The Simulation of Tsunami Wave Propagation

Yen-Ting Kuo
Institute of Multimedia Engineering
National Chiao Tung University
Hsinchu 30010, Taiwan, ROC
is90013@cis.nctu.edu.tw

Zen-Chung Shih
Department of Computer Science
National Chiao Tung University
Hsinchu 30010, Taiwan, ROC
zcshih@cs.nctu.edu.tw

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 more understanding about tsunami waves.

1. Introduction

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.

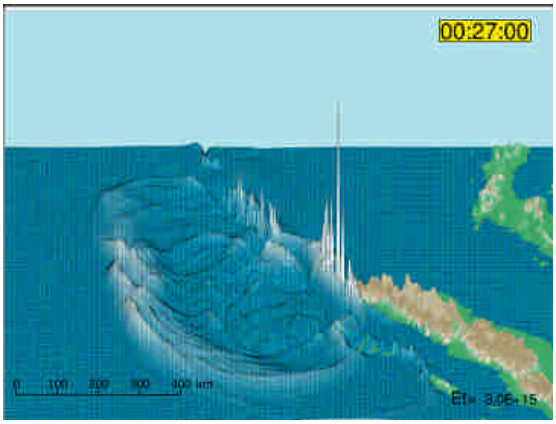

Figure 1.1: The simulation of tsunami surface.

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.

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

0-7695-3084-2/07 $25.00 © 2007 IEEE
DOI 10.1109/ISM.Workshops.2007.7

and results will be demonstrated in Chapter 4. Finally, Conclusion and future works will be given in Chapter 5.

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 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.

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] presents a multi-band Fourier domain approach to synthesize the ocean surface and rendering the waves on GPU entirely.

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.

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. Ocean surface construction is described in section 3.3. Finally, we show how we render the ocean scene in section 3.4.

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 original form. The shallow water equations are shown as (1a-c),

$$\frac{\partial u}{\partial t}+u\frac{\partial u}{\partial x}+v\frac{\partial u}{\partial y}=-g\frac{\partial h}{\partial x} \quad (1a)$$

$$\frac{\partial v}{\partial t}+u\frac{\partial v}{\partial x}+v\frac{\partial v}{\partial y}=-g\frac{\partial h}{\partial y} \quad (1b)$$

$$\frac{\partial h}{\partial t}+u\frac{\partial h}{\partial x}+v\frac{\partial h}{\partial y}=-(\frac{\partial u}{\partial x}+\frac{\partial v}{\partial y})h \quad (1c)$$

We then solve the above equations via the semi-Lagrangian method [10].

3.2 Applying semi-Lagrangian method to SWE

To show how the SWE can be integrated by the semi-Lagrangian method, we first consider equations (1) and then we replace the substantial derivates in (1) with ordinary derivates

$$\frac{du}{dt}=-g\frac{\partial h}{\partial x}, \quad (2a)$$

$$\frac{dv}{dt}=-g\frac{\partial h}{\partial x}, \quad (2b)$$

$$\frac{dh}{dt}=-(\frac{\partial u}{\partial x}+\frac{\partial v}{\partial y})h. \quad (2c)$$

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

$$\frac{du}{dt}=\frac{u(t+\Delta t)-\tilde{u}(t)}{\Delta t}, \quad (3a)$$

$$\frac{dv}{dt}=\frac{v(t+\Delta t)-\tilde{v}(t)}{\Delta t}, \quad (3b)$$

$$\frac{dh}{dt}=\frac{h(t+\Delta t)-\tilde{h}(t)}{\Delta t}. \quad (3c)$$

Functions $\tilde{u}$, $\tilde{v}$ and $\tilde{h}$ 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 following:

$$\begin{bmatrix}\tilde{x}(t-\Delta t)\\ \tilde{y}(t-\Delta t)\end{bmatrix}=\begin{bmatrix}x(t)\\ y(t)\end{bmatrix}-\Delta t\begin{bmatrix}u(t)\\ v(t)\end{bmatrix}, \quad (4)$$

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

$$\tilde{u}=(1-a)(1-b)u(\lfloor\tilde{x}\rfloor,\lfloor\tilde{y}\rfloor)+a(1-b)u(\lceil\tilde{x}\rceil,\lfloor\tilde{y}\rfloor)+, \quad (5)$$
$$abu(\lceil\tilde{x}\rceil,\lceil\tilde{y}\rceil)+(1-a)bu(\lfloor\tilde{x}\rfloor,\lceil\tilde{y}\rceil)$$

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 (2) and combine with (3), then we obtain,

$$\frac{u(t+\Delta t)-\tilde{u}(t)}{\Delta t}=-g\frac{\partial h(t+\Delta t)}{\partial x}, \quad (6a)$$

$$\frac{v(t+\Delta t)-\tilde{v}(t)}{\Delta t}=-g\frac{\partial h(t+\Delta t)}{\partial y}, \quad (6b)$$

$$\frac{h(t+\Delta t)-\tilde{h}(t)}{\Delta t}=-(\frac{\partial u(t+\Delta t)}{\partial x}+\frac{\partial v(t+\Delta t)}{\partial y})h(t), \quad (6c)$$

Then we calculate the derivative of (6a) with respect to x and (6b) with respect to y, we obtain,

$$\frac{\partial u(t+\Delta t)/\partial x-\partial\tilde{u}(t)/\partial x}{\Delta t}=-g\frac{\partial^2 h(t+\Delta t)}{\partial x^2}, \quad (7a)$$

$$\frac{\partial v(t+\Delta t)/\partial y-\partial\tilde{v}(t)/\partial y}{\Delta t}=-g\frac{\partial^2 h(t+\Delta t)}{\partial y^2}. \quad (7b)$$

Next we eliminate $u(t+\Delta t)$, $v(t+\Delta t)$ in (21c) by using (7a) and (7b) as,

$$\frac{h(t+\Delta t)-\tilde{h}(t)}{\Delta t}=-(\frac{\partial\tilde{u}(t)}{\partial x}-\Delta tg\frac{\partial^2 h(t+\Delta t)}{\partial x^2}+\frac{\partial\tilde{v}(t)}{\partial y}-\Delta tg\frac{\partial^2 h(t+\Delta t)}{\partial y^2})h(t), \quad (8)$$

after rearranging the terms,

$$h(t+\Delta t)=\tilde{h}(t)-\Delta th(t)(\frac{\partial\tilde{u}(t)}{\partial x}-\Delta tg\frac{\partial^2 h(t+\Delta t)}{\partial x^2}+\frac{\partial\tilde{v}(t)}{\partial y}-\Delta tg\frac{\partial^2 h(t+\Delta t)}{\partial y^2}). \quad (9)$$

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

$$\begin{aligned}h_{i,j}(t+\Delta t)&=\tilde{h}_{i,j}(t)-\Delta th_{i,j}(t)\frac{\partial\tilde{u}(t)}{\partial x}+(\Delta t)^2h_{i,j}(t)g\frac{\partial^2 h_{i,j}(t+\Delta t)}{\partial x^2}\\&\quad-\Delta th_{i,j}(t)\frac{\partial\tilde{v}(t)}{\partial y}+(\Delta t)^2h_{i,j}(t)g\frac{\partial^2 h_{i,j}(t+\Delta t)}{\partial y^2}\\&=\tilde{h}_{i,j}(t)-\Delta th_{i,j}(t)(\frac{\tilde{u}_{i+1,j}(t)-\tilde{u}_{i-1,j}(t)}{2\Delta x})\\&\quad+(\Delta t)^2h_{i,j}(t)g(\frac{h_{i+1,j}(t+\Delta t)-2h_{i,j}(t+\Delta t)+h_{i-1,j}(t+\Delta t)}{\Delta x^2})\\&\quad-\Delta th_{i,j}(t)(\frac{\tilde{v}_{i,j+1}(t)-\tilde{v}_{i,j-1}(t)}{2\Delta y})\\&\quad+(\Delta t)^2h_{i,j}(t)g(\frac{h_{i,j+1}(t+\Delta t)-2h_{i,j}(t+\Delta t)+h_{i,j-1}(t+\Delta t)}{\Delta y^2})\end{aligned} \quad (10)$$

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

$$\begin{aligned}&h_{i,j}(t+\Delta t)-(\Delta t)^2h_{i,j}(t)g(\frac{h_{i+1,j}(t+\Delta t)-2h_{i,j}(t+\Delta t)+h_{i-1,j}(t+\Delta t)}{\Delta x^2})\\&\quad-(\Delta t)^2h_{i,j}(t)g(\frac{h_{i,j+1}(t+\Delta t)-2h_{i,j}(t+\Delta t)+h_{i,j-1}(t+\Delta t)}{\Delta y^2})\\&=\tilde{h}_{i,j}(t)-\Delta th_{i,j}(t)(\frac{\tilde{u}_{i+1,j}(t)-\tilde{u}_{i-1,j}(t)}{2\Delta x})-\Delta th_{i,j}(t)(\frac{\tilde{v}_{i,j+1}(t)-\tilde{v}_{i,j-1}(t)}{2\Delta y})\end{aligned} \quad (11)$$

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.

3.3 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.

$$fractal\_noise(x) = \sum_{i=0}^{n} \alpha^i \cdot perlin\_noise(2^i \cdot x). \quad (17)$$

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, and 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 (18) demonstrates the interpolation method for three fractal surfaces.

$$\begin{aligned} w_1 &= \cos(dt) \\ w_2 &= \cos(\frac{\pi}{3} + dt) \\ w_3 &= \cos(\frac{2\pi}{3} + dt) \\ Final\_surface &= w_1 * f_1 + w_2 * f_2 + w_3 * f_3 \end{aligned} \quad , \quad (18)$$

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

3.4 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 is split into two components. First, rays 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.

3.4.1 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,

$$r(x,t) = \vec{y}h(x,t) + x, \quad (19)$$

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

$$s(x,t) \equiv \nabla h(x,t), \quad (20)$$

then the surface normal is expressed as,

$$Normal_{surface}(x,t) = \frac{\vec{y} - s(x,t)}{\sqrt{1 + s^2(x,t)}}. \quad (21)$$

We denote the direction of incident ray of light as N_i. Since the angle of incidence equals to the angle of reflection, the reflected ray of light is,

$$N_r = N_i - 2N_s(N_s \cdot N_i). \quad (22)$$

3.4.2 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,

$$\sin\theta_i = \sqrt{1 - (N_i \cdot N_s)^2} = |N_i \times N_s|, \quad (23)$$

and the angle of the transmitted ray will be

$$\sin\theta_t = |N_t \times N_s|. \quad (24)$$

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

3.4.3 Fresnel Reflectivity and Transitivity. 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 transitivity T are related by this following constraint,

$$R + T = 1. \quad (25)$$

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 :

$$F(N_i, N_r) = \frac{1}{2}\left\{\frac{\sin^2(\theta_t - \theta_i)}{\sin^2(\theta_t + \theta_i)} + \frac{\tan^2(\theta_t - \theta_i)}{\tan^2(\theta_t + \theta_i)}\right\}. \quad (26)$$

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. The environment mapping is implemented as a shader program.

Figure 4.1 is a height map which we download from a weather website [14], and Figure 4.2 is the one edited with image processing tool [13].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

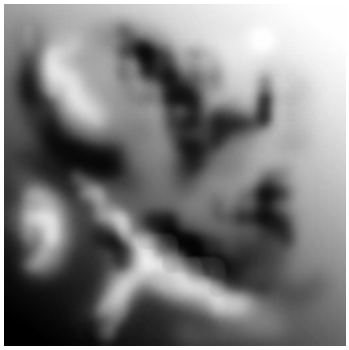

Figure 4.2 The edited height map

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 4.3a-b and Figures 4.4a-c are the simulation of tsunami waves with applying ocean surface and different amplitudes.

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.

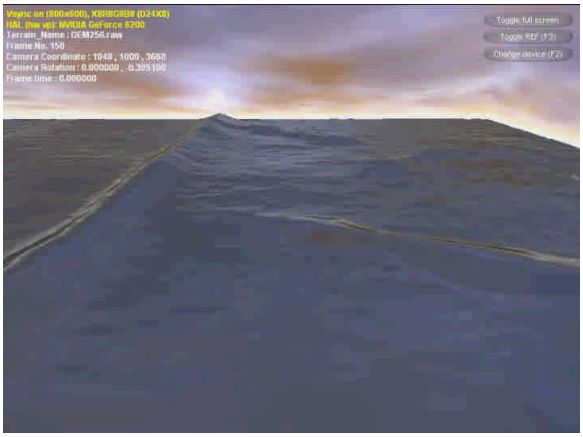

Figure 4.3a Frame No. 150 (amplitude 15m)

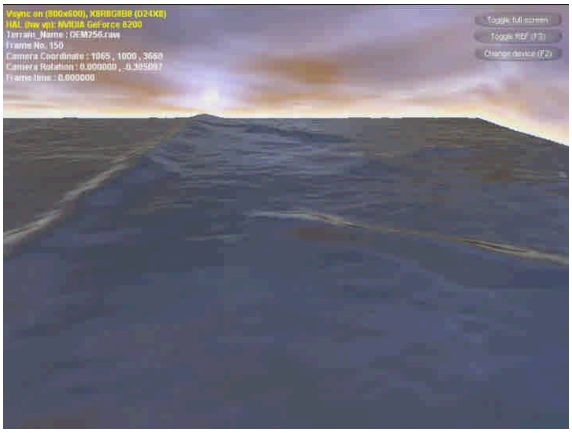

Figure 4.4a Frame No. 150 (amplitude 1m)

Figure 4.3b Frame No. 540 (amplitude 15m)

Figure 4.4b Frame No. 540 (amplitude 1m)

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.

In the future, we may apply statistical wave models and the Fast Fourier Transform to generate much more realistic ocean surface. 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.

6. References

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

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

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

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

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

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

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

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

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

[10] Stam, J., Stable Fluids. In the Proceedings of ACM SIGGRAPH 99, 121-128.

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

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

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

[14] Adobe System. Adobe Photoshop.

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