國立交通大學

多媒體工程研究所

碩 士 論 文

海嘯波之傳遞模擬

The Simulation of Tsunami Wave Propagation

研 究 生：郭彥廷

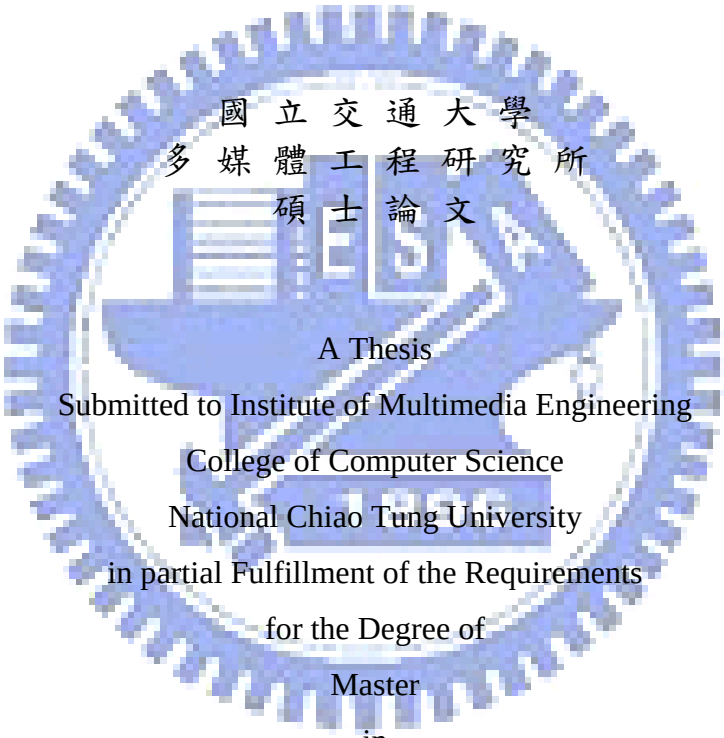
指導教授：施仁忠　教授

中 華 民 國　九 十 六　年 六 月

海嘯波之傳遞模擬
The Simulation of Tsunami Wave Propagation

研 究 生：郭彥廷　　Student：Yen-Ting Kuo

指導教授：施仁忠　　Advisor：Zeng-Chun Shih



國 立 交 通 大 學

多 媒 體 工 程 研 究 所

碩 士 論 文

A Thesis

Submitted to Institute of Multimedia Engineering

College of Computer Science

National Chiao Tung University

in partial Fulfillment of the Requirements

for the Degree of

Master

in

Computer Science

June 2007

Hsinchu, Taiwan, Republic of China

中華民國九十六年六月

海嘯波之傳遞模擬

研究生：郭彥廷　　指導教授：施仁忠 教授

國立交通大學多媒體工程研究所

摘　　要

二零零四年的南亞大海嘯造成相當慘重的人員傷亡與經濟損失，透過電視媒體放送海嘯沖上陸地的影像也使人類對真正的海嘯模樣有所認識。然而海嘯的波動並非皆巨浪滔天有如電影般誇張，有感於海洋工程方面對海嘯模擬多半只有海嘯波面的模擬，而缺少真實景觀伴襯；而在電腦圖學方面的海洋流體模擬多為模擬海洋表面的小幅度波動，極少關於海嘯運動的模擬。本篇論文希望結合兩種領域的優點，透過有具物理意義的淺水波方程式來達到模擬真實海嘯波的形成與傳遞，再透過電腦圖學的技術使得海面景象能較逼真，給予一般大眾瞭解海嘯波的運動以及作為日後研究海嘯運動的參考。

The Simulation of Tsunami Wave Propagation

Student: Yen-Ting Kuo Advisor: Dr. Zen-Chung Shih

Institute of Multimedia Engineering
National Chiao Tung University

ABSTRACT

The tsunami which happened in 2004 has made lots of damages, and people saw the video from TV, they finally know how terrible the tsunami is. However, the movements of tsunami waves are not like those which are shown in Hollywood movies. In ocean engineering, they only simulate the surface of the tsunami wave movements, and the simulations are not rendered in a realistic way. In computer graphics, the ocean simulations often show the results which focus on the small movement on the ocean surface. In this thesis, we want to combine advantages in these two domains. We hope that we can simulate the tsunami wave propagation by physically-correct equations developed in ocean engineering and we render the simulated scene by the techniques developed in computer graphics to make it realistic. And we hope that our simulation system can make people having know more understanding about tsunami waves more and the system can be referenced by whom wants to study wave motions of tsunami some other days.

Acknowledgements

First of all, I would like to thank to my advisor, Dr. Zen-Chung Shih, for his supervision and help in this work. And, I want to thank for all the members in Computer Graphics & Virtual Reality Lab for their comments and instructions. Finally, special thanks for my family and my dear friends in the past two years, and this achievement of this work dedicated to them.

Contents

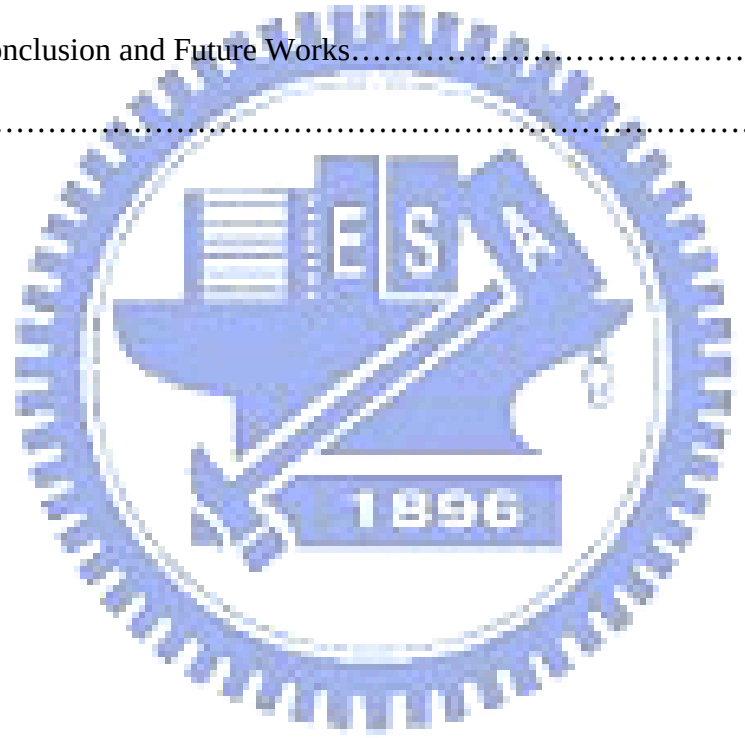


Figure List

Chapter 1
Introduction

1.1 Motivation

The ***tsunami*** happened in south Asia in 2004 has made 200,000 people dead and 1.5 million people homeless. People were also shocked by the power of tsunami waves. But when we saw the video broadcasted on TV, we find out that the real tsunami waves are not like what we often see in the Hollywood movies. Real tsunami waves are usually only several meters high.

In ocean engineering, causes of Tsunami, energy propagation of tsunami and wave movement have been researched for a long time, but the simulated image of the result of wave propagation is really dull, as Shown in Figure 1.1.

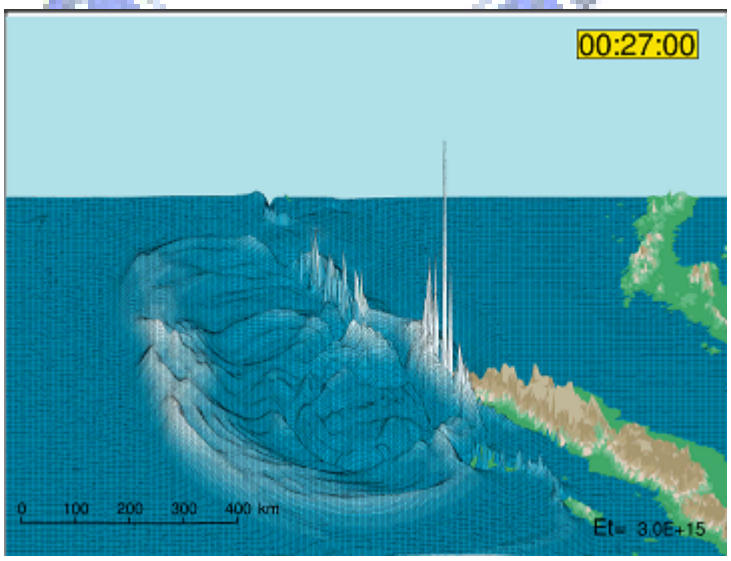


Figure 1.1: The simulation of tsunami surface.

The simulated result only shows the tsunami surface. It is different from what

the ocean surface should be in the real world. This makes it hard to let the masses know the real power of tsunami. In computer graphics, researches about fluid simulation have been quite a lot, but most of the simulations focus on liquid behavior in small region. Though there are some researches about ocean simulation, they just simulate light fluctuation on ocean surface.

The goal of this thesis is to make a combination of advantages of two domains mentioned above. Using ***shallow water equation[1]*** which is a physical equation that governs the fluid behavior well to calculate the height of water surface and simulate the propagation of waves. Then we render the scene by using techniques in computer graphics to make it vivid. We hope that our simulation results can make people having more understanding of tsunami waves and these works can be referenced by whom wants to study wave motions of tsunami some other days.

1.2 Introduction to Tsunami

Tsunami is a kind of wave with great destructive power. When earthquake happens under the bottom of the sea, water undulates violently by the motive force caused by seismic waves, and water becomes strong waves that push ahead to the coast.

Tsunami is usually caused by earthquake which happens at 50 kilometers below the seabed in over 6.5 Richter magnitude scale. The wave length of a tsunami is larger than the deepest depth of the ocean. There is no block in the ocean so no matter how deep the ocean is energy will propagate through. The period of tsunami is larger than four minutes, and the velocity of tsunami is about 500 kilometers to 1000 kilometers per hour.

After tsunami waves pass through continental shelf, the height of waves will increase in very high speed because the water depth becomes smaller. The amplitude

might be several meters high, and becomes a shockwave. Wave motion on ocean surface is quite different from it caused by earthquakes. The former undulates only within a certain depth below the sea surface but the later undulates from the seabed to the sea surface.

1.3 Thesis Organization

The following chapters are organized as follows. In Chapter 2, we will introduce two mainstream directions of fluid simulation in computer graphics, one is small volume liquid simulation and the other is ocean simulation. In Chapter 3, we introduce our proposed tsunami model and how we simulate the propagation of tsunami. Then we introduce how to generate light reflection on the ocean surface. The implementation and results will be demonstrated in Chapter 4. Finally, Conclusion and future works will be given in Chapter 5.

Chapter 2
Related Works

2.1 Fluid Simulation

Currently, the simulation of complex water effects is often based on three-dimensional *Navier-Stokes [1]* equations. Through 3D Navier-Stokes equations, we can simulate churn, splash and sprinkle in the 3D space in detail. With advanced physically-based rendering system, we can simulate a really vivid scene. But the drawback of methods mentioned above is computational-consuming. It is hard to simulate in real-time. Foster and Fedkiw [2] made significant contributions to the simulation and control of 3D fluid simulations through the introduction of a hybrid liquid volume model combining implicit surfaces and mass-less marker particles.

Considering the simulation of wave motions of large water volume by using 3D Navier-Stokes equations, in order to reduce the computational cost, hybrid models are popular. Losasso et al. [7] introduced a way that coarsens away from the water surface with an octree data structure. But this doesn't make full use of the highly accurate MAC grids and numerical dissipation will increase. Besides large octree cell can not represent bottom topography. There are also methods based on height fields such as 2D shallow water equations [8]. Though this approach can capture effects of complex bottom topography rather well, it does not support overturning or other 3D behaviors. Irving and Guendelman [5] proposed a water volume model which makes a combination of uniform grids and tall cells. In uniform grids near the surface, the

advection is calculated by 3D Navier-Stokes equations; in tall cells, advection equations are simplified. Pressures and velocities are interpolated within the cells. This hybrid method can keep details of the water surface and reduce the computation cost under water surface.

2.2 Ocean Simulation

The classic reference on ocean waves rendering is the work of [3]. Fournier and Reeves presented a simple model for the surface of the ocean. It is based on the Gerstner model where particles of water describe circular or elliptical stationary orbits and the foam generated by the breaker is modeled by particle systems. The wave shape is modeled as follows,

$$x = x_0 + r * \sin(\kappa x_0 - \omega t) \quad (1a)$$

$$z = z_0 - r * \cos(\kappa x_0 - \omega t) \quad (1b)$$

where t is the time, z is the vertical axis and (x_0, z_0) is the particle location at rest, r is a distance from the center of a circle of radius $\frac{1}{\kappa}$ rolling over a line. Figure 2 shows the trochoid of Gerstner wave model. The surface is modeled as a parametric surface and can be rendered by traditional rendering methods. By combining more sine, cosine functions, the wave surface will be less regular.

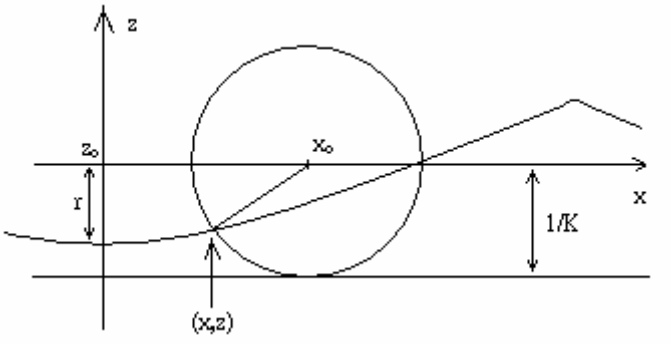


Figure 2.1: The trochoid of Gerstner wave model

Gonzato and Bertrand [4] proposed a complete geometrical model of the ocean waves accounting for refraction, diffraction, reflection, transmission and multi-wave

trains. Then they rendered the scene by using a particular illumination model and displacement mapping texture to deal with multi-wave trains. It makes physically-realistic result but not in real-time.

However oceanographers do not take Gerstner wave as a realistic model of the ocean. Instead, they introduce statistical models which combine experiment observations. In statistical model, the wave height is formulated as a function $f(x,t)$, where x is the horizontal position and t is the time. Jason L. Mitchell [6] present a multi-band Fourier domain approach to synthesize the ocean surface and rendering the waves on GPU entirely. The height of the water at location x at time t is :

$$h(x,t) = \sum_k \tilde{h}(k,t) * e^{ikx} \tag{2}$$

where k is a 2D vector with components (k_x, k_y), $k_x = 2\pi m / L_x, k_y = 2\pi m / L_y$, and n and m are integers with bounds –N/2 < n < N/2 and –M/2 < m < M/2. Then IFFT (Inverse Fast Fourier Transform) will generate a height field at discrete point $x = (nL_x / N, mL_y / M)$, and we can finally construct ocean water.

Yaohua Hu et al. [12] presented a system that can render calm ocean wavers with sophisticated lighting effects at a high frame rate. They construct the wave geometry as a displacement map with surface detailed by a dynamic bump map. Xudong Yang et al. [11] presented a multi-resolution mesh model of the ocean surface based on a straightforward terrain level of detail scheme, Tiled Quad-tree. By this scheme, the ocean surface can be extended limitless and accelerated by GPU.

Chapter 3
Tsunami Wave Model

In the following sections, we will describe how we construct the tsunami wave model and render the complete ocean scene. In section 3.1, we describe the basic equations for our simulation system. Then we demonstrate how we obtain the linear system from equations we described above in section 3.2. In section 3.3, we introduce a numerical method to solve the linear system. Ocean surface construction is described in section 3.4. Finally, we show how we render the ocean scene in section 3.5.

3.1 Shallow Water Equation

Two-dimensional depth-averaged equation is generally called shallow water equation (SWE) which is integrated from 3-D Navier-Stokes equations. The depth is integrated from the bottom to the top surface. The SWE describes the evolution of a hydrostatic homogeneous, incompressible flow on the surface of the sphere. This equation is accurate when the ratio of the vertical scale to the ratio of the horizontal scale is small. SWE can be used in atmospheric and oceanic modeling, but are much simple than its primitive equations (Navier-Stokes equations).

3.1.1 Derivation of Shallow Water Equation

Assuming a column of fluid of height h, and a base area A. Then the total mass of fluid at the base over the area A is

$$m = hA\rho, \tag{3}$$

and it corresponds to a force in the direction of gravity such as

$$f = mg. \tag{4}$$

Then we can find out that the pressure at the base area is therefore

$$p = f / A = hA\rho g / A = h\rho g . \tag{5}$$

Hence we can rewrite the pressure term in (5) into

$$\nabla p = \nabla h\rho g . \tag{6}$$

In large-scale motion of the fluid, “shallow” means that there is only little variation in *y* direction, so the vertical acceleration can be ignored. Now the fluid is said to be in hydrostatic equilibrium, and it implies that the Newton’s second law of motion can be simplified as

$$\rho(\frac{\partial}{\partial t} + U \cdot \nabla)U = -g\rho\nabla h + \mu\nabla^2 U + Fe , \tag{7}$$

where F_e is the external forces, $U = [u, v, w]^T$, and the changes in the horizontal plane are nonzero. So we can reduce (7) to the following two equations:

$$\rho(\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} + v\frac{\partial u}{\partial y}) = -g\rho\frac{\partial h}{\partial x} + \mu(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}) + Fe1 , \tag{8}$$

$$\rho(\frac{\partial v}{\partial t} + u\frac{\partial v}{\partial x} + v\frac{\partial v}{\partial y}) = -g\rho\frac{\partial h}{\partial y} + \mu(\frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial y^2}) + Fe2 . \tag{9}$$

The mass preservation in terms of the height is found by make an integration on *z*, so

$$\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} + \frac{\partial w}{\partial z} = 0 , \tag{10}$$

$$\downarrow$$

$$\int_0^h \frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} + \frac{\partial w}{\partial z} dz = (\frac{\partial u}{\partial x} + \frac{\partial u}{\partial y})h + w_h - w_0 = 0 . \tag{11}$$

Since w_0 is the vertical speed at the bottom, and w_h is the rate of rise on the fluid surface, hence

$$w_0 = 0 , \tag{12}$$

$$w_h = \frac{dh}{dt} . \tag{13}$$

Then insert (12), (13) into (11), we can find that

$$(\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y})h = -\frac{dh}{dt} = -\frac{\partial h}{\partial t} - u\frac{\partial h}{\partial x} - v\frac{\partial h}{\partial y} , \tag{14}$$

$$\downarrow$$

$$(\frac{\partial u}{\partial x}+\frac{\partial v}{\partial y})h+\frac{\partial h}{\partial t}+u\frac{\partial h}{\partial x}+v\frac{\partial h}{\partial y}=0. \quad (15)$$

Then the SWE can be obtained by using $F_e=[0,0,g]^T$, and assuming inviscid fluid, that is $\left\|\mu\nabla^2 U\right\|_2 << \left\|\nabla p\right\|_2$, hence

$$\frac{\partial u}{\partial t}+u\frac{\partial u}{\partial x}+v\frac{\partial u}{\partial y}=-g\frac{\partial h}{\partial x} \quad (16a)$$

$$\frac{\partial v}{\partial t}+u\frac{\partial v}{\partial x}+v\frac{\partial v}{\partial y}=-g\frac{\partial h}{\partial y} \quad (16b)$$

$$\frac{\partial h}{\partial t}+u\frac{\partial h}{\partial x}+v\frac{\partial h}{\partial y}=-(\frac{\partial u}{\partial x}+\frac{\partial v}{\partial y})h \quad (16c)$$

3.2 Implicit Semi-Lagrangian Time Integration Method

3.2.1 Semi-Lagrangian Method

The semi-Lagrangian method can be viewed as a combination of Eulerian method and Lagrangian method. In the Eulerian method, we describe one kind of characteristics of the fluid at a fixed point in the world space by a function E(x,y,z,t), and we observe the world evolving around the fixed point. In the Lagrangian method, we follow a particle moving along its trajectory, so the physical factors which at certain point X in the world space at time T are only affected by their initial positions and the time variable T.

In the semi-Lagrangian scheme, at every time step the grid points of the numerical mesh are representing the arrival points of backward trajectories at the future time steps. The point reached during this backward tracking defines where a small space was at the beginning of the time step. The particle is subjected to different physical forces during its traveling. All predictive variables on this departure point are then found by interpolating the values on the grids at the previous time steps, as

shown in Figure 3.1.

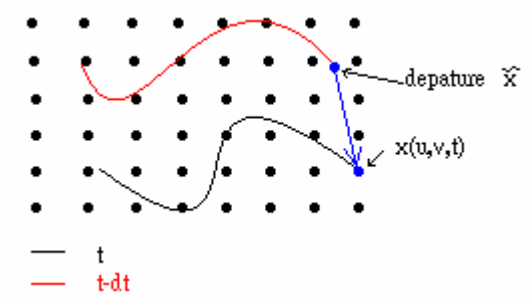


Figure 3.1: The Lagrangina derivation is approximated along particle trajectories

The advantage of semi-Lagrangian method is that it allows the use of large time step without limiting the stability and it is superior in performance. The limitations for stability are that trajectories of fluid particles should not cross and not overtake another. Hence, the choice of time step in the semi-Lagrangian method is only limited by numerical accuracy.

3.2.2 Applying semi-Lagrangian method to SWE

To show how the SWE can be integrated by the semi-Lagrangian method, we first consider equations (16) and then we replace the substantial derivates in (16) with ordinary derivates

$$\frac{du}{dt} = -g\frac{\partial h}{\partial x}, \tag{17a}$$

$$\frac{dv}{dt} = -g\frac{\partial h}{\partial x}, \tag{17b}$$

$$\frac{dh}{dt} = -(\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y})h. \tag{17c}$$

Then, by using forward differences, we can estimate the ordinary derivates by tracking a particle at grid point (i, j),

$$\frac{du}{dt} = \frac{u(t+\Delta t) - \tilde{u}(t)}{\Delta t}, \tag{18a}$$

$$\frac{dv}{dt} = \frac{v(t+\Delta t) - \tilde{v}(t)}{\Delta t}, \tag{18b}$$

$$\frac{dh}{dt} = \frac{h(t+\Delta t) - \tilde{h}(t)}{\Delta t}. \tag{18c}$$

Functions $\tilde{u}$, $\tilde{v}$, $\tilde{h}$ have to be estimated by using the current values of *u*, *v*, and *h*, at grid point (*i*, *j*), based on an assumption that these current values are also good approximated results in the previous steps. The departure point of hypothetical particle at (*i*, *j*) can be estimated as follows :

$$\begin{bmatrix} \tilde{x}(t-\Delta t) \\ \tilde{y}(t-\Delta t) \end{bmatrix} = \begin{bmatrix} x(t) \\ y(t) \end{bmatrix} - \Delta t \begin{bmatrix} u(t) \\ v(t) \end{bmatrix}, \tag{19}$$

and then we can estimate the value of $\tilde{u}$, $\tilde{v}$, and $\tilde{h}$ by bilinear interpolation,

$$\tilde{u} = (1-a)(1-b)u(\lfloor \tilde{x} \rfloor, \lfloor \tilde{y} \rfloor) + a(1-b)u(\lceil \tilde{x} \rceil, \lfloor \tilde{y} \rfloor) + abu(\lceil \tilde{x} \rceil, \lceil \tilde{y} \rceil) + (1-a)bu(\lfloor \tilde{x} \rfloor, \lceil \tilde{y} \rceil), \tag{20}$$

where $a = \tilde{x} - \lfloor \tilde{x} \rfloor$, $b = \tilde{y} - \lfloor \tilde{y} \rfloor$. Then $\tilde{v}$ and $\tilde{h}$ are also similar. Now we discretize (17) and combine with (18), then we obtain,

$$\frac{u(t+\Delta t) - \tilde{u}(t)}{\Delta t} = -g\frac{\partial h(t+\Delta t)}{\partial x}, \tag{21a}$$

$$\frac{v(t+\Delta t) - \tilde{v}(t)}{\Delta t} = -g\frac{\partial h(t+\Delta t)}{\partial y}, \tag{21b}$$

$$\frac{h(t+\Delta t) - \tilde{h}(t)}{\Delta t} = -(\frac{\partial u(t+\Delta t)}{\partial x} + \frac{\partial v(t+\Delta t)}{\partial y})h(t), \tag{21c}$$

Then we calculate the derivative of (21a) with respect to *x* and (21b) with respect to *y*, we obtain,

$$\frac{\partial u(t+\Delta t)/\partial x - \partial \tilde{u}(t)/\partial x}{\Delta t} = -g\frac{\partial^2 h(t+\Delta t)}{\partial x^2}, \tag{22a}$$

$$\frac{\partial v(t+\Delta t)/\partial y - \partial \tilde{v}(t)/\partial y}{\Delta t} = -g\frac{\partial^2 h(t+\Delta t)}{\partial y^2}. \tag{22b}$$

Next we can eliminate $u(t+\Delta t)$, $v(t+\Delta t)$ in (21c) by using (22a) and (22b) as,

$$\frac{h(t+\Delta t)-\tilde{h}(t)}{\Delta t}=-(\frac{\partial\tilde{u}(t)}{\partial x}-\Delta tg\frac{\partial^2 h(t+\Delta t)}{\partial x^2}+\frac{\partial\tilde{v}(t)}{\partial y}-\Delta tg\frac{\partial^2 h(t+\Delta t)}{\partial y^2})h(t)\,, \qquad (23)$$

after rearranging the terms,

$$h(t+\Delta t)=\tilde{h}(t)-\Delta th(t)(\frac{\partial\tilde{u}(t)}{\partial x}-\Delta tg\frac{\partial^2 h(t+\Delta t)}{\partial x^2}+\frac{\partial\tilde{v}(t)}{\partial y}-\Delta tg\frac{\partial^2 h(t+\Delta t)}{\partial y^2})\,. \qquad (24)$$

Then we apply first order and second order central differences for the spatial derivates, we can get the following equation,

$$\begin{aligned}h_{i,j}(t+\Delta t)&=\tilde{h}_{i,j}(t)-\Delta th_{i,j}(t)\frac{\partial\tilde{u}(t)}{\partial x}+(\Delta t)^2 h_{i,j}(t)g\frac{\partial^2 h_{i,j}(t+\Delta t)}{\partial x^2}\\&\quad-\Delta th_{i,j}(t)\frac{\partial\tilde{v}(t)}{\partial y}+(\Delta t)^2 h_{i,j}(t)g\frac{\partial^2 h_{i,j}(t+\Delta t)}{\partial y^2}\\&=\tilde{h}_{i,j}(t)-\Delta th_{i,j}(t)(\frac{\tilde{u}_{i+1,j}(t)-\tilde{u}_{i-1,j}(t)}{2\Delta x})+(\Delta t)^2 h_{i,j}(t)g(\frac{h_{i+1,j}(t+\Delta t)-2h_{i,j}(t+\Delta t)+h_{i-1,j}(t+\Delta t)}{\Delta x^2})\\&\quad-\Delta th_{i,j}(t)(\frac{\tilde{v}_{i,j+1}(t)-\tilde{v}_{i,j-1}(t)}{2\Delta y})+(\Delta t)^2 h_{i,j}(t)g(\frac{h_{i,j+1}(t+\Delta t)-2h_{i,j}(t+\Delta t)+h_{i,j-1}(t+\Delta t)}{\Delta y^2})\end{aligned}$$

. (25)

Finally, we isolate terms with $h(t+\Delta t)$,

$$\begin{aligned}&h_{i,j}(t+\Delta t)-(\Delta t)^2 h_{i,j}(t)g(\frac{h_{i+1,j}(t+\Delta t)-2h_{i,j}(t+\Delta t)+h_{i-1,j}(t+\Delta t)}{\Delta x^2})\\&\quad-(\Delta t)^2 h_{i,j}(t)g(\frac{h_{i,j+1}(t+\Delta t)-2h_{i,j}(t+\Delta t)+h_{i,j-1}(t+\Delta t)}{\Delta y^2})\,, \qquad (26)\\&=\tilde{h}_{i,j}(t)-\Delta th_{i,j}(t)(\frac{\tilde{u}_{i+1,j}(t)-\tilde{u}_{i-1,j}(t)}{2\Delta x})-\Delta th_{i,j}(t)(\frac{\tilde{v}_{i,j+1}(t)-\tilde{v}_{i,j-1}(t)}{2\Delta y})\end{aligned}$$

and we can assume that $h(t)$ on the left-hand side is a constant, so we can solve this system as a linear system for $h(t+\Delta t)$ by using conjugate gradient method.

The algorithm is shown in Figure 3.2:

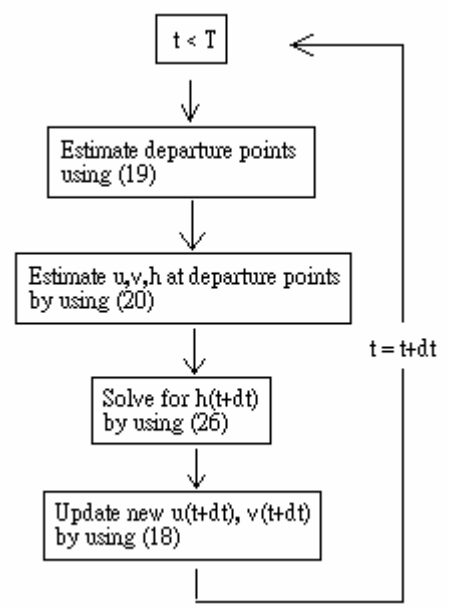


Figure 3.2: Algorithm for the SWE

3.3 Conjugated Gradient Method

In mathematics, the conjugated gradient method is an algorithm for the numerical solution of particular system of linear equations, namely those whose matrix is symmetric and positive definite. The method proceeds by generating vector sequences of iterates, residuals corresponding to iterates, and searching directions used in updating iterates and residuals. It is an algorithm for finding the nearest local minimum of a function of *n* variables. The idea of this algorithm is that the convergence to the solution could be accelerated if we minimize a function *P* over the hyperplane that contains all previous search directions, instead of minimizing *P* over just the line points down gradient. There are several attractive features of conjugated gradient method：First, under the premise that we regard numerical round-off, we can obtain acceptable accuracy of convergence within N iterations. Second, applying different preconditioning schemes, the rate of convergence can be improved greatly. Third, users do not have to choose parameters experimentally.

Conjugated gradient method can be viewed as an improvement of steepest descent method [1]. Now suppose we want to solve the following system of linear

equations,

$$AX = b, \tag{27}$$

where *A* is an *n* by *n* symmetric and positive definite matrix and *A* is real. We denote the unique solution of this system is $X^*, X^* \in R^N$. Without loss of generality, $X_0 = 0$ is the initial guess for X^*. Suppose that D_k is a sequence of *n* mutually conjugate direction. Then D_k forms a basis of R^N. Then we can expand the solution X^* in this basis,

$$X^* = \alpha_1 d_1 + \alpha_2 d_2 + ... + \alpha_n d_n. \tag{28}$$

Let (U,V) demonstrates an inner product of *U* and *V*. To solve equation (27), we find the minimum value of this following quadric equation (29),

$$q(X) = \frac{1}{2}(X, AX) - (b, X). \tag{29}$$

This suggests that we can take the first basis vector D_1 to be the gradient of $q(X)$ at $X = X_0$, which means $D_1 = -b$, and the other vectors in the basis will be conjugate to the gradient. Now let r_k be the residual at the *k*th step,

$$r_k = b - AX_k. \tag{30}$$

We note that r_k is the negative gradient of $q(X)$, so the gradient decent method would take r_k as the moving direction. But here we insist that the direction D_k need to be conjugate to each other, so we choose the direction closest to r_k under the conjugate constraint. Then it gives the following equation (31),

$$D_{k+1} = r_k - \frac{(D_{k,} Ar_k)}{(D_k, AD_k)} D_{k,}, \tag{31}$$

and the final resulting algorithm m is shown in Figure 3.3.

Step 1.

$X_0 = X_{guess}$ // *initailiza tion*

$i = 0$ // *Number of iteration*

Im *ax* // max *imum number of iteration to be performed*

$r_0 = AX_0 - b$

$D_0 = r_0$

Step 2.

if $(i <$ Im $ax)$

Goto Step 3

if $(i =$ Im $ax)$

Goto Step 6

Step 3.

$$\alpha_i = \frac{(r_i, d_i)}{(d_i, Ad_i)}$$

$X_{i+1} = X_i + \alpha_i \cdot d_i$

$r_{i+1} = r_i - (\alpha_i, Ad_i)$

Step 4.

if (r_{i+1} *is sufficient ly small*)

Goto Step 6

Step 5.

$$\beta_k = \frac{(r_{i+1}, r_{i+1})}{(r_i, r_i)}$$

$d_{i+1} = r_{i+1} + \beta_k \cdot d_i$

$i = i + 1$

Goto Step 2

Step 6.

The resutl is X_{i+1}

Figure 3.3: The algorithm of conjugate gradient method.

3.4 Ocean Surface Construction

Generally speaking, except using 3D-Navier Stokes equations to simulate ocean surface, most methods for constructing ocean surface are two dimensional methods using height map representing the variation of ocean surface. Though these 2D methods are hard to simulate the splashing or churning behavior of fluid in detail, they can reduce lots of computation in simulating large scale scenes.

We choose a method to produce fractal surfaces by using Perlin noise function [9]. With laying sets of different amplitudes and frequencies of noise function, we can

generate more fractal surface. Equation (32) demonstrates the output height of a fractal surface.

$$fractal\_noise(x) = \sum_{i=0}^{n} \alpha^i \cdot perlin\_noise(2^i \cdot x). \quad (32)$$

Parameter α represents the amplitude. The higher value of α will give a rougher surface. In the other hand the lower value of α will give a smoother surface.

Then we can pre-generate three or more fractal surface height maps with different sets of parameters. We can synthesize these fractal surface maps into a final height map by an interpolation by dividing the phase of cosine function equally. Through this interpolation method, the wave motion may rise and fall much more irregular in comparison with linear interpolation between fractal surface maps.

Equation (33) demonstrates the interpolation method for three fractal surfaces.

$$\begin{aligned} w_1 &= \cos(dt) \\ w_2 &= \cos(\frac{\pi}{3} + dt) \\ w_3 &= \cos(\frac{2\pi}{3} + dt) \\ Final\_surface &= w_1 * f_1 + w_2 * f_2 + w_3 * f_3 \end{aligned}, \quad (33)$$

where t is a time variable, f_1, f_2, and f_3 are fractal surfaces.

3.5 Rendering Ocean Surface

In order to render physically looking ocean surface, we introduce more physically optic reflection model. First of all, we assume that the ocean surface is perfect specular reflector and we know about the relation of reflection and transmission clearly. But under certain circumstances, we can not consider ocean surface as a perfect specular reflector. When the camera is at a large distance to the ocean surface, the reflected sunlight from the waves appears to be spread out and diffuse. Generally speaking, the behavior of light proceeding respected to a surface are split into two components. First, rays of light are reflected above the ocean surface.

Second, rays of light are transmitted though the surface. Here we will discuss how the two components work together.

3.5.1 Reflection

It is well known that in a perfect specular reflection, the incident ray and the reflected ray have the same angle respect to the surface normal. We may build up the following equation to express the reflected ray. In the beginning, let us define some notations: the wave height $h(x,t)$ and horizontal position x, so the point r on the ocean surface in three-dimensional space can be expressed as follows,

$$r(x,t) = \vec{y}h(x,t) + x, \tag{34}$$

$\vec{y}$ is a pointing-up unit vector. At the point r, the normal of the surface can be calculated by its surface slope,

$$s(x,t) \equiv \nabla h(x,t), \tag{35}$$

then the surface normal is expressed as,

$$Normal_{surface}(x,t) = \frac{\vec{y} - s(x,t)}{\sqrt{1+s^2(x,t)}}. \tag{36}$$

We denote the direction of incident ray of light is N_i, according to the angle of incidence equals to the angle of reflection. We can obtain the reflected ray of light,

$$N_r = N_i - 2N_s(N_s \cdot N_i). \tag{37}$$

3.5.2 Transmission

The expression for transmission of light is not as simple as reflection of light. We have to consider two things. First, the direction of transmitted ray is only related with the surface normal and the direction of incident ray. Second, the transmission between two different mediums obeys Snell's Law.

Assuming that the direction of incident ray is N_i in a medium with index of refraction n_i, and the transmitted ray is in the medium with index of n_t, so we can

calculate the angle between incident ray and surface normal,

$$\sin\theta_i = \sqrt{1-(N_i \cdot N_s)^2} = \left|N_i \times N_s\right|, \tag{38}$$

and the angle of the transmitted ray will be

$$\sin\theta_t = \left|N_t \times N_s\right|. \tag{39}$$

According to the Snell's law, we can calculate the direction of refracted ray.

3.5.2 Fresnel Reflectivity and Transtivity

When light proceeds through media, parts of it will transmit from one medium to another. The other parts of it will be reflected. So theoretically no light is lost during the proceeding. In other words, reflectivity R and transtivity T are related by this following constraint,

$$R + T = 1. \tag{40}$$

However the derivation of the expression for R and T is based on the electromagnetic theory of dielectrics. We do not attempt to introduce where it came from. We consult some charts in [10] which referenced some papers in surface wave optics, we directly use the following Fresnel term in our algorithm, which stands for R :

$$F(N_i, N_r) = \frac{1}{2}\{\frac{\sin^2(\theta_t - \theta_i)}{\sin^2(\theta_t + \theta_i)} + \frac{\tan^2(\theta_t - \theta_i)}{\tan^2(\theta_t + \theta_i)}\}. \tag{41}$$

Chapter 4
Implementation and Results

We implement our system on a PC with 1.6GHz AMD Sempron Processor, 2.0 GB of system RAM, and a GeFroce 6200 PCI-X with 128MB of video memory. In average, it takes 0.29s to generate each frame. The conjugated gradient method solver for simulating the tsunami surface movement takes most of the computing time. Though we only store non-zero parts of the n-by-n symmetric and positive-definite matrix in the conjugated gradient method, it indeed reduces the complexity from $O(n^2)$ to $O(n)$ (n depends on the number of vertex). Solving conjugated gradient method is still time-consuming. Then we implement environment mapping in our shader program on GPU to make the surface reflection more physically real.

In the following two sections, we will demonstrate how we simulate the propagation of tsunami wave in our system. In Section 4.1, we show the difference of water surface when different ocean topographies are applied for simulation. In Section 4.2, we show the simulations with different initial tsunami amplitudes. And the boat appearing in the scene is about 60 meters in length.

4.1 Different ocean topographies

Figure 4.1 is a height map which we download from a weather website [14]. It is an ocean topography in the direction of northeast of Taiwan. The brighter pixels mean the closer to the sea level.

Figure 4.1 The height map of the underwater terrain in the northeast of Taiwan

In Figures 4.2 - 4.6, we only demonstrate the tsunami surface propagating on the ocean surface by setting Figure 4.1 as the ocean topography at different frames.

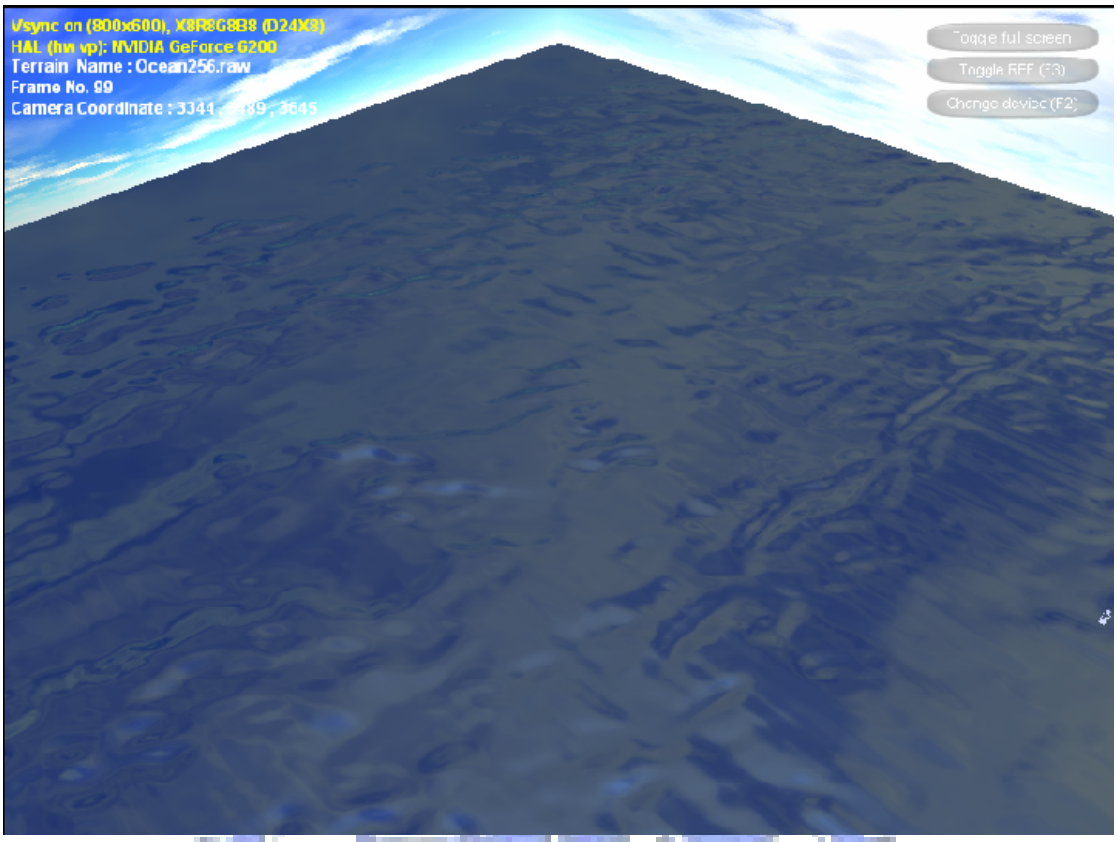


Figure 4.2 Frame No. 99 (Ocean256.raw)

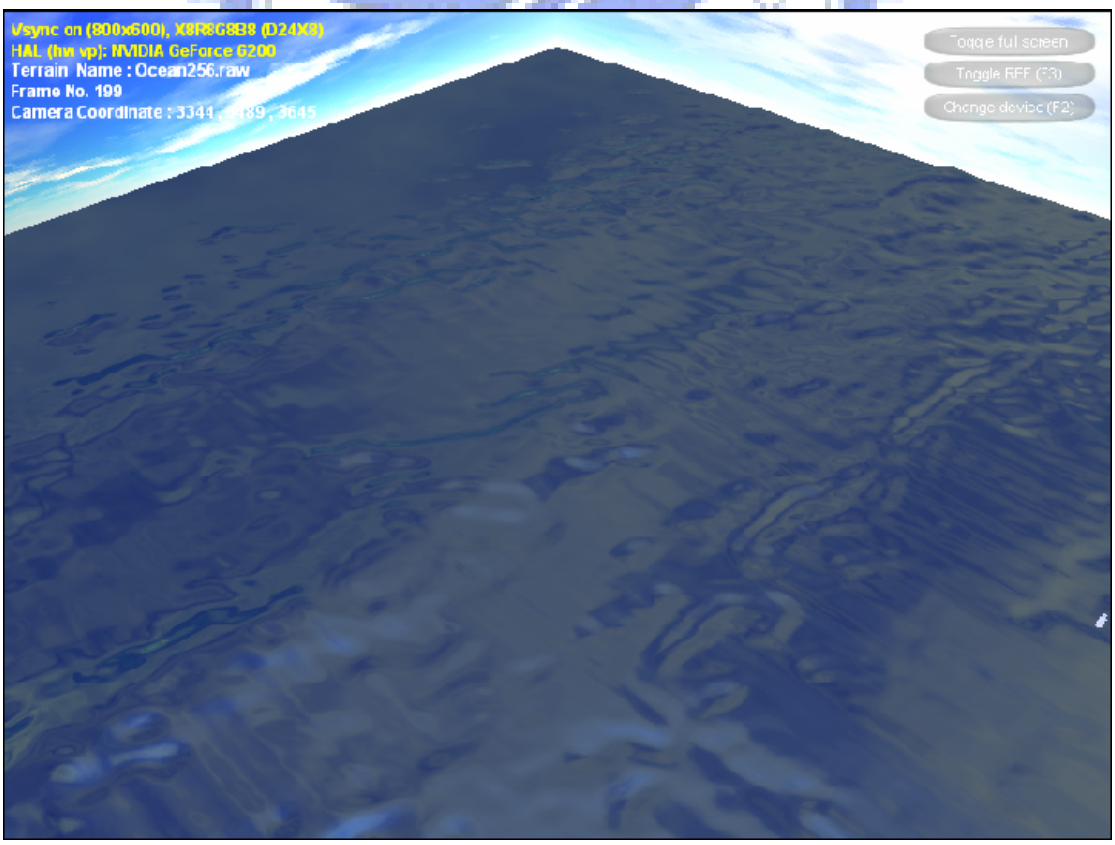


Figure 4.3 Frame No. 199 (Ocean256.raw)

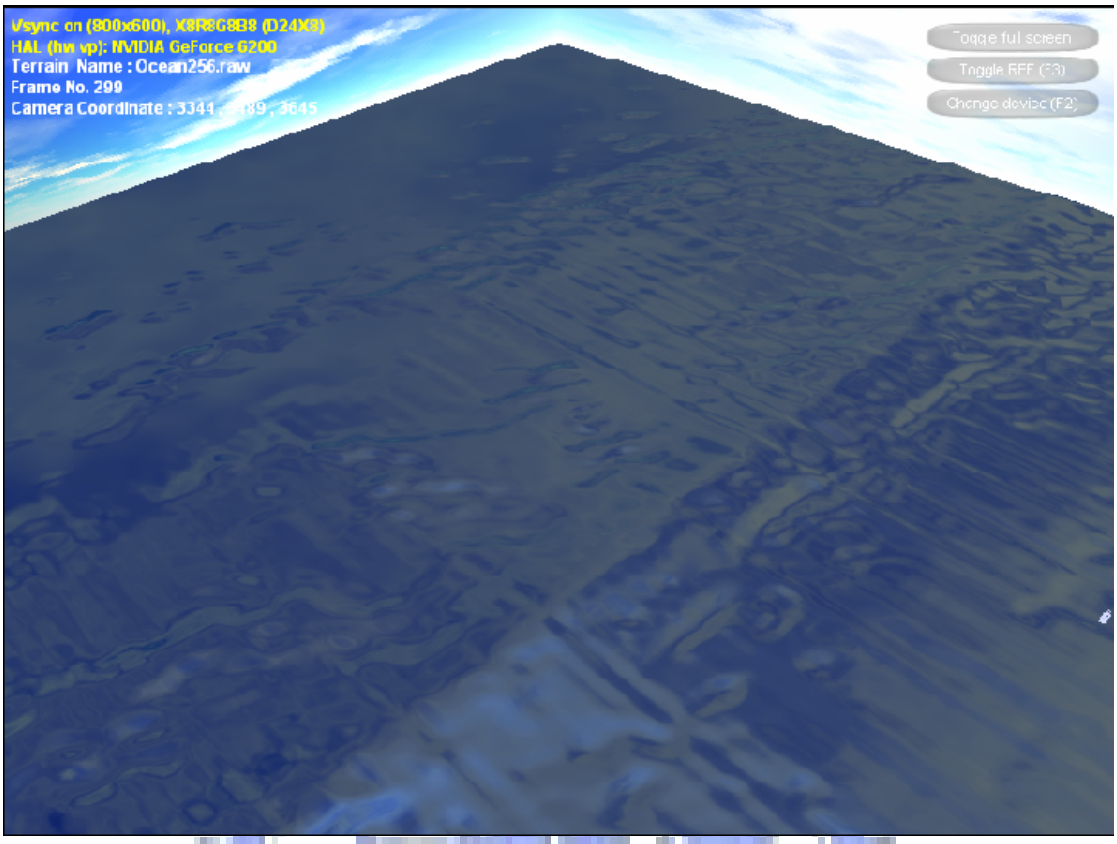


Figure 4.4 Frame No. 299 (Ocean256.raw)

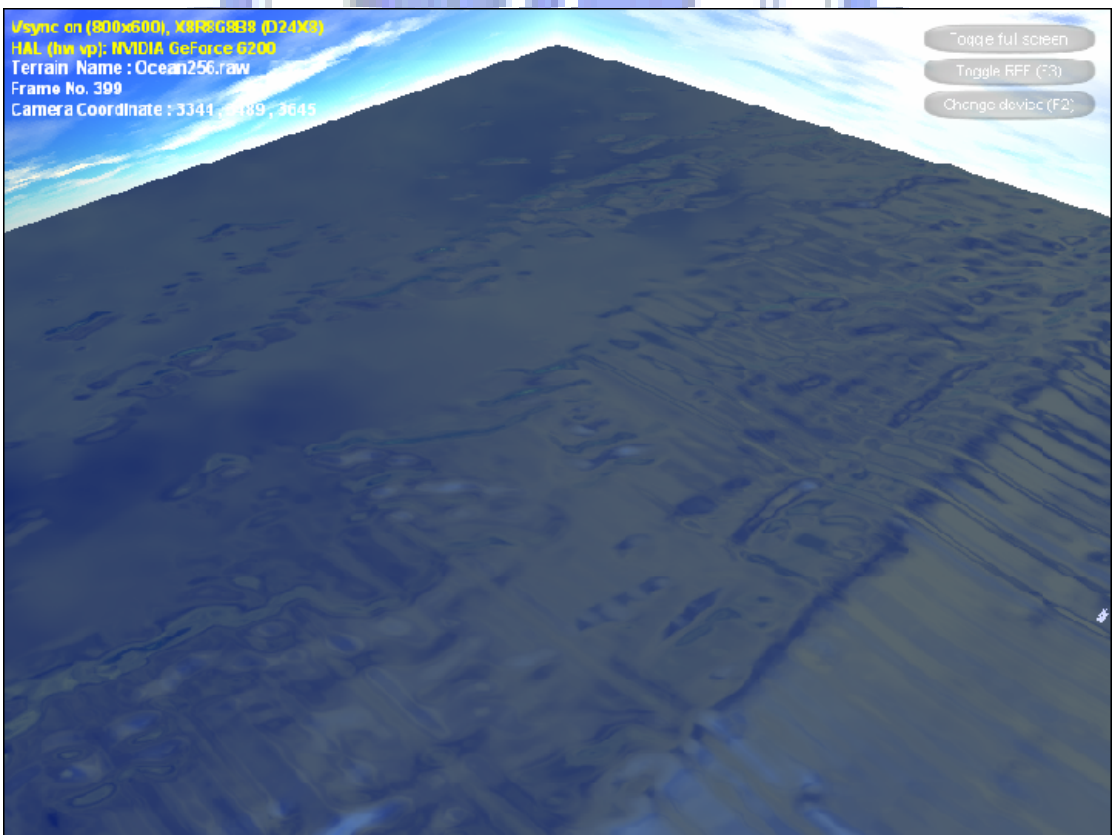


Figure 4.5 Frame No. 399 (Ocean256.raw)

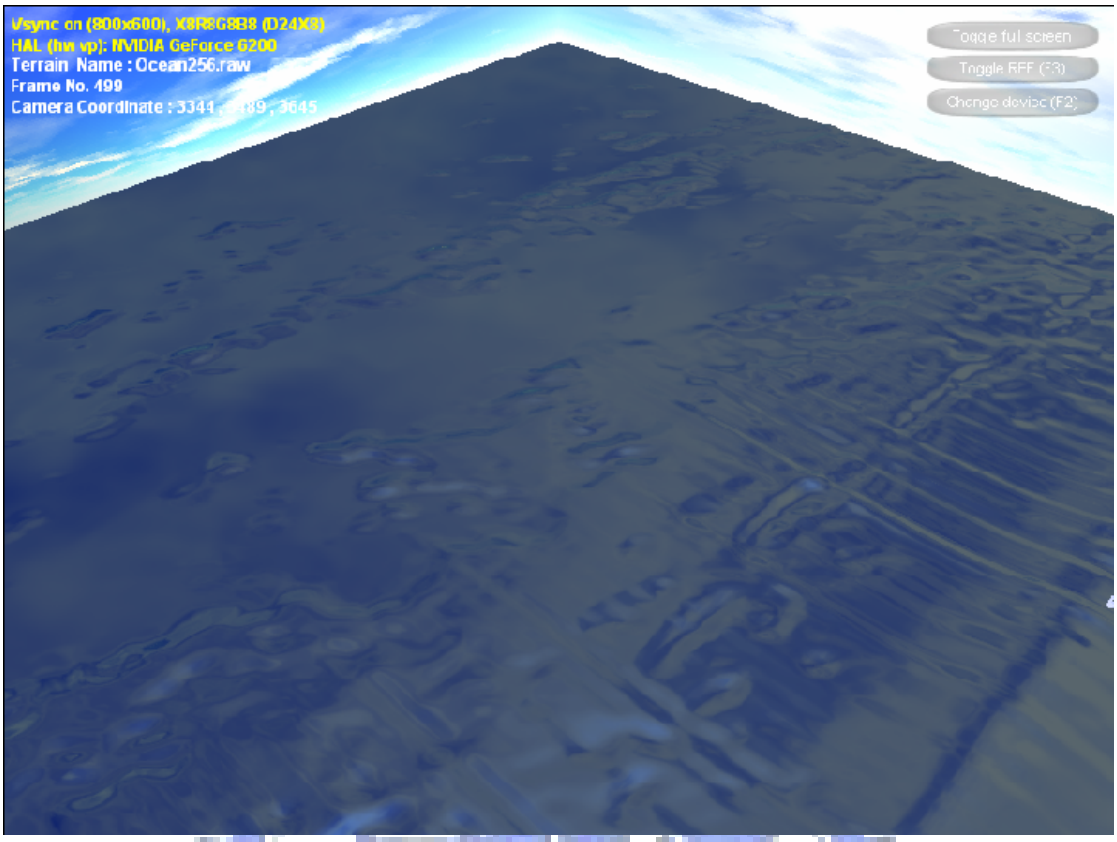


Figure 4.6 Frame No. 499 (Ocean256.raw)

Figure 4.7 A height map edited by using image processing tool [13].

Figure 4.7 The edited height map

In Figures 4.8-4.12, we demonstrate the corresponding movement of surface.

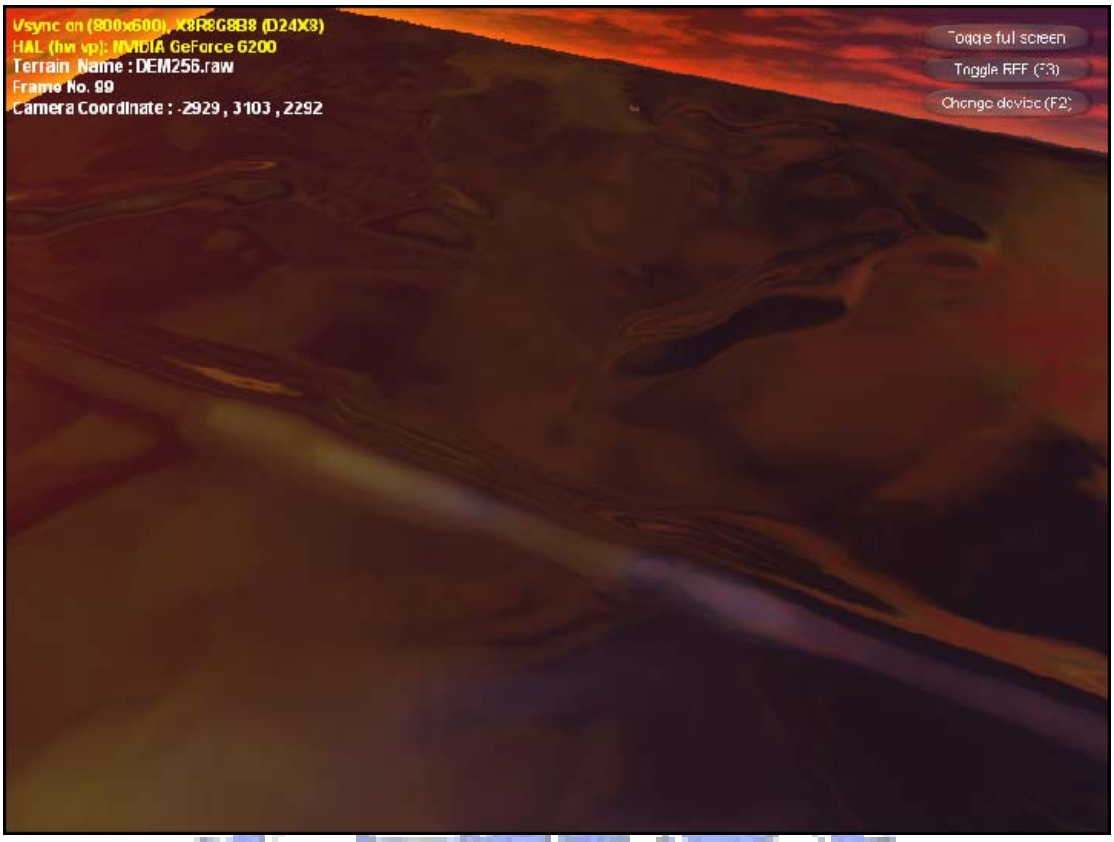


Figure 4.8 Frame No. 99 (DEM256.raw)

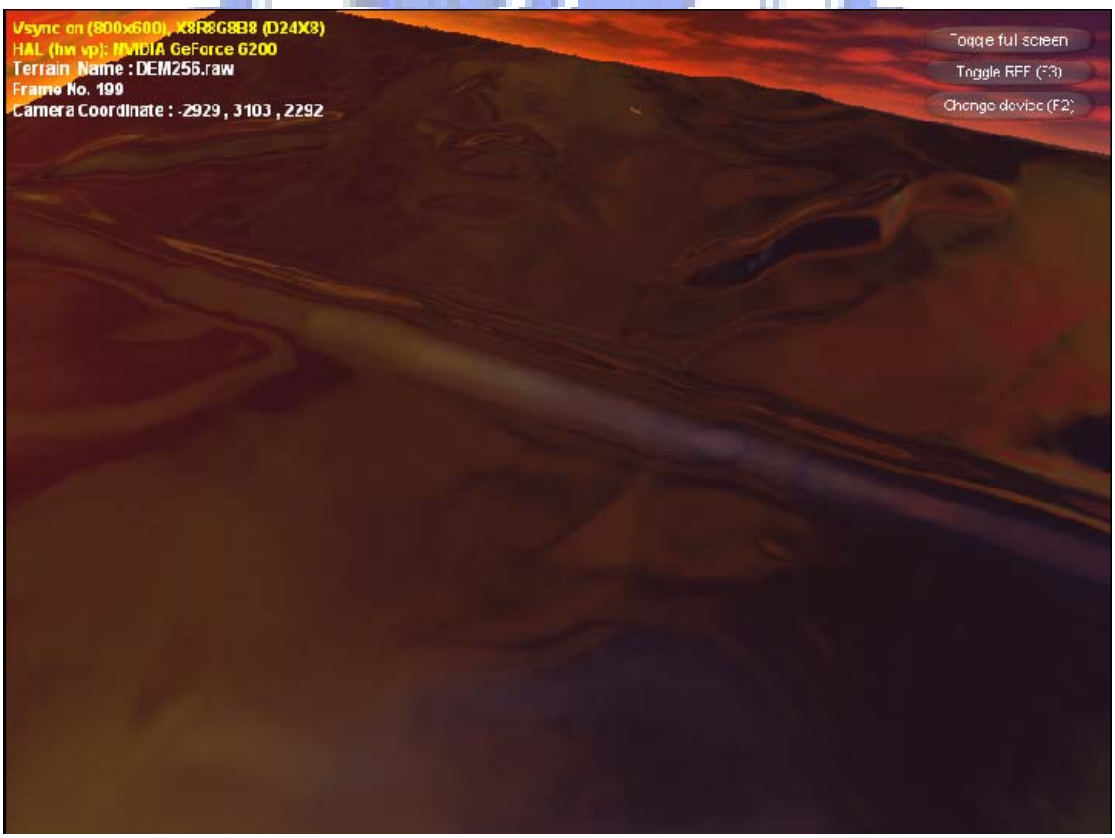


Figure 4.9 Frame No. 199 (DEM256.raw)

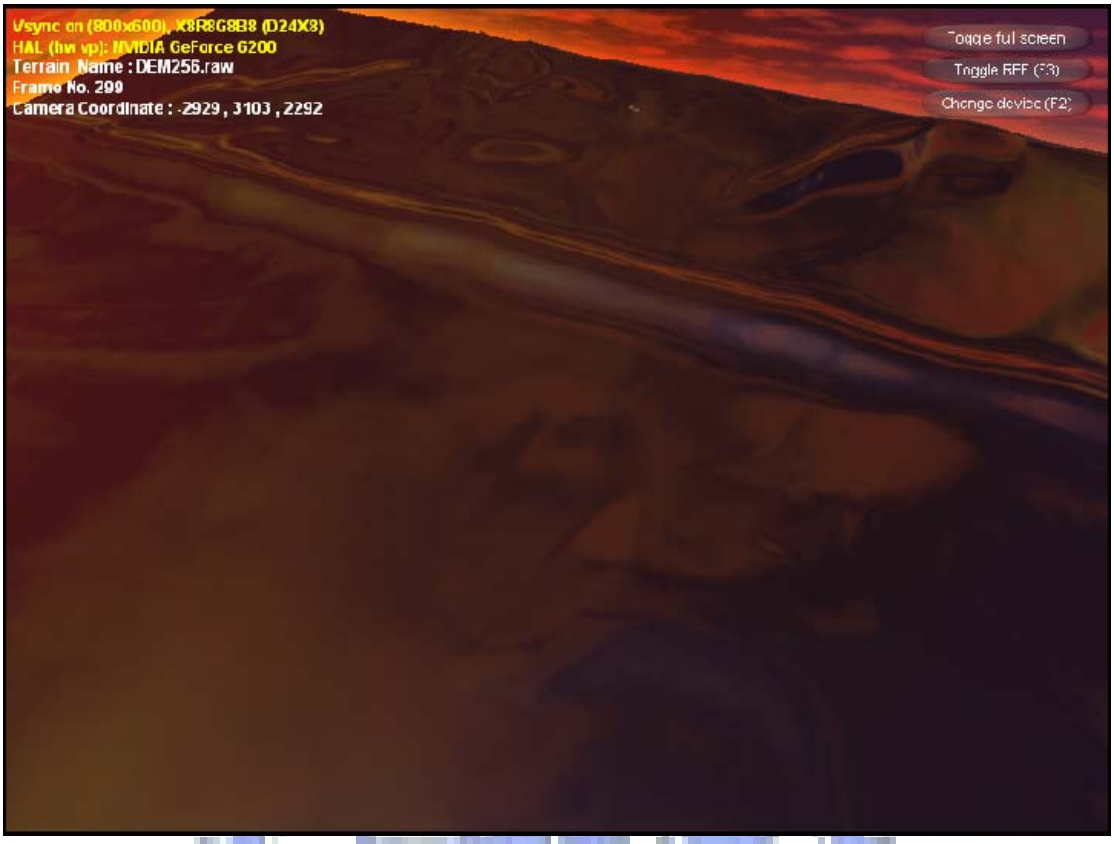


Figure 4.10 Frame No. 299 (DEM256.raw)

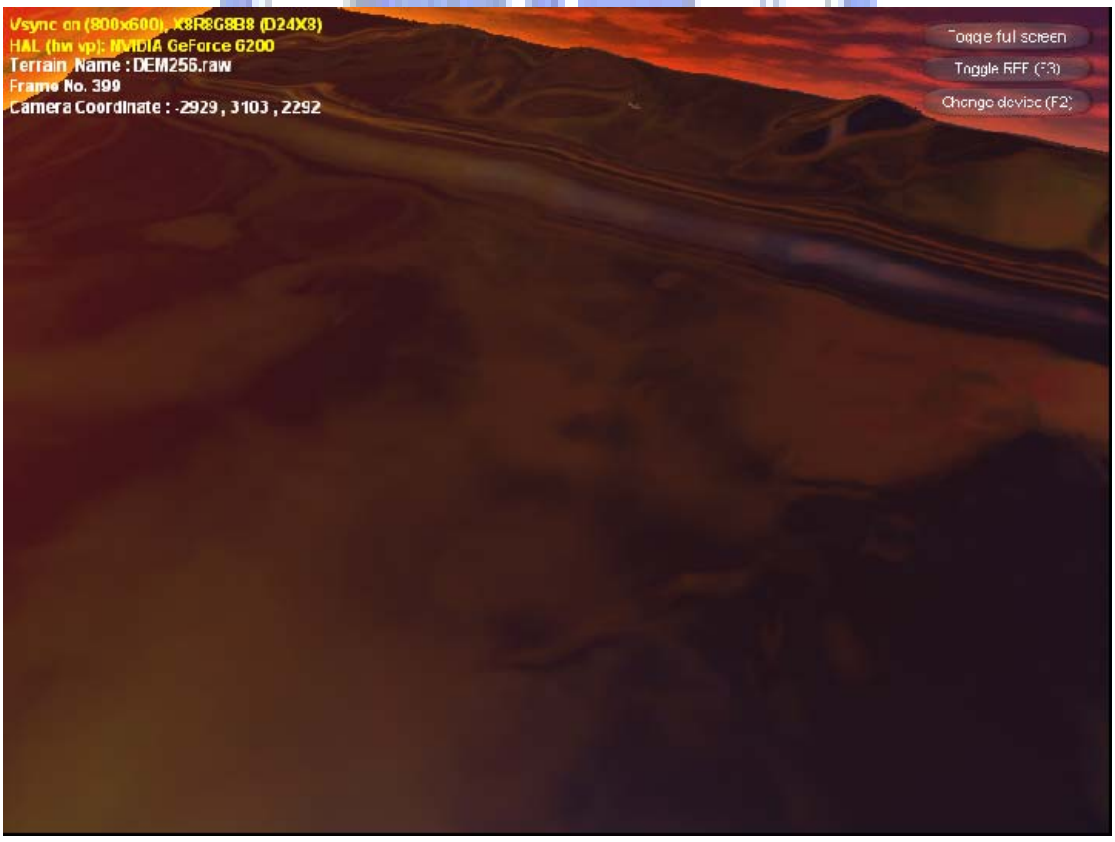


Figure 4.11 Frame No. 399 (DEM256.raw)

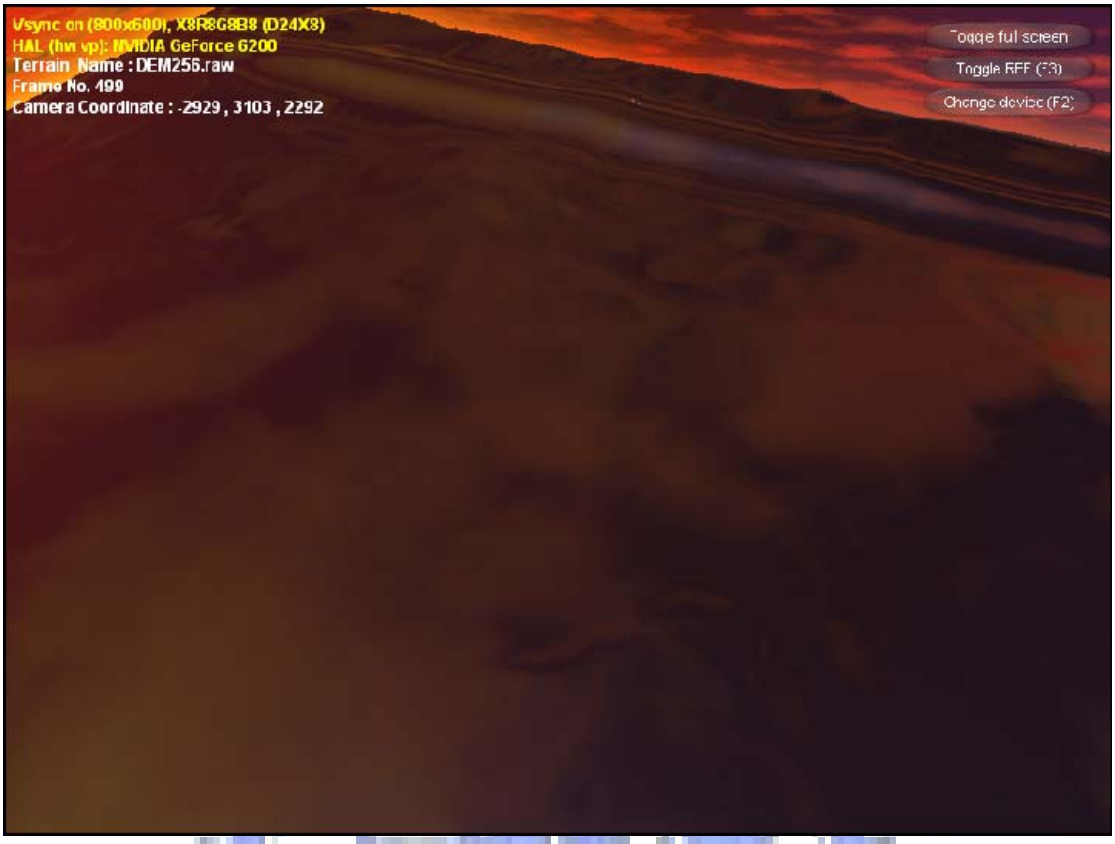


Figure 4.12 Frame No. 499 (DEM256.raw)

Through these simulations, we can find out that the generated tsunami surface with real ocean topography is much rougher than the one with edited height map. The reason is the depth of the ocean determines the velocity of wave. So the initial ocean wave propagating velocities on the surface with real ocean topography are changing. The depth varies so abruptly in the edited height map that the initial ocean wave propagating velocities differ a lot in that area. Then we can observe the surface generating much higher waves at the position.

The following Figures 4.13a-f and Figures 4.14a-f are the simulation of tsunami waves with applying ocean surface.

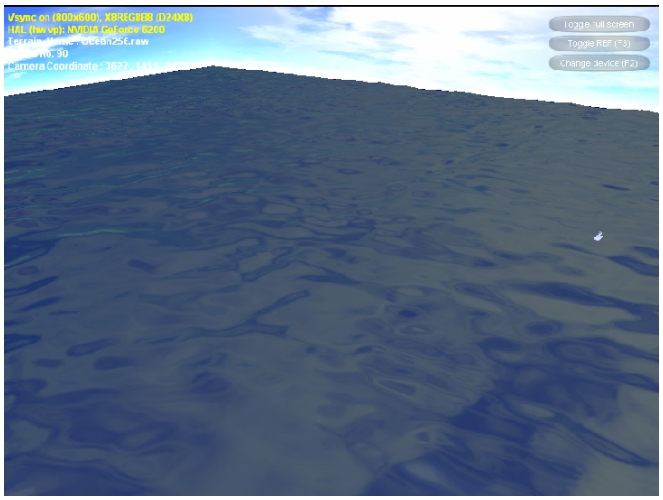


Figure 4.13a Frame No. 90

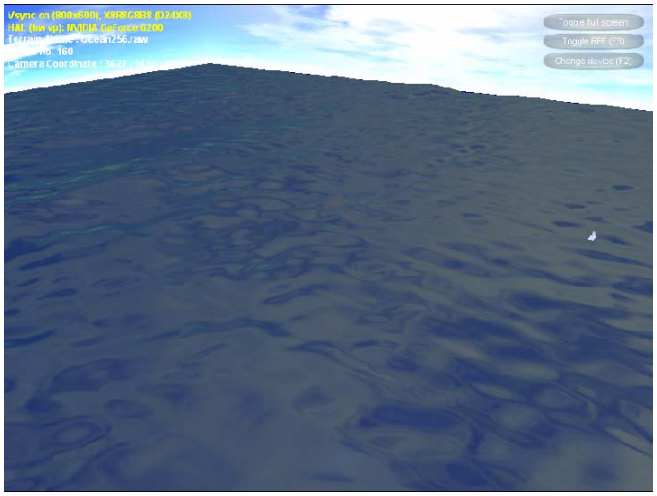


Figure 4.13b Frame No. 160

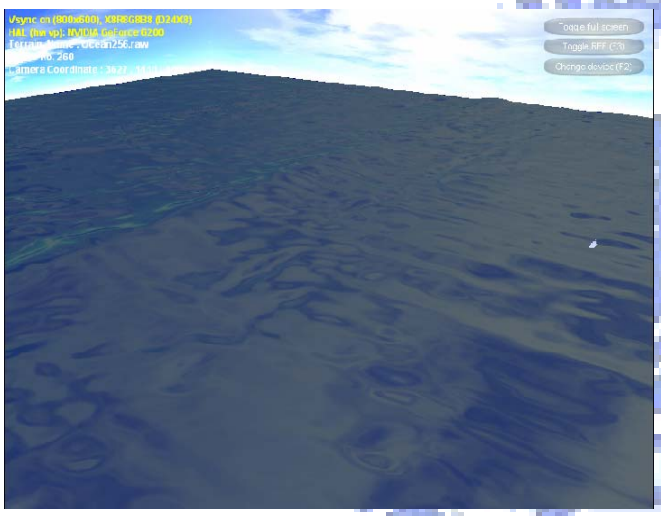


Figure 4.13c Frame No. 260

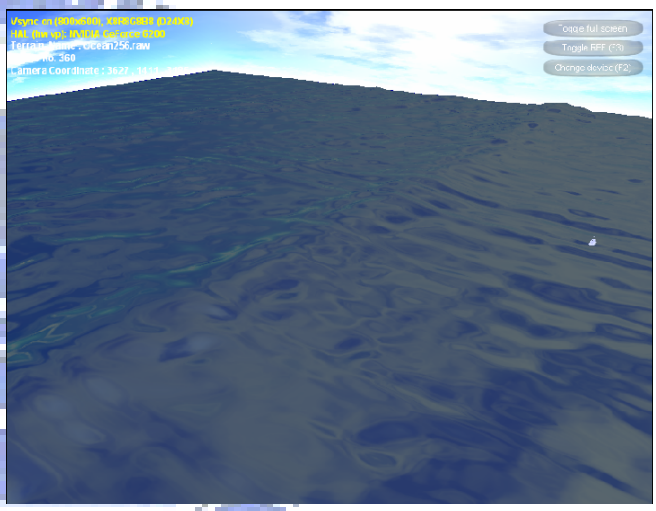


Figure 4.13d Frame No. 360

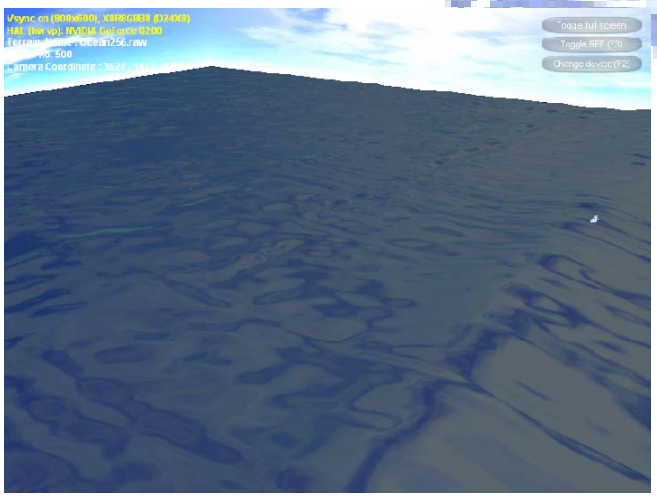


Figure 4.13e Frame No. 500

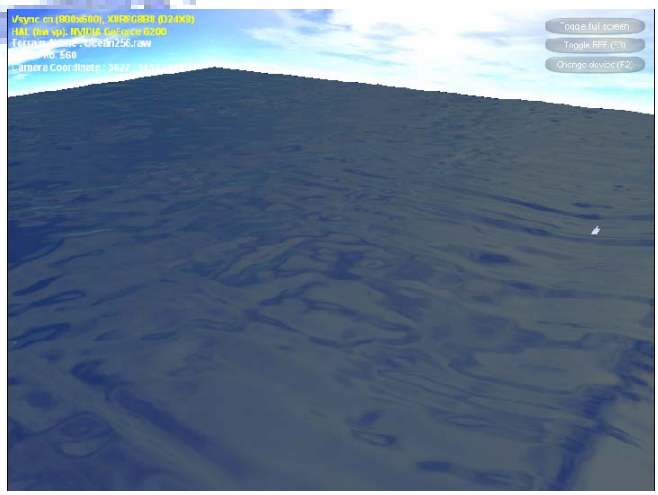


Figure 4.13f Frame No. 560

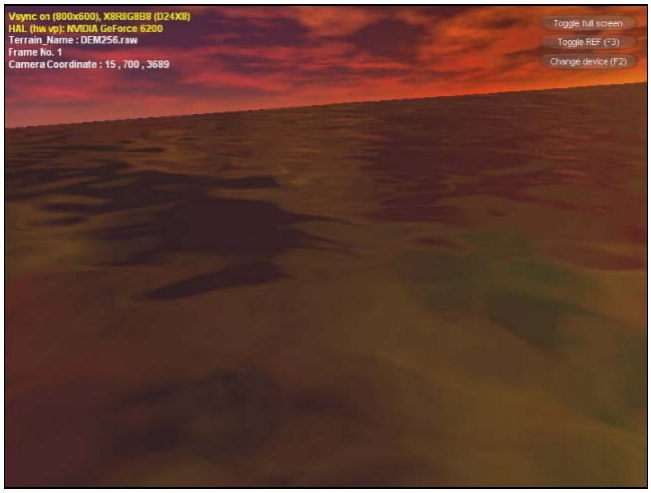


Figure 4.14a Frame No. 1

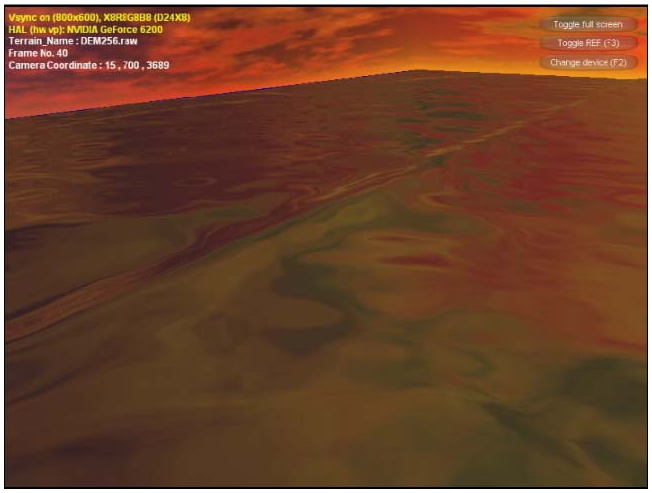


Figure 4.14b Frame No. 40

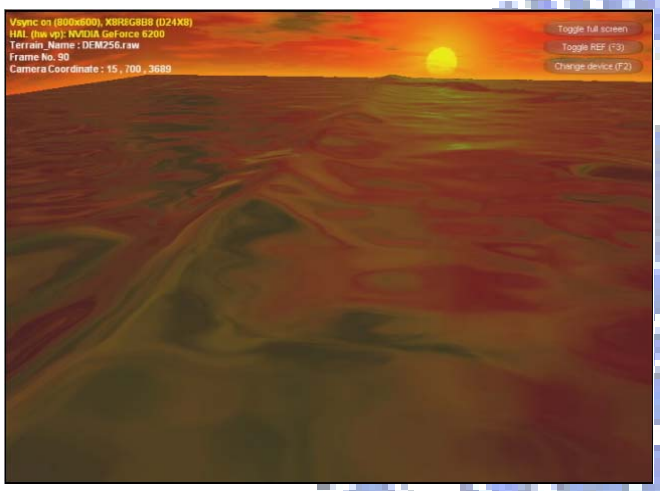


Figure 4.14c Frame No. 90

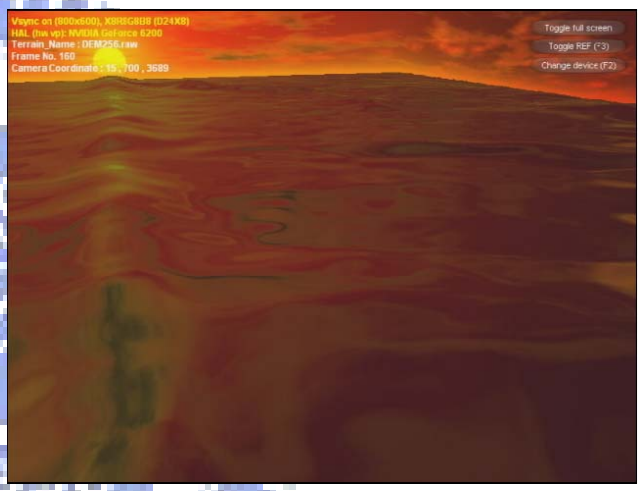


Figure 4.14d Frame No. 160

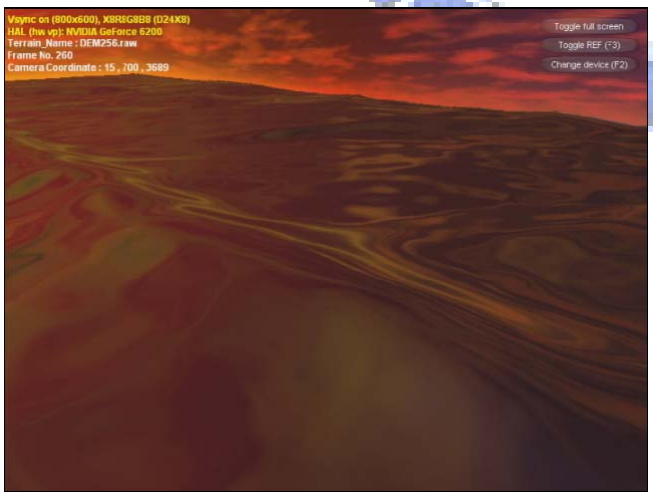


Figure 4.14e Frame No. 260

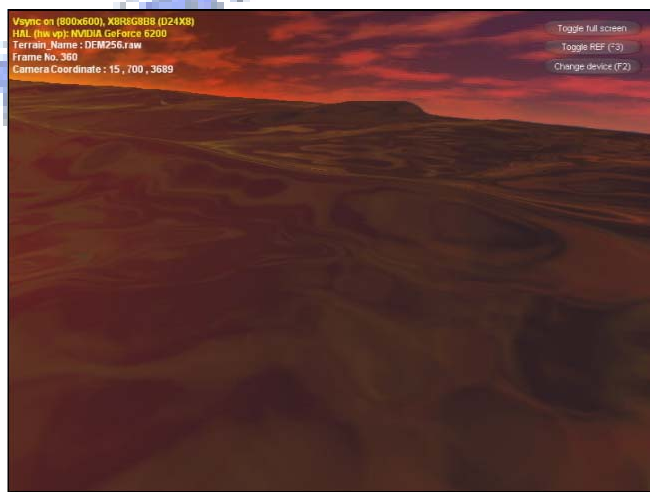


Figure 4.14e Frame No. 360

4.2 Different initial tsunami amplitudes

This section we describe the differences by using different initial amplitudes of tsunami waves.

Figures 4.15a-c show that the initial amplitude of tsunami wave is 15 meters, and Figures 4.16a-c show that the initial amplitude of tsunami wave is 1 meter. We can see that, the simulated tsunami surface is different apparently with different initial amplitude values. We also can notice that different amplitudes result in different wave velocities. So at the same frame, the wave crests are at different positions. Higher amplitude results in higher speed.

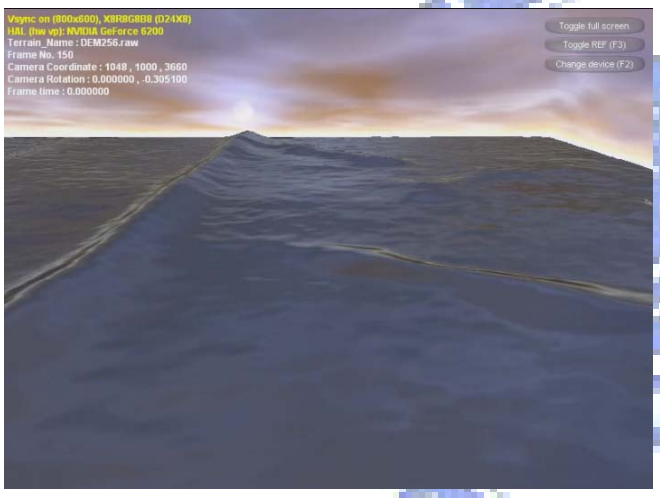


Figure 4.15a Frame No. 150 (amplitude 15m)

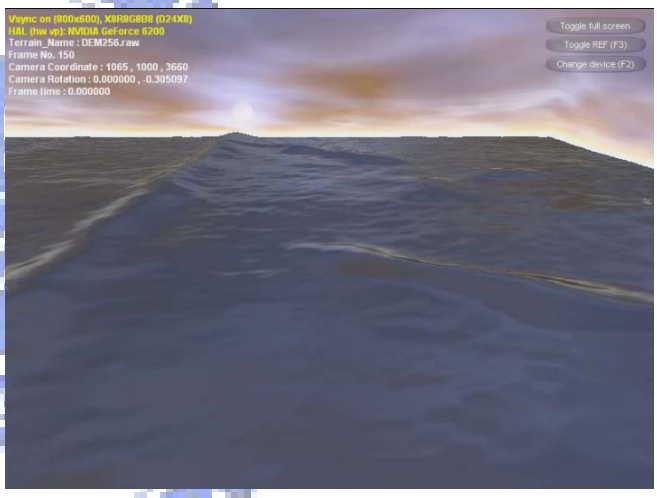


Figure 4.16a Frame No. 150 (amplitude 1m)

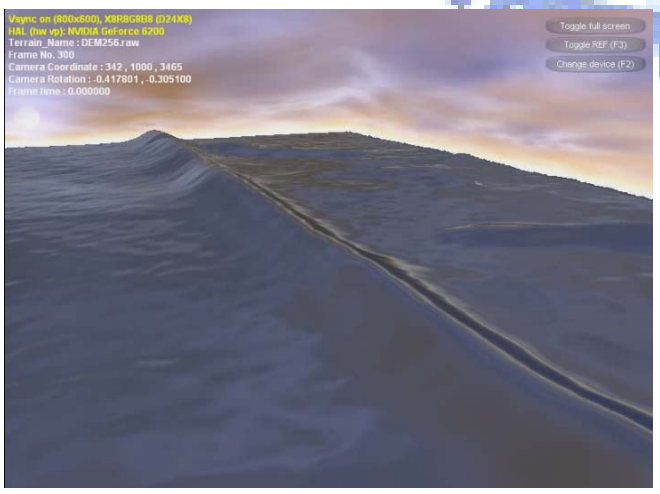


Figure 4.15b Frame No. 300

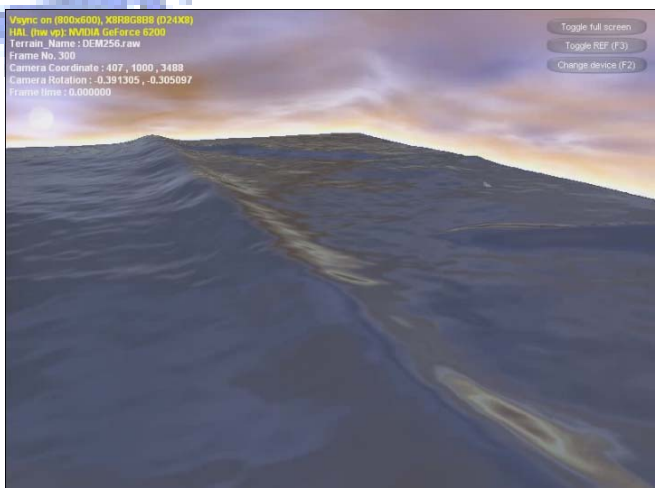


Figure 4.16b Frame No. 300

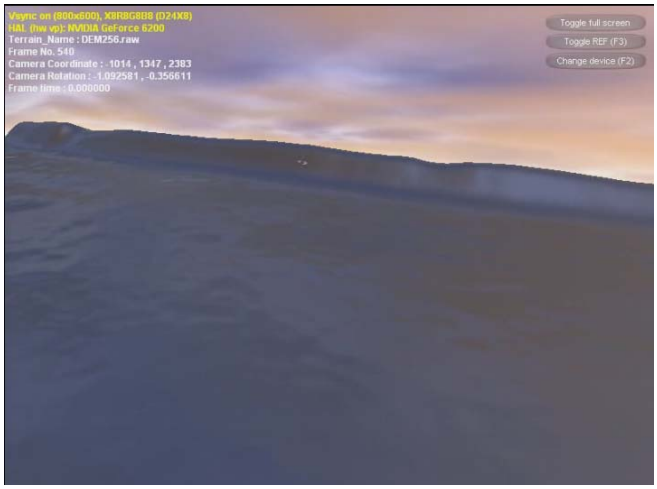


Figure 4.15c Frame No. 540

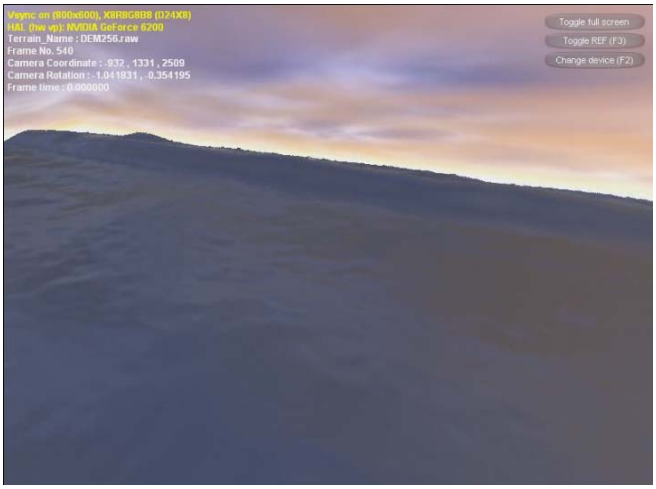


Figure 4.16c Frame No. 540

Chapter 5
Conclusion and Future Works

We have presented a simulation system for tsunami wave propagation. By combining the shallow water equations which are often used in ocean engineering to simulate the movement of tsunami waves and the rendering techniques developed in computer graphics to render the 3D scene. We can obtain more physically-correct tsunami wave propagation. The shallow water equation is simplified by integrating the three-dimensional Navier-Stokes equations in *z* direction. It can simulate the wave movement correctly when the ratio of depth of water to the wave length is rather small, just like tsunami waves. Then we apply a more stable and efficient numerical method, called Semi-Lagrangian method, on SWE and it allows that we can maintain the stability of the simulation system with larger time steps. Conjugated gradient method is then used to solve the linear system because it can reduce convergence time by searching the optimized solution along the conjugated direction. Perlin's noise function is introduced and we generated several fractal surfaces. By blending these surfaces, the final ocean surface can be generated. With environment mapping and physically-based lighting model, we can obtain the final simulation results of tsunami wave propagation.

In the future, we may apply the statistical wave models and the Fast Fourier Transforms to generate much more realistic ocean surface. This method is accepted by most oceanographers and described in oceanographic literature. It is based on the

ability to decompose the wave height field as a sum of sine and cosine waves, and the decomposition uses FFT computationally. Besides, we can divide parts of the ocean surface especially at crests into 3D grids and apply 3D Navier-Stokes equations on these grids. So we may obtain much more detail on the crests and simulate other parts of the surface by SWE. This hybrid simulation structure may bring us more realistic result and it will not spend a lot of time such as simulating the whole scene by 3D Navier-Stokes equations.

References

[1] Erleben., Spopring., Henriksen., and Dohlmann. Physics-based animation. Charles River Media. Graphics Series. 2005

[2] Foster, N., and Fedkiw, R. 2001. Practical animation of liquids. In *Proceedings of SIGGRAPH 2001*, ACM Press / ACM SIGGRAPH, E. Fiume, Ed., Computer Graphics Proceedings, Annual Conference Series, ACM, 23-30.

[3] Fournier, A., and Reeves, W. T., 1986. A simple model of ocean waves. *Computer Graphics (SIGGRAPH '86)*, 20, 75-84.

[4] Gonzato, J. C., and Saëc, B. L. 2000. On modeling and rendering ocean scenes. *The Journal of Visualization and Computer Animation 11*, 1, 27-37.

[5] Irving, G., guendelman, E., Losasso, F., Fedkiw, R. 2006. Efficient Simulation of Large Bodies of Water by Coupling Two and Three Dimensional Techniques. *ACM Transactions on Graphics 25*.

[6] Jason, L. M. 2005. Real-Time Synthesis and Rendering of Ocean Water. *ATI Research Technical Report, April 2005.*

[7] Losasso, F., Gibou, F., and Fedkiw, R. 2004. Simulating water and smoke with an octree data structure. *ACM Trans. Graph. (SIGGRAPH Proc.)*23, 457-462.

[8] O'Brien, J. F., and Hodgins, J. K. 1995. Dynamic simulation of splashing fluids. In *Comput. Anim.* '95, 198-205.

[9] Perlin, K. An image synthesizer. 1985. In *Computer Graphics(SIGGRAPH '85 Proceedings)*, B. A. Barsky, Ed. 1985, vol.19(3), 287-296.

[10] Tessendorf, J., Simulating ocean waters. 2001. In *SIGGRAPH course notes (course 47).*

[11] Xudong, Y., Xuexian, P., Liang, Z., and Sikun, L. 2005. GPU-based real-time simulation and rendering of unbounded ocean surface. *Computer Aided Design and Computer Graphics, 2005.* Ninth International Conference.

[12] Yaohua, H., Luiz, V., Xin, T., Baining, g., and Harry, S. 2006. Realistic, real-time rendering of ocean waves. *Journal of Visualization and Computer Animation, Volume 17*. 59-67.

[13] Adobe System. Adobe Photoshop.

[14] http://ilmuse.gov.tw/~epaper/200609/05.htm