3D Ocean Wave Simulation

By Himanshu Chablani

# Index

1. Introduction

2. Theory

3. Implementation

4. Conclusion

# Introduction

In this project I demonstrate real time water simulation using C++ and Octet framework. For the purpose of this project I have used Gerstner wave theory to simulate deep ocean waves. I this project I explain how the amplitude of wave, phase, and wave vector are related. The user can alter any of these parameters at run time and the ocean surface will change instantly, the user can also run the animation for a set number of frames. The parameters for the wave are saved in a configuration file which can be altered by the user at run time. The user can store as many ocean surface as he wants and they can be loaded in and simulated at runtime. The user can view the wave as wireframe, solid grey mesh without updated normals, solid grey mesh with updated normals, solid coloured mesh without updated normals, or solid coloured mesh with updated normals. My project uses set of eight waves to create the ocean surface. A good wave profile can be achieved by setting certain set of values for the parameters.

# Theory

For the purpose of this project I have used Gerstner wave theory model to calculate the position of any vertex at a given time. In this method we start off by calculating the **Omega** or the angular frequency for the wave from the period **T**, which is the interval of passage of a successive crests through a fixed point. Let **lambda** be the wavelength, which is the distance between two successive crests. Hence, the phase speed **C** = **Lambda**/**T**. As we know **C** = **Omega**/**Wave Number**(**Wave Number** is 2\***Pi**/**Lambda**), hence **Omega** = **C**\***Wave Number**, which gives **Omega**=2\***Pi**/**T**. Deep ocean waves are wave depended and are a ratio of water depth and the wave length **lambda[[1]](#footnote-2)**.

**lambda**=(acceleration due to gravity(**g**)\***T**2 / 2\***PI**)\*tanh(2\***PI**\* water depth(**d**)/**lambda**)

Since tanh(x) = 1, when x >**Pi**,hence for **d**/**lambda** > **Pi** i.e. deep water :

**lambda**=**g**\***T**2 / 2\***PI**

As **C** = **Lambda**/**T**, **C**  for deep water = **g**\***T** / 2\***PI**,

**Wave Number** = 2\***Pi**/**Lambda** = 4\***Pi**2/ **g**\***T**2,

**Omega**  = 2\***PI**/**T**

but we cannot use this relation for deep ocean waves as we want to base **Omega** from deep water **lambda.**

hence we use the equation **C**2 = (**g**/ **Omega**)2.

which gives us **Omega** = (**g\* Wave Number**)1/2

In 3D **Wave Number** is a 2D vector which represents the wave.

Let **Y** be surface elevation at a given point and time and **X0** be the vector point on the horizontal plane.

**Y**= Amplitude of the wave(**A**)\*Cos(**Wave Number.X0** - **Omega\***current time(**t**))

When the steepness of the wave increases, its crests become sharper and its troughs flatter. therefore I have used a more realistic description bases on Trochoids.

**X**  = **X0** - (**Wave Number**/|| **Wave Number** ||)\***A**\*Sin(**Wave Number.X0** - **Omega\*t**)

Extending this, a number of N such wave profiles can be generated from a set of **Wave Number Ki**, Amplitude **Ai** , and phase **Phii,**, giving us the equation :

**Y**= ∑[**Ai** \*Cos(**Ki.X0** - **Omega\* t** + **Phii** )]

**X**  = **X0** -∑ [(**Ki** /|| **Ki** ||)\* **Ai** \*Sin(**Ki.X0** - **Omega\*t** + **Phii**)]

1. https://hal.inria.fr/file/index/docid/307938/filename/frechot\_realistic\_simulation\_of\_ocean\_surface\_using\_wave\_spectra.pdf [↑](#footnote-ref-2)