International Journal of Signal Processing, Image Processing and Pattern Recognition
Vol. 10, No. 3 (2017), pp. 63-74
http://dx.doi.org/10.14257/ijsip.2017.10.3.07

Water Reflection, Refraction Simulation Based on Perlin Noise and Ray-tracing

Hua Li [1, 2], Huamin Yang [1, 2] and Jianping Zhao [1, 2*]

(1.College of Computer Science and Technology, Changchun University of Science and Technology, Changchun 130022)
(2.National Local combined Engineering Research Center of Movie Technology and Equipment, Changchun 130000)
custlihua@gmail.com

Abstract

In order to better obtain the simulation of photorealistic water surface, a computer simulation method of the dynamic water surface was proposed with precise reflection and refraction. The method utilized 2D mesh modeling water surfaces and optimized the uniform grid of surface using level of detail (LOD). Multi-octave of Perlin noise was generated to construct random height field of water surface. Ray-tracing was enforced to trace the reflection path of light through pixels. Our method compared different iterative depth and different octaves. The results show that when the octave interposed of fractal Perlin noise was somewhere in between 4 to 8, and the maximum depth of ray tracing was in between 2-3 , the computing time the results can meet the requirements of refine observation.

***Keywords**: Perlin noise; Ray-tracing; Reflection; Water simulation; Octave*

1. Introduction

Water is ubiquitous in the nature. The simulation of water surface is necessary in many applications, such as surface water movement analysis, physical correct flow simulation and analysis. Water surface simulation with light effects greatly increases the realistic of the virtual scene, which goes prosperity in movies and games production. However, due to the special nature of liquids movement and randomness, computation of water leads to sophisticated algorithms and computationally intensive. Therefore, the simulation of water is an important part of research in the field of computer graphics.

Water modeling research has been undertaken for many years. The way of forming triangle mesh for visualization of the surface is the basis for further rendering. Particle system [1] defines each particle as a separated unit of physical movement, which has the basic light reflection, refraction, transmission capacity. In a particle system, collision detection is done between particles by extrusion and other complex covering analog computing water. Particle systems simulate realistic dynamic effects of water with high complexity; physics-based modeling of water refers to [2] and [3]. Furthermore, the methods based on Fast Fourier Transform (FFT) can get height field on each frame such as the approach proposed by Tessendorf [4] and Chiu [5], in which different amplitude and period of the sine wave summation are regarded as random height variations of the surface of the water. Premoze [6] and Hinsinger [7] put forward the related accumulated waveform and improved methods. Through 2D or 3D mesh modeling is a relevant fast simulation tool, but the realism of direct modeling is rather poor. Generally realistic rendering requires a combination of optimization methods which changing the grid resolution adaptively based on different viewing positions. LOD (Level-Of-Detail, LOD) model [8] divides grid resolution according to observation distance. LOD approach subdivides high resolution grid representation of the areas close to the observer. Subdivision mesh method of Dynamic quad-tree [9] uses the concept subdivision mesh of quad-tree. In addition，Perlin noise [10]and Navier-Stokes equation[11] are also effective methods to generate the random height of the water body. But Navier-Stokes equation is very complex and hard to be achieved; bump mapping can achieve fast simulation results， but it gets rough realistic. The simulation of simple light effect sees the literatures [12, 13]. Eric Bruneton[14] presents a illumination simulation method for real time realistic ocean surface. The core of this algorithm is evaluate the required representation based on view point, a hierarchical strategy combining geometry, normal and BRDF is being proposed, along with Fresnel term makes the results realistic for ocean. So far, a lot of methods are proposed based on ignoring the accurate reflection, refraction and water wave motion, thus it is not suitable for the observation and performance of the nearby water surface. In this paper, we will use the fractal Perlin noise combined with ray tracing technology to realize the simulation of the reflection and refraction of dynamic water.

The experimental results show that this method is an effective way to simulate the dynamic water surface. This paper discussed the generation of Perlin noise, the computing of reflection and refraction based on ray-tracing. Time consumption and simulation effects of the different octave and different ray-tracing maximal depth were conducted. The conclusions provide basis for accurate observation of the surface of the less distance objects and the accuracy of the calculation.

2. Perlin Noise

2.1. The Basic Principle of Perlin Noise

Perlin noise [10] is a natural noise generation algorithm presented by Ken Perlin, by specifying a value and a gradient for a random point in one dimensional or two dimensional spaces. The coordinate which is not the value of an integer in the noise image is calculated by the gradient’s bilinear interpolation. Perlin noise uses interpolation function such as the formula (1) or (2).

$$f(t) = 3t^2 - 2t^3 \tag{1}$$

$$\text{or} \quad f(t) = 6t^5 - 15t^4 + 10t^3 \tag{2}$$

Formula (2) can ensure the continuity of Perlin noise, so it is generally used as a noise generating function. Perlin noise has the characteristic of continuity; it is one dimensional function of continuous variation which changes over time. In this paper, we generate uniform 2D grid, and the level of detail is used based on the observer distance. The values of the non-integer values are obtained by interpolating the gradient values of the vertices of the grid. Mixing superposition of different octaves noise sequence, fractal noise is formed. Noise function is essentially a random number seed generator, with an integer value as a parameter, to return a random number based on the parameter. In the case of one dimensional, the random number among [0, 1] on the X axis is generated, then interpolate these random numbers smoothly. Figure 2 shows 2D Perlin noise visualization of different octaves.

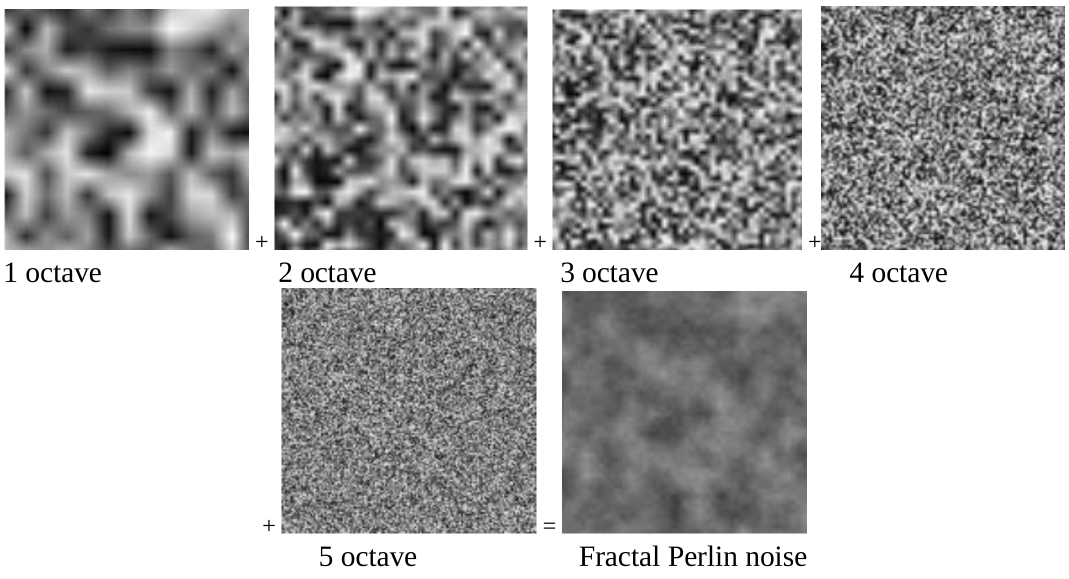


Figure 2. 2D Perlin Noise

2D Perlin noise is obtained by interpolation and gradient calculation in two directions of x and y. First, we calculate the gradient of the vertices of uniform grid, and then calculate the height of any sampling point P in the grid by bilinear interpolation. Figure 3 shows the 2D Perlin noise produced by Matlab, the production function as shown in the formula 3.

$$\begin{cases} f(x,y) = (3x^2 - 2x^3)(3y^2 - 2y^3) \\ f(x_1, x_2, \ldots x_n) = \prod_{i=1}^{n} 3x_i^2 - 2x_i^3 \end{cases} \tag{3}$$

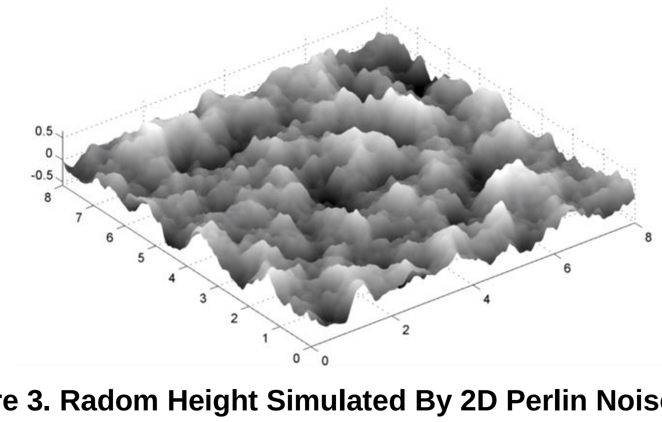


Figure 3. Radom Height Simulated By 2D Perlin Noise [15]

2.2. Level of Detail Model

According to the position of the camera, 2D grid model is optimized by LOD (Level of detail, LOD）. Since this algorithm will be combined with ray tracing algorithm, the complexity of the model optimization method is not mandatory. This method proposes a slight variation of LOD model based only on the distance d_i from observer to subdivide the grid resolution, which makes the position of the near observer have more accurate resolution of 2D grid and more accurate simulation results. The optimization principle is shown in Figure 4. In this algorithm, the maximum range of observation range of camera is d, so the value of d_i are obtained from far to near: $[\frac{d}{2}, d]$ $[\frac{d}{4}, \frac{d}{2}]$ and $[0, \frac{d}{4}]$ three sections, the resolution is double of the original resolution.

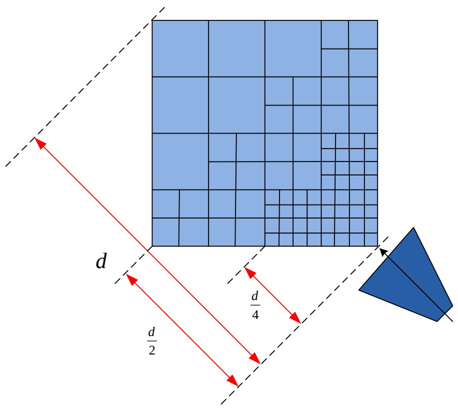


Figure 4. LOD Optimization

2.3. Generation of Surface Height

The algorithm generates test simulation result by different octaves of Perlin 2D noise. Figure 5 shows the surface effects which generated by the 1 to 6 octaves superimposed 2D Perlin noise.

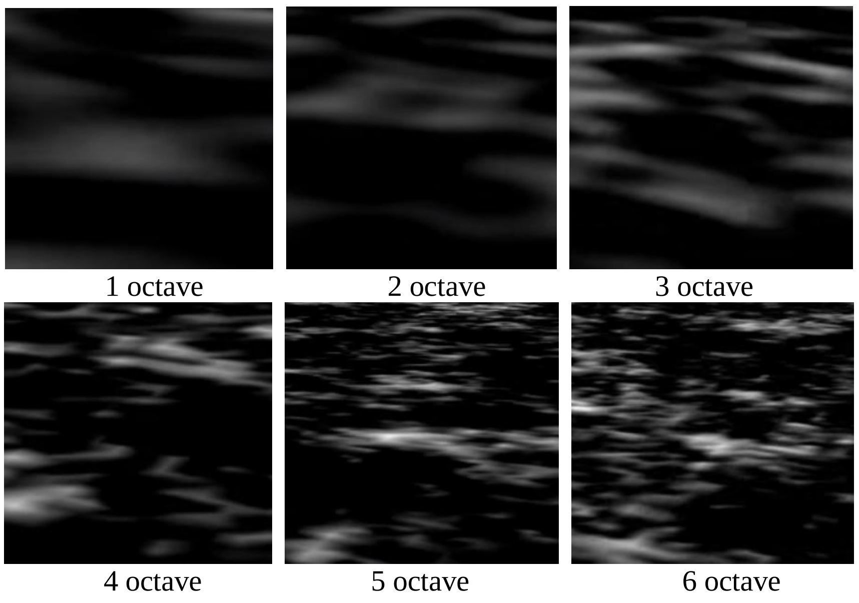


Figure 5. Various Octaves of Perlin Noise of Water Simulation

2.4. Refraction and Reflection

In order to obtain the color in the fragment shader, we calculate the normal vector through the surface of an arbitrary point. Set a surface point p the gradient values of p in direction of x and y are $\frac{\partial f}{\partial x}$ and $\frac{\partial f}{\partial y}$, the normalized vector of point p is $\vec{n}$, as shown in formula (4):

$$\vec{n}_P = normalize(\frac{\partial f}{\partial x} \times \frac{\partial f}{\partial y}) \tag{4}$$

According to the normal value, Snell's law is used to compute the refraction and reflection of different media interface, as shown in Figure 6:

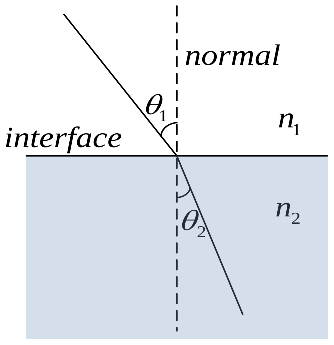


Figure 6. Principle of Surface Reflection and Refraction

$$\frac{\sin\theta_1}{\sin\theta_2} = \frac{n_2}{n_1} \tag{5}$$

In formal (5), n_1 and n_2 are the refractive index of the two medium, θ_1 and θ_2 are the angles between the incident light and the interface normal vector, and the refract light and the interface normal. When n_1 is equal to n_2, it is the reflecting situation, and the reflection value is shown as formula (6):

$$V_{reflect} = 1 + 2\cos\theta_1 n \tag{6}$$

According to formula (7) and (8):

$$\sin\theta_2 = \left(\frac{n_1}{n_2}\right)\sin\theta_1 = \left(\frac{n_1}{n_2}\right)\sqrt{1-(\cos\theta_1)^2} \text{ and} \tag{7}$$

$$\cos\theta_2 = \sqrt{1-(\sin\theta_2)^2} = \sqrt{1-\left(\frac{n_1}{n_2}\right)(1-(\cos\theta_1)^2)} \tag{8}$$

Refraction calculation formula is shown as (9)

$$V_{refract} = \left(\frac{n_1}{n_2}\right)l + \left(\left(\frac{n_1}{n_2}\right)\cos\theta_1 - \cos\theta_2\right)n \ , \ n = \{0,1\} \tag{9}$$

3. Ray Tracing

The basic idea of ray tracing is to track the light. After the light emitted from light source, it is reflected, refracted or absorbed by object surface or the pertinent media at intersection point, most of the light will not enter to the observer's eyes. So ray-tracing refers to tracking the casted light from the observer's eye to the scene pixels. This algorithm computes the intersection point of each light ray and the scene surface. Then light is analyzed and traced recursively of the reflection， refraction， and shadow depending on material of the intersection point. The framework of ray tracing is shown as algoritjm1. Ray tracing has more accurate simulation effect on the reflection and refraction, and it is often used when high quality simulation results is required. Parallel ray tracing algorithm [16,17] is proposed to solve the computational efficiency, also parallel engines are designed such as OptiX. Our method computes 1-3 iteration of reflection rays using standard ray-tracing and the running time is tested.

Algorithm1 Framwork of ray-tracing

```
Ray-Tracing
1 for(each pixel (sample) on the viewing area)
    2 if(ray-pixel intersection)
        3 select the front most intersection;
            4 recursively trace the refection and refraction  rays;
            5 calculate color
```

4. Implementation

The implementation environment is Intel ®Xeon(R) CPU E5620@ 2.40GHz, NVIDIA Quadro K5000 with GLSL. Figure 7 shows simulation results of superimposed fractal Perin noise of various octaves, where a and b are 6 octaves of Perin noise; c and d are 4 octaves of Perin noise, the maximum depth of iteration of ray tracing is 2. With the increasing of octaves, the water ripples gets finer and the reflection gets more precise to close the specular reflection.

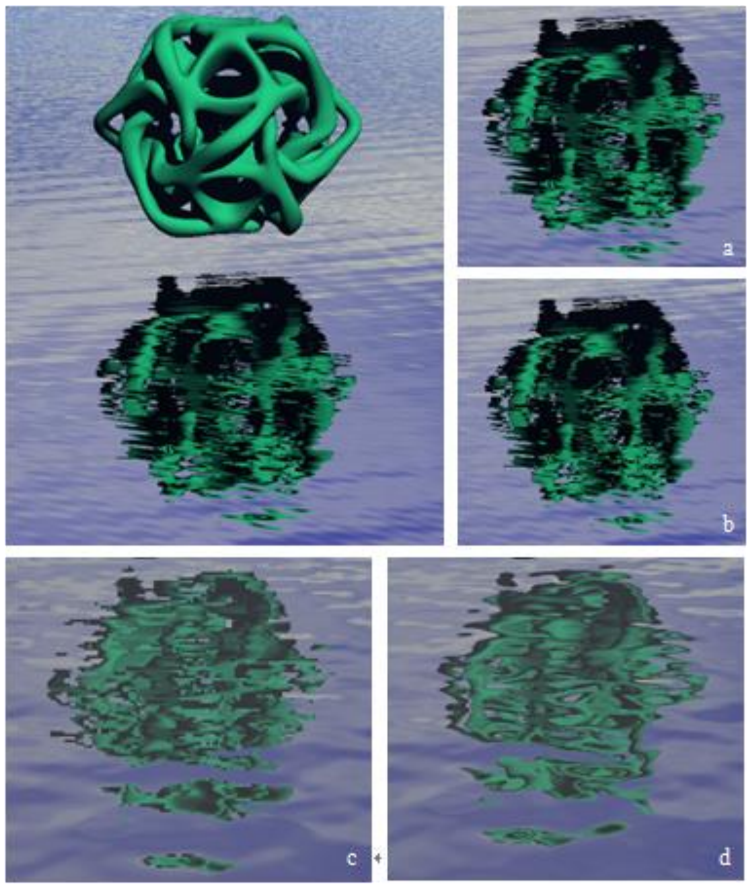


Figure 7 Effects of Various Superimposed Octaves and Reflection Simulation. a and b Shows the Effects of 6 Octaves Fractal Perlin Noise Plus Reflection at Intervals of 2 Frames; c and d Shows the 4 Octaves Fractal Perlin Noise Plus Reflection at Intervals of 5 Frames; the Maximum Iterative Depth of Ray-Tracing is 2

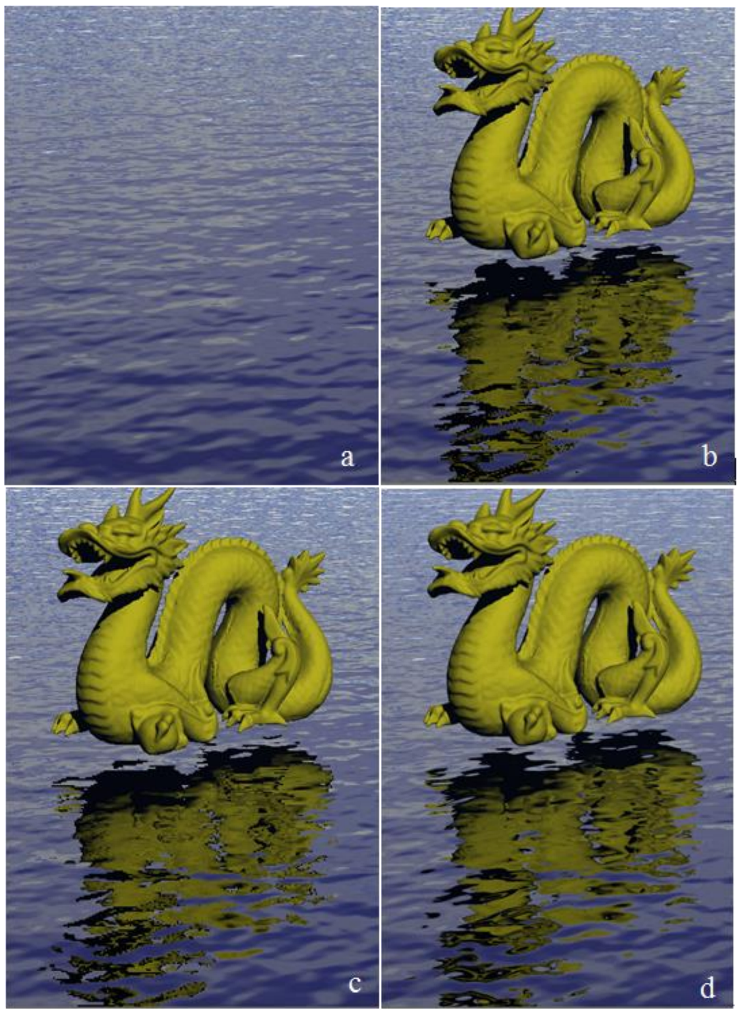
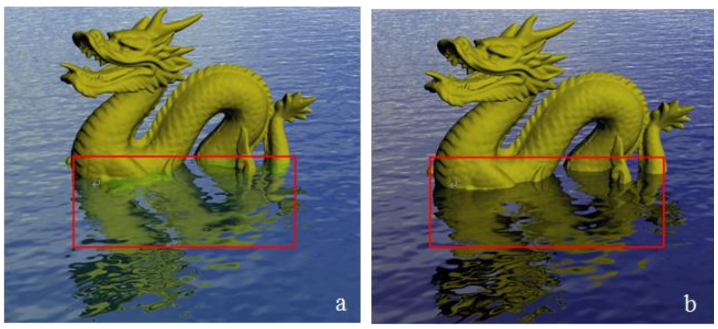


Figure 8 Effects of 6 Octaves and Reflection Simulation; a Shows the effect of 6 Octaves of Water Surface. b, c and d are 2,3 and 4 Maximum Iterative Depth of Ray-Tracing Respectively

Figure 8 shows the effect of 4 at different moment of dynamic water surface. The maximum iterative depth of ray-tracing is 3. With the increasing of the maximum depth of iteration, the reflection effect is more realistic. For different reflection depth, the running time of different models with various iterative depths is compared in Table 1. The computational efficiency of algorithm has relation with the numbers of ray tracing per pixel, the maximum depth of the iteration and the complexity of the models.



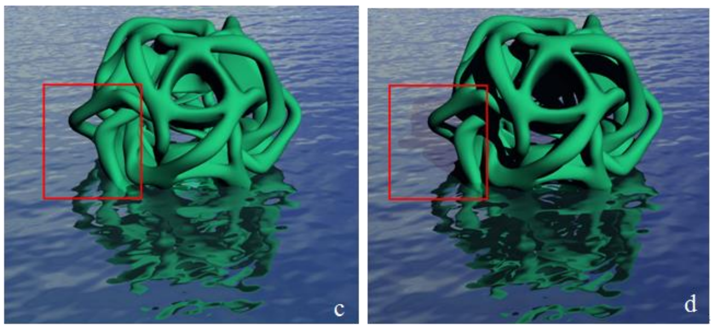


Figure 9. Different Effects of our Method; a. Reflection Plus Refraction; b. Reflection Without Refraction; c. reflection Plus Refraction Without Shadow Tests; d. Reflection Plus Refraction with Shadow Tests

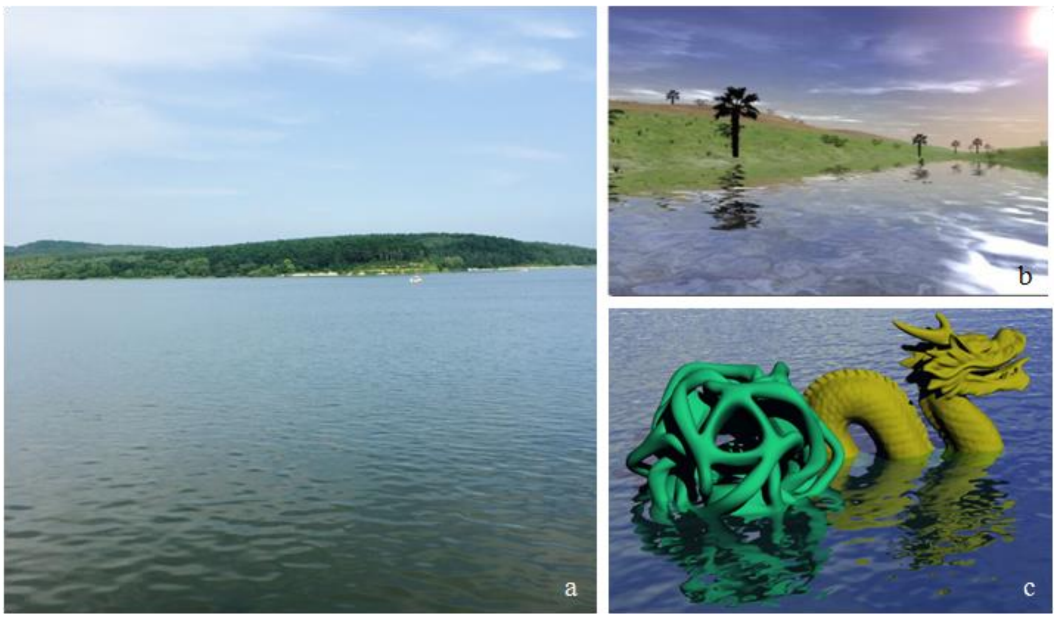


Figure 10. Comparison of Rene's Method and ours; a. Ground Truth; b. Rene's Implementation using Reflection Texture and Wave Distortion [18]. c. Our Method with Perlin Noise with Reflection, Refraction and Shadows

Overall, for our test models, the number of rays per pixel tracking fewer than 9, computational time can be tolerated. When the amount of lights per pixel tracking is more than 9, the computing time will be more than ten seconds. Finally, Figure 10 shows different effects of our method. The results show that our method is an effective way to simulate the water surface. In Figure 10, we compared the ground truth with Rene's method and ours, where Rene's implementation used reflection texture and wave distortion; however our method computed all lighting effects including reflection, refraction and shadows.

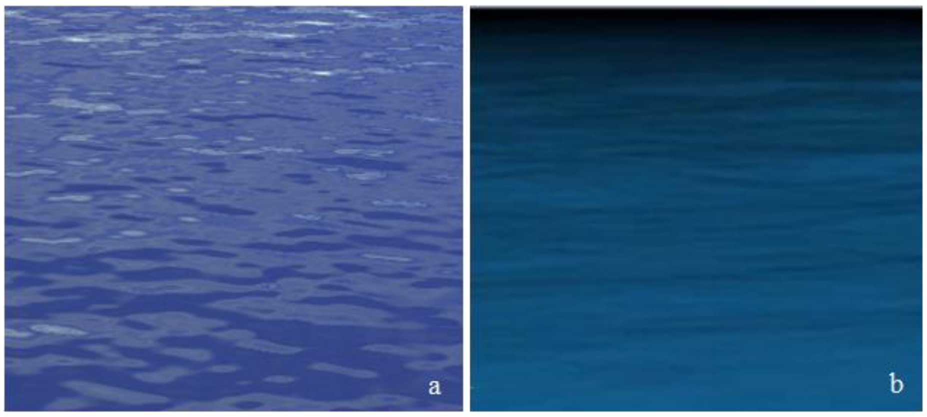


Figure 11. Comparison of Water Wave Simulation between ours and Existed Method. a. ours of Perlin Noise and Ray-Tracing, b. Mixed Method of FFT and Perlin Noise [19]

Finally, we compared the water wave of ours and a mixed method of FFT and Perlin noise [18]. Our method focuses on the reflection effects of objects close to the surface, so the water wave behavior according to the wind are not considerate. Figure 11 shows that different water surface derived by different motivation, while both of the techniques are based on Perlin noise. Although the ray tracing algorithm has high computational complexity, it is still necessary to calculate the photorealistic surface reflection for different rendering sceneries.

Table 1. Comparison of Running Time of Different Test Models（Unit：s）

Tested Model	Number of Triangles	Number of Rays (Per pixel)	Maximum Iterative Depth	Time
Dragon	2400000	2.64	2	20
		2.56	2	16
		9.06	3	57
Green ball	39936	2.64	2	7
		3.71	2	10
		9.33	2	24
		40.87	3	56

5. Conclusions

This paper presents a realistic simulation method of dynamic water surface which uses Perlin noise generating the water surface and computes reasonable reflection combined with ray-tracing technology. This method obtained precise visual effects with correctly calculated of surface reflection and refraction. Experimental result shows that when the testing ray of per pixel ranged from 2 to 4, and the maximum iterative depth the ray tracing ranged from 2 to 3, the running time can be controlled within 20 seconds. These are the acceptable time consumption and effect of simulation. The results have been applied in the movie engineering in practice, and it provided a reference and means for the related simulation and application (such as terrain simulations). Further work will simulates water surface in parallel based on OptiX to better improve the efficiency of simulation, or other alternative method to render the water surface fast.

Acknowledgement

This work was financially supported by development project of Jilin province science and technology (20140204009GX), Jilin upgrade industrial innovation special fund projects(2016C091) and major scientific and technological plan of Changchun (14KG008, 14KG013)

References

[1] Foster, N., & Fedkiw, R. Practical animation of liquids. In Proceedings of the 28th annual conference on Computer graphics and interactive techniques , ACM, **(2001)** August 23-30; Angeles, CA, USA

[2] Hasselmann, D. E., Dunckel, M., & Ewing, J. A. Directional wave spectra observed during JONSWAP 1973. Journal of physical oceanography, 10(8), 1264-1280 **(1980)**

[3] Ross, V., & Dion, D. Sea surface slope statistics derived from Sun glint radiance measurements and their apparent dependence on sensor elevation. Journal of Geophysical Research: Oceans, 112(C9) **(2007)**

[4] Tessendorf, J. Simulating ocean water. Simulating Nature: Realistic and Interactive Techniques. SIGGRAPH, 1(2), 5**(2001)**

[5] Chiu, Y. F., & Chang, C. F. GPU-based ocean rendering. In 2006 IEEE international conference on multimedia and expo, IEEE, **(2006)** July 2125-2128; Toronto, Ontario, Canada

[6] Premoze, S., & Ashikhmin, M. Rendering natural waters. In Computer graphics forum , **(2001)** December 189-199; Vienna Austria

[7] Hinsinger, D., Neyret, F., & Cani, M. P. Interactive animation of ocean waves. In Proceedings of the 2002 ACM SIGGRAPH, Eurographics symposium on Computer animation ,**(2002)** July161-166 San Antonio, Texas, USA

[8] De Boer, W. H. Fast terrain rendering using geometrical mipmapping.Unpublished paper, available at http://www. flipcode. com/articles/article geomipmaps. pdf **(2000)**

[9] Yang, X., Pi, X., Zeng, L., & Li, S. GPU-based real-time simulation and rendering of unbounded ocean surface. In Ninth International Conference on Computer Aided Design and Computer Graphics (CAD-CG'05), IEEE, **(2005)**December 6 ; Hong Kong, China

[10] Perlin, K. An image synthesizer. *ACM Siggraph Computer Graphics*,*19*(3), 287-296 **(1985)**

[11] Wang, Y., Baboulin, M., Dongarra, J., Falcou, J., Fraigneau, Y., & Le Maître, O. A parallel solver for incompressible fluid flows. *Procedia Computer Science*, *18*, 439-448 **(2013)**

[12] Fournier, A., & Reeves, W. T. A simple model of ocean waves. *ACM Siggraph Computer Graphics*, *20*(4), 75-84 **(1986)**

[13] Peachey, D. R. Modeling waves and surf. In ACM Siggraph Computer Graphics ACM. **(1986)** August 65-74 ; Dallas USA

[14] Bruneton, E., Neyret, F., & Holzschuch, N. Real - time Realistic Ocean Lighting using Seamless Transitions from Geometry to BRDF. In *Computer Graphics Forum* , Blackwell Publishing Ltd., **(2010)** May 487-496; New York, NY, USA

[15] https://www.youtube.com/watch?v=Or19ilef4wE

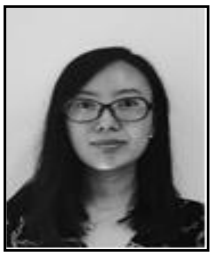
[16] Badouel, D., & Priol, T. An Efficient Parallel Ray Tracing Scheme for Highly. Advances in Computer Graphics Hardware V: Rendering, Ray Tracing and Visualization Systems, 93**(2012)**

[17] Wald, I., Woop, S., Benthin, C., Johnson, G. S., & Ernst, M. Embree: a kernel framework for efficient CPU ray tracing. ACM Transactions on Graphics (TOG), 33(4), 143 **(2014)**

[18] Truelsen R. Real-time shallow water simulation and environment mapping and clouds,**(2007)**

[19] Tian L. Ocean wave simulation by the mix of FFT and Perlin Noise, **(2014)**

Authors



Hua Li (1977.07-), she Graduated from Changchun University of Technology, Bachelor of Engineering in 2000.

Graduated from Changchun University of Technology, master's degree in computer science and technology in 2003.

PhD student since 2011, the main research direction is the realistic rendering and virtual realitytechnology.

Email:custlihua@gmail.com

Huamin Yang (1963.11-), He is a Professor, doctoral supervisor, main research direction is computer simulation technology. Email: yhm@cust.edu.cn

Jianping Zhao (1964.08-), he is a Professor, main research direction is image processing technology. Email: zhaojpin @aliyun.com