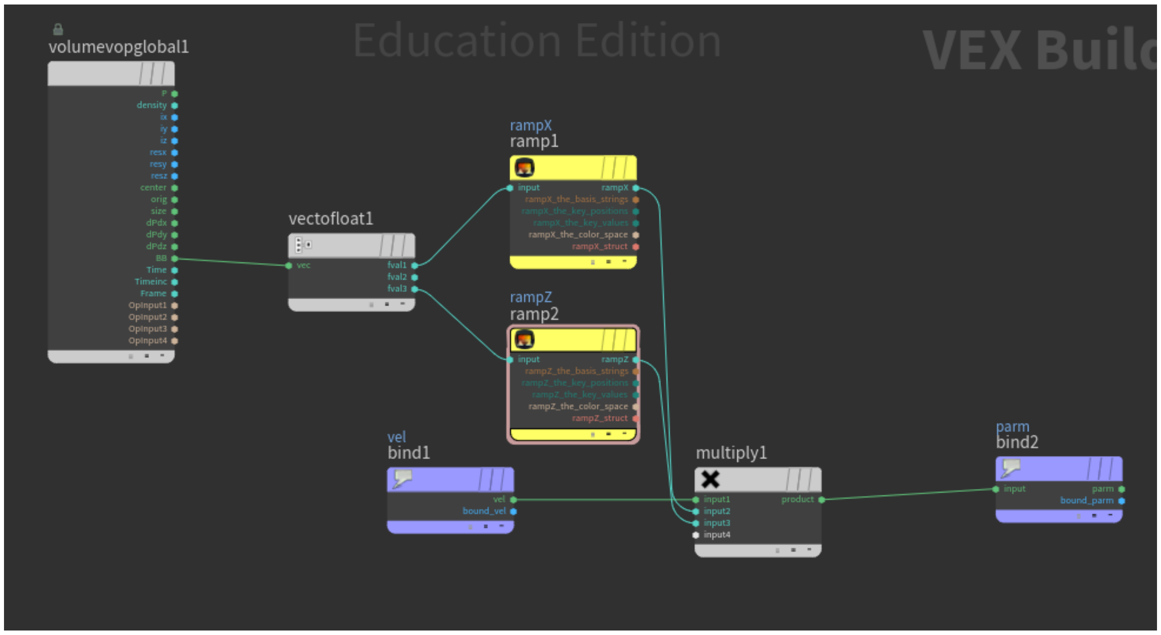
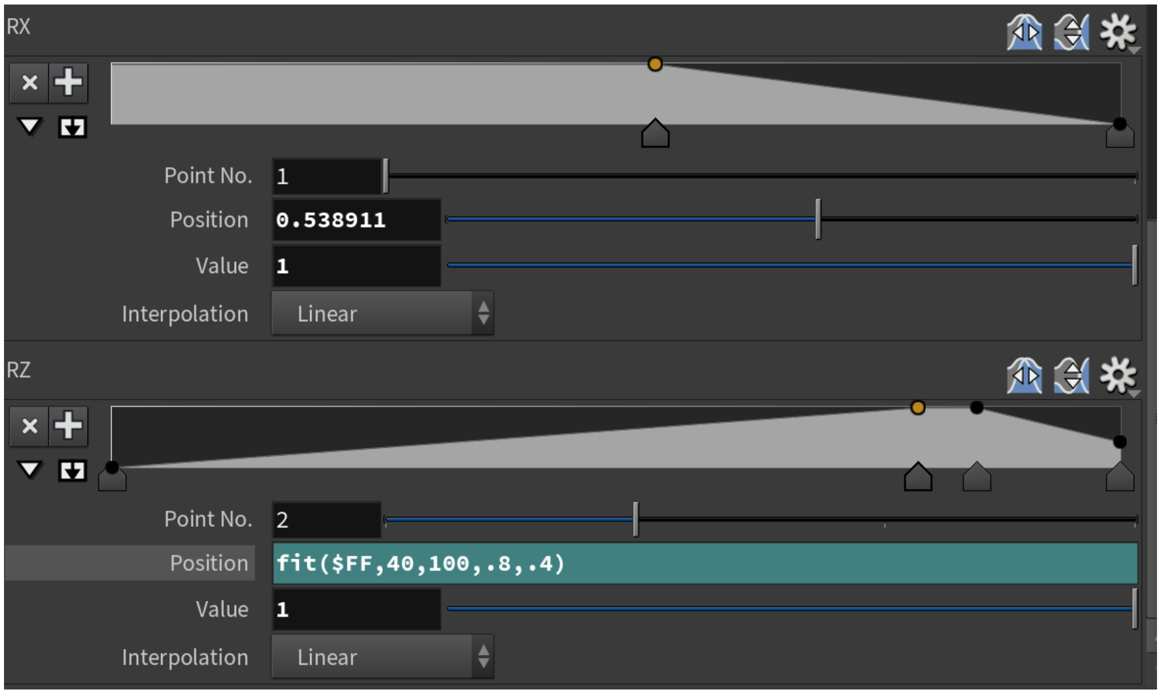


Figure 4.6: VolumeVop Node Inner Topology

by the ramps, thus shaping the velocity according to the ramps. This is wired into the second bind node which then exports this as the new wave velocity value. These ramps are shown in figure 4.7. The X ramp is fairly simple, with the mapping simply mapping to 1 until about halfway through the ramp, at which point there is a linear falloff to 0. This is what will cause the breaking of the wave, as the velocity will be steady until at a certain point it steeply drops off, at which point particles that are farther back will be faster then those in the front, and they will overtake them, as explained in chapter 2.2. The Z ramp is slightly more complicated in order to achieve the rolling break that we want, that goes from the right to the left side of the simulation. First I made a new arrow on the ramp, which you can see is under the yellow circle in figure 4.7 on the RZ ramp. I then input an expression taken from the pipe wave tutorial [11], which you can see in the position box in the image. The "fit" function maps a value from one range to a new range. So the value in question is $FF, which is the current frame, and it maps the old range, 40, 80 to .8, .4. This translates to: between frames 40 and 80 in the simulation, the highlighted yellow arrow will



RX

Point No. 1

Position 0.538911

Value 1

Interpolation Linear

RZ

Point No. 2

Position fit($FF,40,100,.8,.4)

Value 1

Interpolation Linear

Figure 4.7: X ramp and Z ramp at frame 29

begin at 80% of the way along the ramp move closer to the position 40% along the ramp as the simulation progresses to frame 80. The value at this arrow is 1, and the value at the first arrow is 0, so essentially, this function makes the ramp from 0 to 1 steeper and steeper from frame 40 to 80, increasing velocity more rapidly as the frames go on [11]. The first arrow on the RZ ramp, at the 0 position also needed an expression, however the expression at this point controls the value not the position of the arrow. This expression is fit($FF,120,200,0,1), [11] which steadily increases the value of the arrow at position 0 from 0 to 1 from frame 120 to frame 200. This means that while velocity is increasing steadily from 40 to 80, it begins to even out from frames 120 to 200 as the bottom of the ramp increases, thus flattening it out. The velocity from the wave we initialized in oceanwaves is then multiplied by these ramps to shape it, and then the mesh's velocity values are bound to the output using another bind node.

Now that the wave velocity is completely shaped to the specifications of a pipe

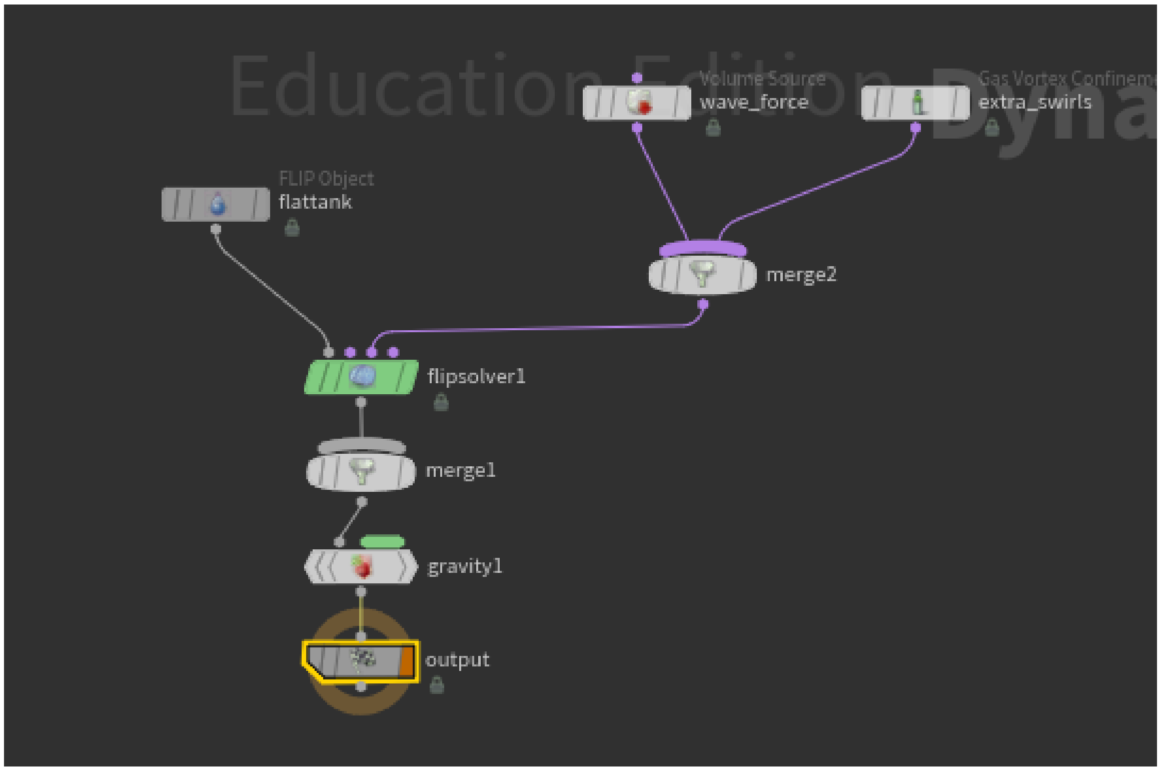


Figure 4.8: Inside the AutoDopNetwork node

wave, the velocity can be wired into the FLIP solver to actually influence the fluid particles. Figure 4.8 shows the inside of the AutoDopNetwork node, which is where the FLIP solver is located. The green flipsolver node is where the volumes are created from the particles at every frame. The volume output by the flipsolver can be subjected to pressure and gravity forces in the way that large bodies of water are affected collectively, rather than as individual particles. The merge node below the flipsolver is only there in case the user intends ot add multiple solvers to their scene. The gravity node applies gravitational forces to the volume, and the output node renders and saves the volume if specified by the user. The interesting parts of this network however are the topmost nodes in the system. The flattank node initializes the particles in the system, and is a direct import from the flattank_initial node that is located at the object level of this node system. It determines the number of particles, the separation between particles, the shape of the box itself, the location of the box and the size of the box among other things. These particles are then fed into the first input of the FLIP solver, providing its basis. The other two nodes, wave_force and extra_swirls are merged and then fed into the volume velocity input to the flipsolver. These nodes do not provide any new particles to the simulation, but simply apply external velocity solvers to the flipsolver once it has created the velocity fields at each frame. The wave_force node is just an import of the exact wave_force node that was explained in the previous section. The extra_swirls node is a gas confinement node, which amplifies smaller vortex swirls in the simulation so that swirls like this are not lost in the larger scale simulation. This adds roughness ad variation to the edges of the wave especially, so the falling of the water is not so clean in the absence of real air resistance. The output of this node is then merged with the flattank_initial node contents and exported as a .bgeo mesh file.

4.2 Calculating Glow

I created a second Houdini network for the glow portion of the simulation, as it doesn't affect the physics of the particles and the mesh. Therefore, once the wave physics were perfected, I exported them as a mesh as described in the previous section so they're not constantly being dynamically solved during the color and glow simulation. This mesh still retains the individual attributes of the particles in the FLIP simulations, so it is still an ideal representation for individually bioluminescent particles. The idea therefore is to calculate the acceleration of each individual particle, and use that to find the force on the particle at any given time, which triggers a glow response above a certain threshold. This glow then decreases steadily from the initial excitement.

4.2.1 Determining Acceleration

The first challenge with the particles in Houdini is that while they each have their own properties for position and color, they have no memory of previous frames, and so have no acceleration property. However, there is a node in Houdini called the Time Shift node, which you can program to keep track of attributes at a frame other than the current one. To calculate acceleration, I created a Time Shift node expression that always holds the attributes of the frame immediately after the current frame. I then used this node to calculate the difference in velocity between the current frame and the next frame for every single particle in the simulation to find the accleration using an Attribute Expression node. This node can be used to create expressions and equations such as the one I used in this case, and then will set the result to a new or existing attribute of the particles. This new acceleration attribute persists in any node wired below the Attribute Expression node in the topology. Acceleration can then be directly used to calculate the force on each of the particles in addition to collision and gravity forces. This force is what will be used to determine the triggering of glow in the particles. In my first attempt to translate force acting on the particles to glow, I attempted to translate acceleration on the particles directly to glow. This posed multiple problems:

1. Glow was sustained for far too long
2. Glow was too homogeneous in large portions of the simulation
3. Acceleration is a vector broken down into XYZ directions, leading to different results based on direction rather than force alone

The results of this attempt in a 3D model can be seen in figure 4.9 Acceleration in the Z direction is represented in yellow, x direction in red and y direction in blue with brighter colors representing larger accelerations. You can see that the main area of acceleration is just after the crest of the wave, where there is a large

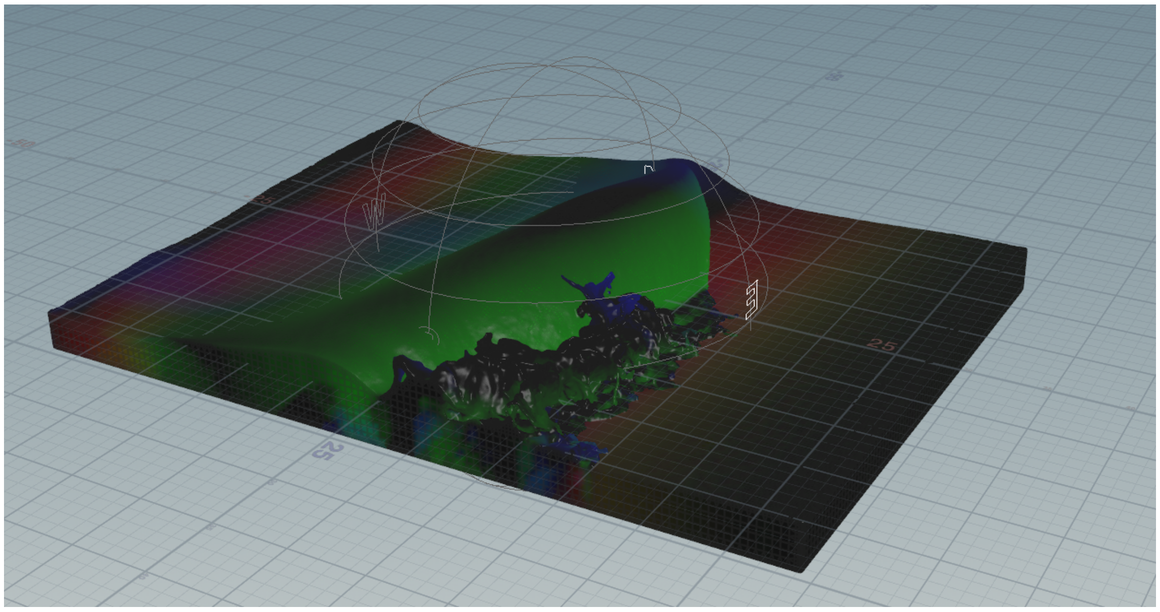


Figure 4.9: Wave Acceleration Colored in XYZ Directions

block of green (a mix of x and y accelerations). I used a color node to change the color of particles according to force, and this color is in turn translated into glow. However, this is ineffective if the force and therefore the glow is segmented into XYZ directions, because the bacterial luciferase reactions are affected by force, but not direction. Therefore I needed to create a scalar attribute that accounted for both initialization and decay of glow. I created a second Time Shift node that keeps track of the attributes of the particles at exactly one frame *before* the current frame. This node was used to calculate the current glow of the particles, which comes partially from the current force acting on the particle, and partially on the amount of glow that the particle possessed in the previous frame. This pattern simulates an exponential falloff in glow from previous frames, and a linear translation between new force and new glow. The specifics of this equation can be found in section 4.3.1. The glow Attribute Expression node is then wired directly into a color node, which sets the surface color of the mesh at each particle location according to the input attribute, in this case glow.

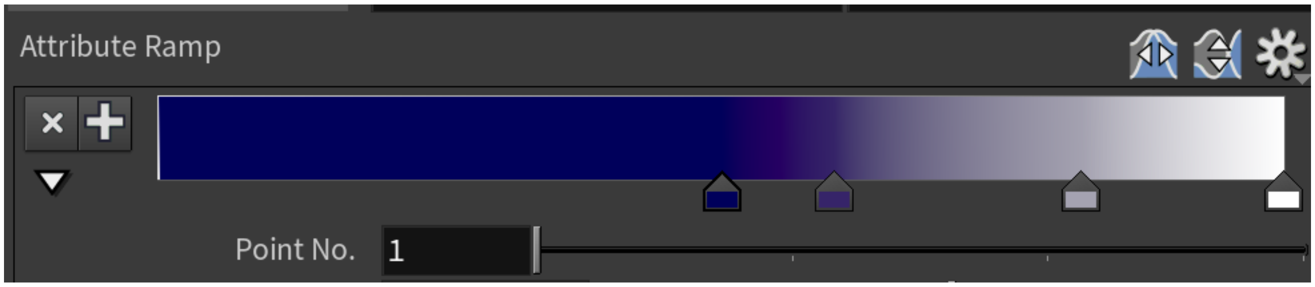


Figure 4.10: Ramp Controlling How Glow Attribute is Mapped to Color

4.3 Shading

Converting from colored particles to light emitting particles happens in the material shader network, rather than at the object level. I therefore created a Principled Shader node in the materials network, and set the material of the mesh to this shader. I wired a Surface Color Node into the Emission Intensity input of the Principled Shader node so that the intensity of the emission would be directly correlated to the intensity of the surface color which I can control directly at the object level. This is because particles can be easily accessed at the object level but not at the shader level. Thus, coloring the particles according to glow intensity at the object level allows them to be easily grabbed to be illuminated by the shader. The emission color itself I set manually to be the aqua-cyanne color observed in bioluminescent waves in the ocean. For the base color of the shader, I chose a dark navy color, which mimicks the color of the ocean at night when these phenomena are recorded. I also used this shader to set the reflectivity and translucency of the material.

4.3.1 Attribute Ramp and Falloff Equation

As mentioned in section 4.3, the emission intensity is controlled directly by the range of color intensities in the mesh surface. The ramp that converts the glow attribute to color is shown in figure 4.10. As seen in the figure, the glow attribute ranges from just 4 to 5.5, although the maximum glow value calculated throughout the entire simulation is actually around 9.5. This is because particles only achieve this maximum value for a few frames, while the actual phenomenon of bioluminescent waves is strikingly bright and sustained. Thus setting the top of the ramp to be lower than the maximum value of the glow attribute created a more dramatic rendering. The ramp is not linear however, and remains at the base color until about halfway through the ramp. At this point, the color begins to change from dark blue to white. From the middle of the ramp to the end of the ramp is a gradient to white, which contributes to the fade of intensity corresponding to the amount of glow calculated for this particle.

The fade of each particles' glow intensity is also controlled by the glow attribute equation itself. While incorporating quorum sensing was too big of a problem for the scope of this project, I attempted many falloff equations to account for how particles would behave after initial illumination. I began with a simple exponential falloff reducing the glow from the previous frame by 90% at each particle before adding that value to 50% of the current force on the particle to determine the amount of total glow at this particle. I then adjusted the percentages used to determine which value pairings best mimicked the appearance of the real phenomenon. I had to adjust the maximum value of the attribute ramp shown in Figure 4.10 accordingly. This process of rendering the same sequence many times with only small adjustments made in between is called wedging, and the wedges produced during this process are shown in Figure 4.11. As you can see in these wedges, the first figure has a much bigger impact

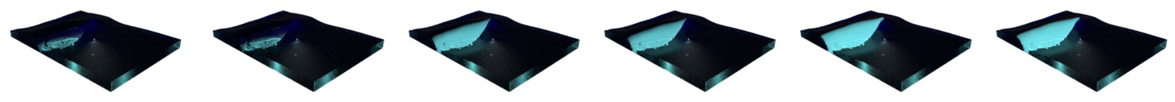


Figure 4.11: Different falloff equations for percentage of current and previous force

on the end result than the second figure. I determined that 50%, 10% formulation for current force and previous force looked the most similar to the phenomenon itself.

However for more precise control over the end result, I had to edit the tone mapping of the rendering when converting from EXR to .MP4 files.

4.4 Tone Mapping

Houdini's render engine exports the rendered images as EXR files, which are rasterized file formats with many more channels of information than are translated into a JPEG or MP4. It is therefore necessary to manually adjust the way that the colors and lights are converted to images through a process called tone mapping. The EXR files have a much larger range of values than most image files, so tone mapping is how you translate those more diverse colors to a simpler format so the colors aren't as blown out as you see they are in Figure 4.11. This tone mapping is done in Adobe After Effects, which can take EXR files as input, allowing the user to directly control how the EXR channels are interpreted.

4.5 Whitewater

At this point, the only missing component from the simulation is whitewater, which is always present when a wave breaks. The surface tension and the direction of the fluid flow is broken where the wave breaks, as the back of the wave spills over the front of the wave. This rapid change in water direction leads to a disrupted "laminar flow" which leads to aeration in the fluid which appears as a froth on the fluids surface [2]. To create whitewater in a FLIP simulation, you need to create another solver that uses the same flat tank as a source fluid. The whitewater solver generates white water stemming from the original FLIP simulation, though they still exist as two separate simulations that are technically rendered separately. Thus, I generated and rendered out the white water geometry separately but concurrently with the rest of the FLIP simulation. I then imported the white water geometry into the Houdini file that controls the shading and glow values of the simulation. One limitation of this implementation is that the white water simulation in Houdini is not particle based once the white water is emitted, though it takes particles as input. As a result, the foam itself cannot contain glowing particles, though foam in the ocean still contains the bioluminescent bacteria. Whitewater has not been perfected to the extent that the rest of fluid simulation has, so I could not control it as precisely as the fluid for the wave. Thus I did not concentrate my work on this section of the simulation, and it was added only after the rest of the wave had been perfected. In order to mimic the effect of glowing whitewater, I included a light in the simulation that only illuminates the foam and no other geometries in the scene. Without this, the whitewater was barely illuminated by the dark night lighting and the emission of the wave and the renderings appeared dark like in figure 4.12. Adding a light that only shines on the whitewater makes it seem like the foam itself is glowin as well, like it should in nature, as scene in figure 4.13.

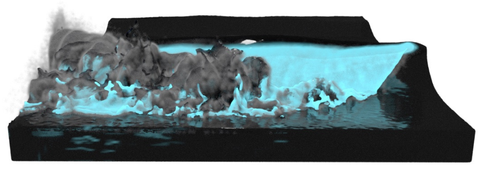


Figure 4.12: Without selective illumination on the foam

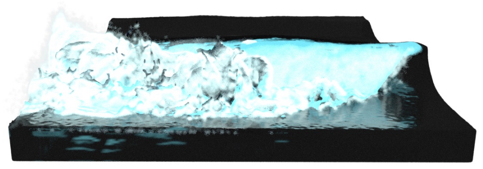


Figure 4.13: With a light that only illuminates foam

Chapter 5

Results

The finished animations and renderings for this project can be found at natalie-oleary.com/portfolio.

Evaluating the success of this project is not easily quantified with statistical error measurements. However I have laid out a set of criteria with which to measure the success of this project. In Chapter 1 I stated that my goal was to create a simulation of bioluminescent bacteria in an ocean wave in an attempt to provide a framework for recreating this phenomenon in Houdini, as well as to provide insight into the phenomenon itself. I wanted to create a simulation that is realistic enough to be recognized as, and possibly mistaken for, the real phenomenon. Thus I will evaluate this simulation based on:

1. It's ability to realistically visually recreate the natural phenomenon

2. The similarity of the underlying physics to that of the real phenomenon

5.1 Visual Comparison to Real Phenomenon

I have selected three photos of bioluminescence in ocean waves in the wild to visually compare to renderings from my simulation. These images are shown in Figures 5.1,

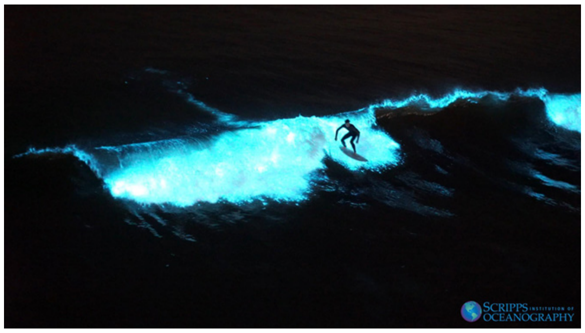


Figure 5.1: Surfer on Bioluminescent Wave, Photo by Michael Latz [9]

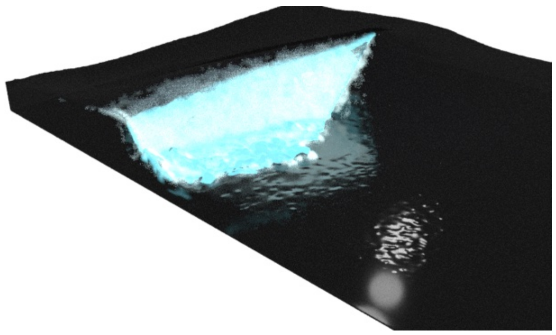


Figure 5.2: A Similar Frame From My Simulation

5.3, and 5.5. My simulation is not a direct recreation of any of these photos, and each reference photo is from a different instance of bioluminescence, while all frames of my simulation come from the same simulated scene. So, the goal is not for my renderings to exactly match any of the reference photos, but instead to capture the general properties of the phenomenon.

The first two images to compare are shown in Figures 5.1 and 5.2. The reference photo in Figure 5.1 is obviously not exactly like my simulation, as the surfer's contact with the water triggers more luminescence reactions and more whitewater where the wave is breaking. However, the important part of the photo to compare is to the right of the surfer. My simulation achieves a similar color of illuminated particles as this photo, and the angle and shape of the wave is very similar. The bioluminescence begins at a similar location on the wave in both photos, and the slope of the demarcation line up the wave is similar, assuming the line in the surfer photo would extend at a linear rate were the surfer not present. The reflection of the glow on the water is present in both images, though it is difficult to compare with these two images, because reflection does not play a large part in the image as a whole. One shortcoming is that the wave in the real image is surrounded by much choppier water, which appears more natural. It was necessary to limit other waves and forces on the water in order to maintain complete control over the main wave in my simulation. However the result is somewhat unrealistic. Additionally, the white water in my simulation is not photo-realistic, however this effect was added on at the end using a preset Houdini tool and is not the focus of my simulation.

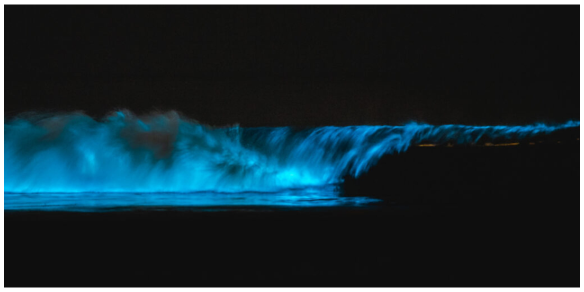


Figure 5.3: Bioluminescent Pipe Wave, Photo by Rob Wessels [16]

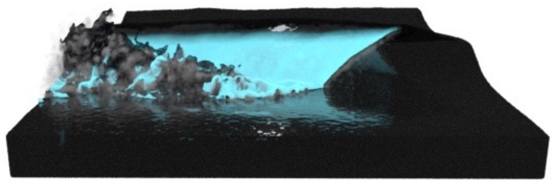


Figure 5.4: Simulated Recreation of Figure 5.3

Figures 5.3 and 5.4 show a later stage of the wave, when about 80% of the wave has spilled over. You can see that the shapes of the wave as well as the resulting spray of water are congruent in the two photos. The color is not quite the same, however I had to pick one color for my simulation, and there is a wide variety of colors in bioluminescent bacteria. This frame of my simulation really showcases the success of the reflection in my simulation. The shape of the reflection just below the breaking line of the wave is a hooked shape in both cases, with a comparable light intensity. The dark spots in the upward spray from the wave are analagously placed in both images, though once again, the foam is the least photo-realistic part of the simulation. Additionally the banding of darker shadows across the wave in Figure 5.3 is not present in my simulation, an error which I discussed in Section 5.2.1. The shape of my simulated wave is too clean, and lacks the rougher texture of the water as it breaks in the real image. Where my simulation has a clean line as the water descends toward the surface, the real image has a choppier texture as the water aerates and breaks its surface tension. The FLIP simulation attempts to emulate this by putting whitewater on top of the fluid particles, however the whitewater is a completely separate simulation, and thus does not quite achieve the exact effect.

Increasing the vorticity in the underlying FLIP simulation could help decrease the regularity of the wave, though it may disrupt the shape.

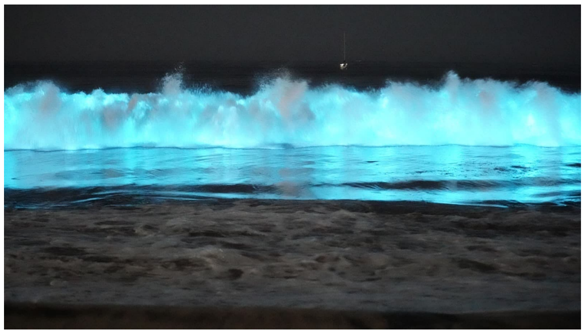


Figure 5.5: Wave Rolls into the Beach, Photo by Andrew Loera [10]

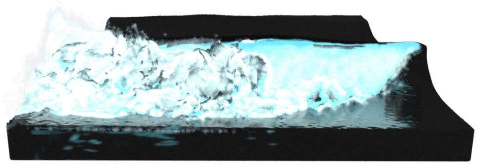


Figure 5.6: Frothy Frame of My Simulation

The final two figures, 5.5, and 5.6 compare the aftermath of the wave, and the resulting foam in the water. Unfortunately, for efficiency, my simulation is only a small segment of the ocean, and thus cannot showcase the reflection of the glow in front of the wave in this frame. However the glowing spray of water itself can still be compared between these two photos. As I mentioned in Chapter 4.5, one of the biggest limitations of my simulation is that whitewater is not a particle based simulation in Houdini. As a result, it does not have velocity attributes that could be wired into some form of illumination in this simulation. Thus, the whitewater in my simulation frame occludes the glow of the particles rather than glowing itself. However, the amount that the whitewater itself glows depends on the concentration and intensity of the luminescence in a given body of water. The whitewater in Figure 5.3 is darker and shadowed, though that entire image is darker than Figure 5.5. Thus, I tried to strike some balance in my simulation, however it will not perfectly match every reference image, and the foam in Figure 5.6 is a litle too bright in color, to compensate for it's lack of light emission.

5.2 Emulation of Underlying Physical Behavior

Often computer based simulation is actually not entirely rooted in the underlying physics of the phenomena. For example, the incredibly popular Phong reflectance model for lighting and reflection is commonly used in animation and simulation. However it is based in empirical observations of light behavior, rather than the underlying physics. On the other hand, attempting to recreate the physical behavior in simulation can give insight to the behavior of the bacteria in nature. Therefore it is worthwhile to investigate how well a model emulates the physics behind the natural phenomenon, though it is by no means a failure if an empirically derived method diverges from the physical basis. The majority of the simulation is physically based. The entire structure of the pipe wave is created directly from the natural forces that drive this exact wave shape in nature. However, the velocities are directly wired into the wave, rather than occurring as a result of wind speed and ground slope. Additionally, like the bioluminescent bacteria, the particles in my simulation are directly stimulated to glow by the acceleration and resulting force from the wave. The simulation as a whole is largely true to the natural phenomenon, however there are a few deviations which I will discuss in the next two sections.

5.2.1 Concentration of Bacteria

One of the hardest things to emulate in this project was the randomess that is present in nature that can only be artificially reproduced using computers. Computers can simulate artificial randomness, however they can never generate truly random numbers. Bioluminescence is a relatively elusive phenomenon, as a very specific set of conditions must be present in order to observe it in the wild. Weather and wind and marine conditions all affect the concentration of bioluminescent algae or bacteria in a body of water, so even in places where the phenomenon is relatively common, it is never exactly the same. The phenomenon appears visually different depending on the concentration of the bacteria, but the way that my simulation is designed, all of the particles in the simulation were originally included in the glow calculations. This meant that if all the particles in a certain area of the simulation had a similar amount of force acting on them at that point, they would all have a similar glow value as well. However, in the real setting, not only do bioluminescent bacteria only make up a small portion of the organisms present in the water, but also not all the bioluminescent bacteria present even translate to glow. Valiadi et al. observed a similar measurement of light intensity in samples with 120 bioluminescent cells/liter and 1000 bioluminescent cells/liter [15]. They also measured instances of visible biluminescence at concentrations as high as 100,000 bioluminescent cells/liter [15]. Thus there is no exact concentration that I could program into my simulation to accurately mimic the real occurrence, and I had to use visual observations and trial and error to perfect this. You can see bands and blobs of light and dark in Figure 5.3 which I tried to emulate by selecting only fractions of the particles to glow.

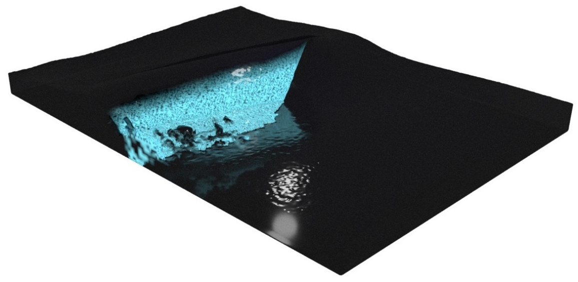


Figure 5.7: Wave With 2/3 of particles selected to glow

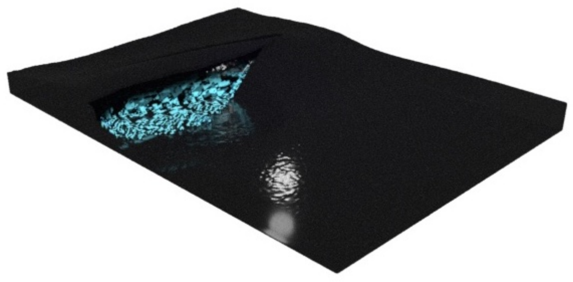


Figure 5.8: Also 2/3 of Particles Glowing, Selected Differently

Figure 5.7, shows an attempt to recreate these sections of non-light emitting water. In the example in Figure 5.7, 60% of the particles in the simulation are selected to glow. However, at any point in the simulation, there are between 30,000 and 70,000 individual particles present, and the way that they are selected greatly changes the patterns of the glow. In the case of the example in Figure 5.7, the particles selected to glow were chosen by particle id, and in order to get 60% of the particles to glow, out of every 3 particles organized in ascending order by index, 2 were selected to glow and 1 was not. However, the effect appears more like unnatural static than the banding of distributed bacteria in the ocean. Another attempt in which 60% of the particles have a glow property is shown in Figure 5.8. Although the same percentage of particles are selected to glow, the visual effect is vastly different because of the distribution of the selected particles. The glowing sections appear in more blobby chunks, however they do not resemble those blobs in Figure 5.3. There are harsh demarcation lines between glowing and non-glowing sections in Figure 5.8, while the lines are softer in Figure 5.3. Additionally, you can see distinct layers of glowing particles in Figure 5.8 where glowing particles lie beneath a layer of non-glowing particles. This depth effect takes away from the illusion of the simulation, as bioluminescent bacteria and algae typically lie on or close to the surface of the water. Thus, the deeper luminance appears unnatural, and dissimilar to the reference photos shown above. Many similar attempts to replicate the random streaks and separate areas of bioluminescence in reference photos proved ineffective as well without an effective means to generate a similar distribution as was present when these photos were taken. Without more information about how the bacteria are distributed throughout the water at the exact moment that these images were captured, it is very difficult to create a random distribution of glowing particles that looks realistic. Additionally, in other reference photos, such as Figure 5.1, There is are no observerable dark spots in the main areas of glow. There are some darker spots at the back of the wave, but it appears as though once the bacteria begin to glow, you cannot see any distinct dark sections among the glowing bacteria. Thus, I decided that in my final rendering, 100% of the particles would be candidates for emitting glow in order to create the most natural visual effect. This is a deviation from the underlying phenomenon, however it is one that ultimately makes the simulation look more natural.

5.2.2 Bacterial Behavior

Although I would have liked to explore the possibility of incorporating a quorum sensing-like communication amongst the particles in my simulation, I was not able to do so in the time that I had to complete this project. Research into the visual effects of quorum sensing in bioluminescence have only been done on stagnant bacterial colonies and do not directly apply to a dynamic wave simulation such as mine. Thus, while the end product of my simulation appears similar to the behavior of the bacteria, the underlying equation does not incorporate quorum sensing at all. Instead I chose to use an excitation threshold and exponential falloff, where the previous value of glow is reduced by 90% before being added to the glow generated by the current force on the particle. This comes close to approximating the fade factor of the bacteria, however it does not account for the fact that at a certain point, bioluminescent bacteria are no longer able to produce light even as force continues to be applied. With the tools I am currently using for this project, I was unable to incorporate conditional if statements dynamically into the node network. Because of this, I was unable to create a variable that tracks the moment when the glow is activated, and I was also unable to use this variable to attenuate the the glow based on the amount of time since this particle began to glow. However, in spite of this shortcoming, like the Phong reflectance model, my empirical observations and visual recreation of the phenomenon still achieve a passable recreation of the real thing.

Chapter 6

Conclusions and Future Work

Summary

Through trial and error and plenty of editing and adjusting, I have create a viable model to recreate the behavior of bioluminescent bacteria in ocean waves. My model achieves similar luminance colors and values as the real phenomenon, and the wave form that I created follows the progression of a natural ocean wave. The pattern of the luminance within the wave is comparable to that observed in nature, as explored in chapter 5. Aesthetically, the project is largely a success, and achieved a level of photo-realism beyond my expectations for the project. The majority of the future work lies in improving the light and attenuation equations for the particles to better reflect the behavior of bioluminescent bacteria.

6.1 Future Work

I intend to pursue this project long after the submission of this paper and my graduation from Princeton University. Thus, this section details the future contributions that I intend to make in this project, as well as work that others may pursue in the future.

6.1.1 Whitewater

One of the greatest limiting factors in the realism of my simulation is the simulation of whitewater on the surface of the wave. At this point in time, there is no perfect model for simulating whitewater, and the most recent SIGGRAPH breakthrough paper on the subject was published in 2007 by Froemling et al. [6]. Currently, whitewater in Houdini is simulated by emitting bubbles and spray from the source fluid. The whitewater exists as a completely separate simulation and solver from the fluid in the simulation, and so can act in a disjoint way from the rest of the simulation. It is sometimes very clear that the whitewater simply rests on top of the simulated water, rather than being a part of it. It is also not particle based and holds no mutable velocity or position attributes. As a result I was not able to treat it like water that may contain bioluminescent bacteria in my simulation, though this is the case with real life whitewater. In the future I would like to be able to make the whitewater react to forces in the same way as the rest of the FLIP simulation. This area of study is currently something that I am not equipped to improve upon, however I look forward to any developments in the field that may allow me to improve upon my simulation.

6.1.2 Quorum Sensing and Attenuation

As discussed in section 5.2.2, I did not get to explore the possibility of quorum sensing to the extent that I would like in this project. It is definitely possible to create some framework of communication between particles in Houdini, and though it was beyond the scope and time frame of this project, it is something I will be exploring in the future. As discussed in Chapter 2, Side et al. propose a viable set of equations for modeling the onset and quenching of bioluminescence involving quorum sensing [13]. This method takes into account bacterial population size, percentage of bioluminescent bacteria, and quorum sensing over time to determine the amount of bioluminescence that will be emitted from a bacterial colony at a given time.

This study was done on stagnant bacteria, and the proposed model does not take outside forces into account which are the primary catalyst for bioluminescence in my simulation. However, the findings from this study can certainly inform the inter-particle communication in a FLIP based bioluminescence simulation.

6.1.3 Distribution of Particles

The final area that warrants continued study after the completion of this project is particle separation and distribution throughout the simulation. While I was unable to find many studies that detailed the visual behavior of bioluminescent bacteria when interacting with outside forces, I did find an abundance of studies on bacterial population size and concentration in the presence of bioluminescence [15], [13], [5]. There is certainly no lack of information that could inform a more accurate ratio of bioluminescent to non-bioluminescent particles within the simulation. While the concentration experiments that I conducted in this project were unsuccessful, it was mostly a lack of time that limited the success. With further exploration of the patterns of particles in Houdini FLIP simulations, I am certain that I can figure out a way to create a simulation that more realistically depicts a fluid wherein only a portion of the particles have glow properties.

Bibliography

[1] R. G. Andrews. Has science solved one of history's greatest adventure mysteries? *National Geographic*, 2021.

[2] A. Babanin. *Breaking and dissipation of ocean surface waves.* Cambridge University Press, 2011.

[3] A. Bogdanoff. Through the looking glass of the sea surface. Woods Hole Oceanographic Institution.

[4] J. U. Brackbill and H. M. Ruppel. Flip: A method for adaptively zoned, particle-in-cell calculations of fluid flows in two dimensions. *Journal of Computational physics*, 65(2):314–343, 1986.

[5] E. Brodl, J. Niederhauser, and P. Macheroux. In situ measurement and correlation of cell density and light emission of bioluminescent bacteria. *JoVE Journal*, 2018.

[6] E. Froemling, T. Goktekin, and D. Peachey. Simulating whitewater rapids in ratatouille. In *ACM SIGGRAPH 2007 Sketches*, SIGGRAPH '07, New York, NY, USA, 2007. Association for Computing Machinery.

[7] P. Janssen. *The Interaction of Ocean Waves and Wind.* Cambridge University Press, 2004.

[8] T. Kim, N. Thurey, D. James, and M. Gross. Wavelet turbulence for fluid simulation. *ACM Trans. Graph.*, 27(3):1–6, Aug. 2008.

[9] M. Latz. A surfer rides a bioluminescent wave from a bloom of l. polyedra. https://cdn.coastalscience.noaa.gov/csmedia/2020/05/surfer-rides-bioluminescent-wave-in-Southern-California-spring-2020-credit-SCRIPPS.jpg.

[10] A. Loera. Bioluminescent organisms turn ocean waves into a neon blue light show on southern california beaches. News Maven.

[11] C. Parmentier. Pipewave flip simulation tutorial. Youtube. https://www.youtube.com/watch?v=FqunirhLNwc.

[12] E. Photography. Big, blue banzai. EPK. https://epkcollection.com/wp-content/uploads/2017/07/Surfline-EPK-Surf-Poster-Big-Blue-Banzai.jpg.

[13] D. Side, V. Nassisi, C. Pennetta, P. Alifano, M. Di Salvo, A. Tala, A. Chechkin, F. Seno, and A. Trovato. Bacterial bioluminescence onset and quenching: a dynamical model for a quorum sensing-ediated property. *Royal Society Open Science*, 4(12), 2017.

[14] S. Smith, N. Brown, and S. Summers. *Toy Story: How Pixar Reinvented the Animated Feature*. Bloomsbury Academic and Professional, New York, 2018.

[15] M. Valiadi, S. C. Painter, J. T. Allen, W. M. Balch, and M. Iglesias-Rodriguez. Molecular detection of bioluminescent dinoflagellates in surface waters of the patagonian shelf during early austral summer 2008. *PLoS One*, 9(6), 06 2014.

[16] R. Wessels. Bioluminescent waves in oceanside, california. Wordpress.