Euphemia Wu
CS 6323.001 Computer Animation and Gaming
Dr. Xiaohu Guo
11/27/2023

Water Dynamics

Section 1. Summary:

I found inspiration in a scholarly paper titled “Generalizing Shallow Water Simulations with Dispersive Surface Waves” available on the ACM website (https://dl.acm.org/doi/10.1145/3592098) that is published in 2023 SIGGRAPH. This paper has been reviewed by many others and is considered a reputable source of information. What really interests me about this paper is its exploration of creating realistic water simulations in 3D animation, a topic that has always fascinated me when I've seen it in games and movies. I am enthusiastic about creating my own version of lifelike water waves after seeing the inspirations in movies and video games. My plan involves implementing water dynamics and physics, utilizing formulas for sine waves and amplitude to achieve the desired effects. Additionally, I intend to add different kinds of objects that will float on the water's surface, enhancing the realism of the simulation. I will also add shader graphs in unity for the water surface to look more realistic.

Section 2. Description of Work:

There are two parts to this water dynamics project. The first part is about showing the objects floating on the water, and the second part is rendering the wave of water so that the water surface looks more realistic.

Section 2.1 Water Floating Section:

In the Water Floating part, it shows the floating cube objects on the surface as well as a floating rectangle (resembles a boat) that are implemented using two different ways of floating effect.

Section 2.1.1 Floating Cube:

I first added a plane unity object onto the screen and used that as the water plane. I then added a cube object on top of the water plane and added script to make the floating effect. The script FloatingCube.cs is implemented in the language of C# to work within Unity. When each object is initialized, it's given a random offset for added variety so that each cube object behaves differently when floating. During every FixedUpdate frame, the script calculates a factor based on the object's vertical position relative to a specified water level. It ensures that the buoyant force is stronger when the object is submerged deeper in the water and decreases as the object approaches its maximum float height. This factor is then used to compute an uplift force, applied at a specific action point slightly away from the object's center. This force counteracts gravity, causing the object to appear as if it's floating and moving on the water. To control the bounce of objects upon entering the water, the script integrates a damping factor, contributing to a more realistic simulation of floating cubes in the Unity scene.

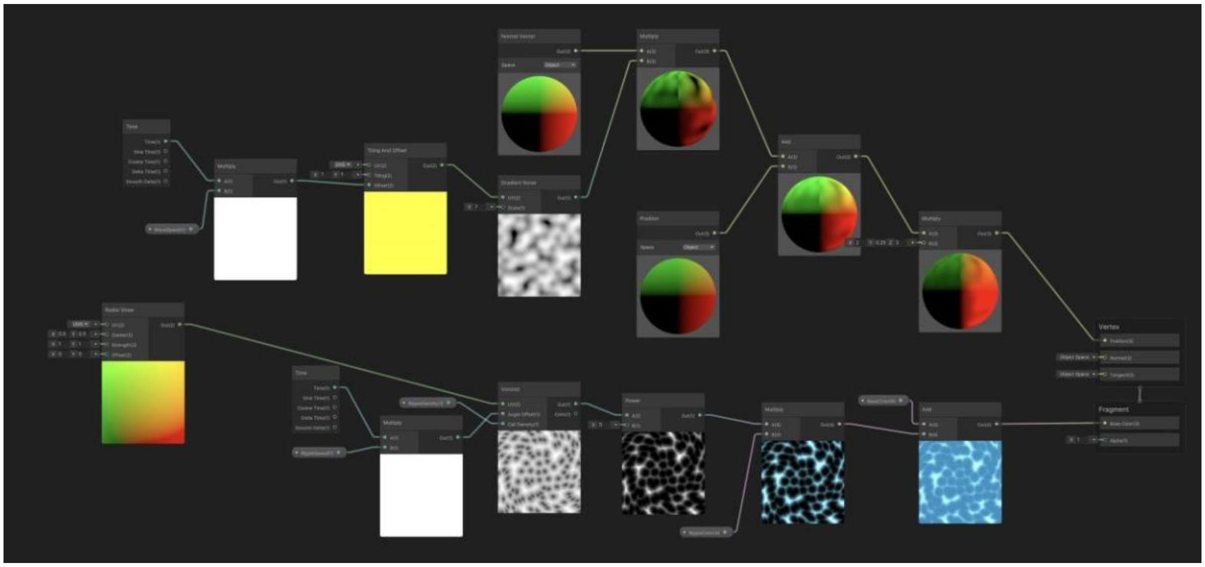
Section 2.1.2 Floating Boat:

To create the illusion of a floating boat, I used a rectangular object designed specifically for this purpose, deviating from the previously mentioned cube. The four corners of the rectangle act as floaters, responding to physical forces like buoyancy and drag. This design choice allows the rectangle to convincingly mimic the natural motion of a boat gently swaying on the water's surface. Unlike cube objects, the boat doesn't drift with the water's flow but stays within the screen, swaying in sync with the simulated wave movement. This approach aims to offer a more realistic portrayal of a boat on water, enhancing visual appeal and creating a captivating simulation.

This Unity script FloatingBoat.cs is intended to replicate the buoyancy and floating characteristics of the boat within a virtual environment. The boat's buoyancy relies on a series of Transform objects representing floaters. For each floater, the script calculates the difference between its position and a specified water height. If a floater is positioned below this water height, I apply a buoyant force to the Rigidbody at that floater's position, mimicking the upward force exerted by water. The script dynamically adjusts the boat's drag and angular drag properties based on whether any floaters are underwater, allowing it to switch between underwater and above-water states. The boat's floating behavior is influenced by parameters like UnderWaterDrag, UnderWaterAngularDrag, AirDrag, AirAngularDrag, FloatingPower, and WaterHeight. I've utilized the FixedUpdate method for physics-related updates, and the SwitchState method handles toggling the boat's drag properties based on its underwater status.

Section 2.2 Water Shader:

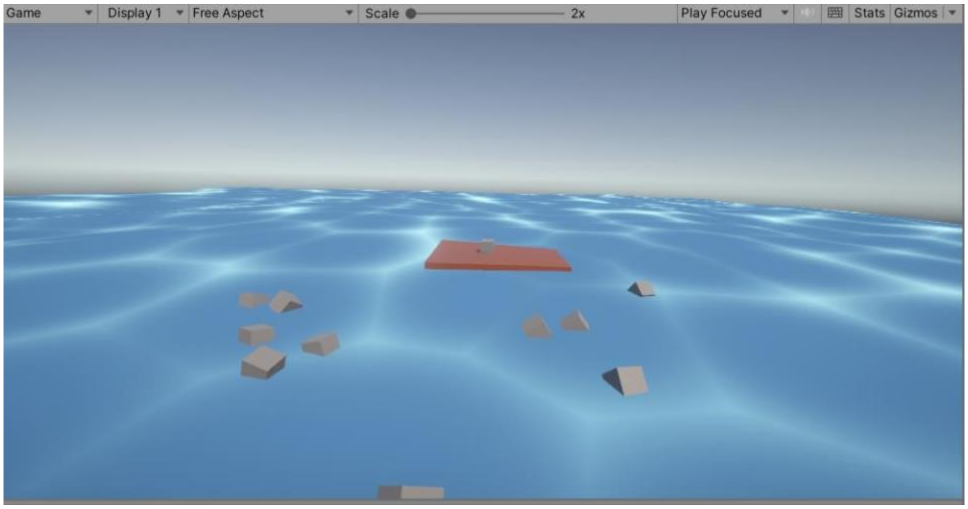
The primary goal of the shader is to enhance the realism of the water. Initially, the water surface, represented by a plain object in Unity, lacked visual depth and failed to resemble authentic water. Leveraging Unity's lit shader graph, I devised a Voronoi component. This component, characterized by circles and dots, effectively mimics the patterns found on the surface of water. To heighten realism, I multiplied it with a power component. The base color was modified to a light blue shade, creating a visual effect reminiscent of shallow water. Incorporating a Time component ensures that the Voronoi effect on the water surface dynamically moves in both directions, simulating the natural flow of water. Additionally, a Gradient Noise component was integrated to further augment the overall realism of the water shader. The complete shader graph of the water can be seen below:



Section 3. Results:

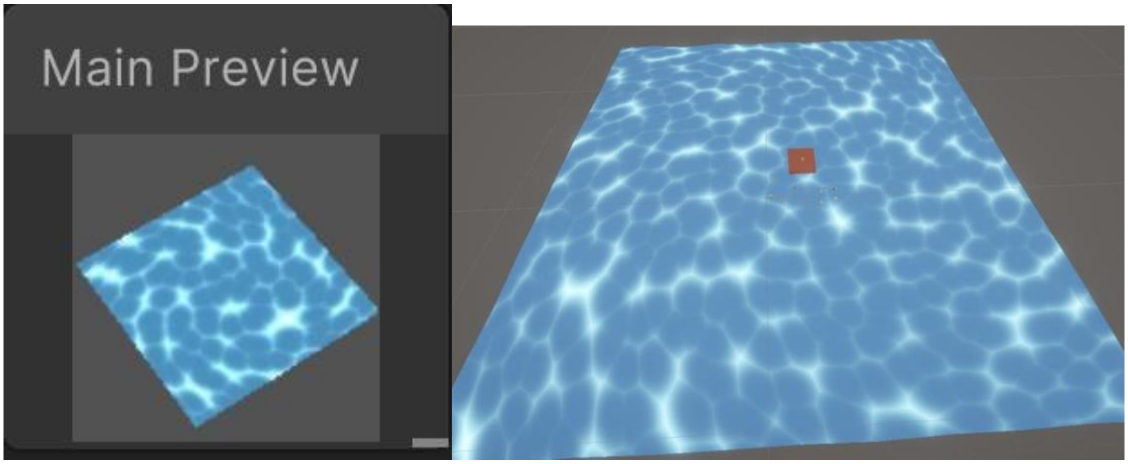
Section 3.1 Water Floating Result:

As we can see clearly in the picture, both the cubes and the boat (red rectangle) float on the water nicely. The cube objects flow with the water once they get released from the sky (please see the attached video demo to see the animation) and gradually drift away from the center. The boat, on the other hand, stays put but sways with the water's waves.



Section 3.2 Shader Result:

In the depicted image, the water does look realistic. Noticeable wave patterns gently traverse its surface, and the light blue hue convincingly mimics the visual charm of shallow water. Similarly, in the provided video, the wave pattern dynamically mirrors the natural play of light on a water surface, capturing the nuances created by sunlight at varying angles and positions.



Section 4. Analysis of Work:

I completed the task successfully. I made sure the cubes and the boat look like they're floating realistically on the water by creating a shader that mimics the water's movement. I achieved the main goals, showing that the objects interact convincingly with the water. However, after some thought, I see room for improvement. Right now, the simulation of waves relies solely on Unity's shader graph without specific math formulas. Specifically, the gradient noise node in the shader graph is used to make he water move up and down, creating a waving effect. The objects' movements are random, following offset values in the algorithm, making them flow naturally with the simulated water. Looking ahead, a good improvement would be to refine how objects interact with the waves. Using specific formulas could sync their movements with the waves, making the simulation even more realistic. This would enhance the overall representation and contribute to a more immersive experience.

In summary, while the current setup met the main goals, recognizing and addressing potential improvements, like refining how objects move with waves, could make the simulation more realistic. This ongoing process mirrors the dynamic nature of computer graphics development, always striving for better authenticity and visual appeal.

Link to the source code and video demo:

https://utdallas.box.com/s/cquc70h9pqzdb7l3f0pu31mb9ek8qcmc