Journal of System Simulation

Volume 29 | Issue 10 Article 25

6-4-2020

Real-Time Modeling of Interaction between Floating Objects and Large-Scale Water

Xiaolong Yang
1. Marine College, Northwestern Polytechnical University, Xi'an 710072, China;;2. National Key Laboratory of Underwater Information Process and Control, Xi'an 710072, China;

Hongtao Liang
1. Marine College, Northwestern Polytechnical University, Xi'an 710072, China;;2. National Key Laboratory of Underwater Information Process and Control, Xi'an 710072, China;

Fengju Kang
1. Marine College, Northwestern Polytechnical University, Xi'an 710072, China;;2. National Key Laboratory of Underwater Information Process and Control, Xi'an 710072, China;

Gu Hao
1. Marine College, Northwestern Polytechnical University, Xi'an 710072, China;;2. National Key Laboratory of Underwater Information Process and Control, Xi'an 710072, China;

Follow this and additional works at: https://dc-china-simulation.researchcommons.org/journal

Part of the Artificial Intelligence and Robotics Commons, Computer Engineering Commons, Numerical Analysis and Scientific Computing Commons, Operations Research, Systems Engineering and Industrial Engineering Commons, and the Systems Science Commons

This Paper is brought to you for free and open access by Journal of System Simulation. It has been accepted for inclusion in Journal of System Simulation by an authorized editor of Journal of System Simulation.

Real-Time Modeling of Interaction between Floating Objects and Large-Scale Water

Abstract

Abstract: Large scale water is an integral part of nature and of high interest for interactive 3D applications, e.g., computer games and virtual environments. Allowing the water to interact with floating objects is essential for applications, but traditional height field interaction methods concentrate on water-to-body effects by letting water flow through the bodies. *A novel GPU-based method for rapid simulation of large scale water interacting with objects was proposed. A specially designed simulation grid was used to solve the interactions between dynamic objects and the surrounding environment. The model of rigid body was realized with a pre-rigid body method*. The method runs in real time for large areas of water even with a very limited GPU budget.

Keywords

large scale water, interaction, rigid body, simulation grid

Recommended Citation

Yang Xiaolong, Liang Hongtao, Kang Fengju, Gu Hao. Real-Time Modeling of Interaction between Floating Objects and Large-Scale Water[J]. Journal of System Simulation, 2017, 29(10): 2423-2431.

This paper is available in Journal of System Simulation: https://dc-china-simulation.researchcommons.org/journal/vol29/iss10/25

第 29 卷第 10 期
2017 年 10 月

系统仿真学报©
Journal of System Simulation

Vol. 29 No. 10
Oct., 2017

Real-Time Modeling of Interaction between Floating Objects and Large-Scale Water

Yang Xiaolong[1,2], *Liang Hongtao*[1,2], *Kang Fengju*[1,2], *Gu Hao*[1,2]

(1. Marine College, Northwestern Polytechnical University, Xi'an 710072, China;
2. National Key Laboratory of Underwater Information Process and Control, Xi'an 710072, China)

Abstract: Large scale water is an integral part of nature and of high interest for interactive 3D applications, e.g., computer games and virtual environments. Allowing the water to interact with floating objects is essential for applications, but traditional height field interaction methods concentrate on water-to-body effects by letting water flow through the bodies. *A novel GPU-based method for rapid simulation of large scale water interacting with objects was proposed. A specially designed simulation grid was used to solve the interactions between dynamic objects and the surrounding environment. The model of rigid body was realized with a pre-rigid body method.* The method runs in real time for large areas of water even with a very limited GPU budget.

Keywords: large scale water; interaction; rigid body; simulation grid

大规模水体与物体实时交互可视化仿真

杨小龙[1,2]，梁洪涛[1,2]，康凤举[1,2]，顾浩[1,2]

（1. 西北工业大学航海学院，西安 710072；2. 水下信息处理与控制国家级重点实验室，西安 710072）

摘要：大范围水体是大自然的重要组成部分，对于诸如计算机游戏和虚拟环境的交互式 3D 应用来说是非常重要的。模拟水体与漂浮物体相互作用对于实际应用来说是必不可少的，但是传统基于高度场的交互方法大多采用让水体穿过物体发生作用。提出*一种新型的基于 GPU 的快速模拟大范围水体与物体的交互方法，设计一种特殊的仿真网格来解决动态对象与周围环境之间的交互*，刚体模型采用预刚体化算法实现。仿真结果表明在 GPU 资源预算有限的情况下，该方法也可进行对大规模水体的实时仿真。

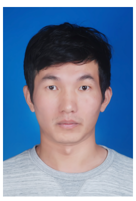
关键词：大范围水体；固液交互；刚体；仿真网格

中图分类号：TP391.9　　文献标识码：A　　文章编号：1004-731X (2017) 10-2423-09

DOI: 10.16182/j.issn1004731x.joss.201710025

Introduction

Water is a common and important element in nature. It is also frequently encountered in games and other 3D virtual environments. A lot of attention has been devoted to rendering water realistically, but interaction with it is typically limited in games. Bodies of water are often simply modeled as static planes with possibly some procedural waves that have no effect on gameplay and therefore no interactivity. Interaction is sometimes possible with small amounts of particle-based liquids, but they are



Received: 2017-05-20　**Revised:** 2017-07-18;
Foundation items: Northwestern Polytechnical University Doctoral Dissertation Innovation Fund (CX201701).
Biography: Yang Xiaolong (1988-), Male, Linyi, Shandong, China, Ph.D. Research direction for system modeling and simulation.

Published by Journal of System Simulation, 2017

also more commonly used only for visual effect.

Interactive systems are at the heart of gaming. Off-the-shelf rigid body physics solvers have revolutionized 3D environments in part because they provide so many natural interaction possibilities. Similar interaction with large amounts of water is currently not possible in the real-time 3D realm. A more versatile interaction of rigid body physics and large-scale bodies of water could enrich virtual worlds tremendously and even enable completely new game genres.

Performance is one of the main reasons why large-scale water areas are still static in most game worlds. Large-scale fully 3Dwater is still out of reach, but the continuing increases in the available parallel GPU computing power are making some simpler methods fast enough for consideration.

In this paper, we proposed a novel method for rapid simulation of open water interacting with objects based on GPU. Our main contributions can be summarized as follows:

1. To model the rigid body efficiently, a pre-rigid body method was proposed to achieve plausible visual results at higher rendering rates.

2. The interactions between dynamic objects and the surrounding environment were realized with a specially designed simulation grid. PML (Perfectly Matched Layers) method was further introduced to ensure the continued stability of the simulation grid's boundary fluctuations.

3. Based on the above techniques, realistic open water environments interacting with objects can be generated stably at higher rendering rates.

The rest of the paper is structured as follows. Section 2 introduces some related work. Our simulation method, including body-to-water effects, is described in detail in Section 3. Section 4 discusses the method of rigid body dynamics simulation. Section 5 evaluates the results and Section 6 concludes.

1 Related Work

Fluid simulation has a long tradition in engineering. However, most methods from this literature strive for realism, for example, bridge building, and are much too slow for games. In computer graphics, water simulation started to become popular in the 1990s with, for example, the work of Foster and Metaxas[1], but the main application has been in special effects for movies, with processing times for single frames often measured in minutes. We concentrate on methods that are suitable for real-time purposes. For a good introduction on the more realistic approaches, see Bridson's book[2] for the Eulerian approach and the recent review of the SPH literature for the Lagrangian perspective[3].

Currently, the methods of interactive simulation of open water environment can be classified into two categories: heuristic-based method and physically based method. For the former one, Fast Fourier Transform (FFT) spectrum analysis method and Perlin noise height field method (Peachey[4]; Tessendorf[5]; Johanson[6]; Darles et al.[7]) are typical. In these methods, the motion of water surface is regarded as a wave generated on a flat elastic membrane, which can be described using a 2D wave equation simply. By calculating the partial derivatives using the central difference method, the movement of the wave can be modeled. However, it is difficult to simulate the details (such as foams, bubbles and shocking wave) of water surface with heuristic-based method. Although they can be modeled using particle systems, the results are less

https://dc-china-simulation.researchcommons.org/journal/vol29/iss10/25
DOI: 10.16182/j.issn1004731x.joss.201710025

realistic. The interactions between the water surface and objects are also hard to be modeled using the above heuristic-based method.

To handle the above problems, physically based methods (Kass and Miller[8]; Stam[9]; O'Brien and Hodgins[10]; Yuksel, et al.[11]) were proposed. These methods come from fluid mechanic's theory, which can generate more realistic motion of water surface. The Navier–Stokes equations in computational fluid dynamics are usually adapted to express most types of fluid motion including the dynamic ocean. For the calculation of Navier–Stokes equations, Euler-based method and Lagrange-based method were proposed. As solving Navier–Stokes equations is time-consuming, some simplified methods were developed. The shallow water equation was introduced to approximate the Navier–Stokes equations, which only described the motion of water in horizontal direction. Thus, it reduces the computation to one dimension and simplifies the calculation. On the other hand, the shallow water equation can be combined with height field easily to model the fluid–solid coupling. O'Brien and Hodgins[10] described the fluid velocity in the vertical direction and realized the splatting effects of water particles. Lagrange-based method is a type of nongrid-based particle system method, which can be used to simulate the surface distortions, the liquid separation and sprays in dynamic water body efficiently and realistically. Yuksel, et al.[11] proposed the wave particles method to represent height field, making the simulation more efficient and easy to implement. Yan, et al.[12] proposed a physically based wave scene rendering method. They used smoothed particle hydrodynamics (SPH) method to model the curved waves on GPU. Compared with the heuristic method, physically based method requires large amount of calculation, it can produce better rendering results.

Another related topic is simulation of the interactions (Baraff[13]) between fluid and objects such as floating objects in water and flying flag in wind. Wejchert and Haumann[14] simulated flying leaves in wind based on combination of basic flow elements. Wei, et al.[15] adopted Lattice Boltzmann Method (LBM) to simulate the motion of bubbles and feathers in wind. Fedkiw, et al.[16] simulated realistic smoke scene. They also rendered the interactions between smoke and obstacles made of simple geometries like cuboids and columns. Yngve, et al.[17] proposed a method for simulation of explosion and its damage on environment objects. The motion of shock wave was expressed by compressible Navier–Stokes equations. Gomez[18] proposed a rigid body dynamics method for simulation of dynamic water surface with floating object. As this method needs to recalculate the cross-product of the angular momentum and angular velocity in each calculation cycle, it is very time-consuming. Cords and Staadt[19] simulated the interactions between water surface and moving boat. By sampling the particles on the boat surface, it can render the interactions in real time. Carlson et al.[20] proposed a method for simulation of two-way interaction between rigid objects and fluids. In their method, they first resolve the occupied area by fluid, and then the rigidity and rigid motion are enforced by applying the deformation-free constraints. The results were wonderful. However, their method was too complex when handling thin objects. Genevaux, et al.[21] addressed the problems of two-way interactions between fluids and rigid thin shells. This method can also be extended to simulate interactions between fluids and deformable materials.

2 Simulation

Simulating large water environments in real-time with highly detailed waves everywhere is not feasible. Hence, we only simulate the liquid surface where fine detail is needed (e.g., in the wake of moving water crafts) and animate it in regions with no object–water interaction. Most ocean waves are created by wind and move independently. We classify waves into interactive waves when interaction with objects occurs and ambient waves otherwise Ambient Waves For ambient wave animation, we use the effective method of Tessendorf[5]. This results in realistic wave propagation for open water. The model is based on statistical measurements of real ocean environments. The frequency spectrum is generated by the Phillips spectrum, supporting handling of wind direction and wind strength. Since this method generates tiled height fields, several fields with different aperiodic scales can be used to generate an infinite ocean surface, where the tiling is not visible.

2.1 Ambient Waves

For ambient wave animation, we use the effective method of Tessendorf[5]. This results in realistic wave propagation for open water. The model is based on statistical measurements of real ocean environments. The frequency spectrum is generated by the Phillips spectrum, supporting handling of wind direction and wind strength. Since this method generates tiled height fields, several fields with different aperiodic scales can be used to generate an infinite ocean surface, where the tiling is not visible.

The wave equation describes the propagation of waves at time t and position $\mathbf{x} \in R^2$. A 2D linear partial differential equation (PDE) can be used for simulating radial wave propagation of liquid surface waves:

$$\Delta \mathrm{f}(\mathbf{x},t) - \frac{1}{c^2}\frac{\partial^2 \mathrm{f}(x,t)}{\partial t^2} = 0 \quad (1)$$

here, $\Delta=\nabla^2=\sum_1^2 \frac{\partial^2}{\partial x_i^2}$ is the Laplacian in 2D and c is the velocity at which waves propagate across the surface. This PDE can be solved in a fast and straightforward manner using an Eulerian grid-based approach. Boundary conditions have to be included for collision objects as well as for the boundary of the simulation grid. Such bounding conditions can be modeled within the wave equation directly. Adapting $f(\mathbf{x},t_i)$ to constant values at the boundary of collision objects within the liquid models objects and, hence, the corresponding reflection of waves. Radially propagating waves can be created at any position $\mathbf{x}_0$ and time t_i by displacing $f(\mathbf{x}_0,t_i)$. Displacing several $f(\mathbf{x}_i,t_i)$ can create, for example, large waves or wave trains.

Additional maps that influence the simulation of the wave equation can be used to model specific behavior. In shallow water, for example, velocity depends on water depth. Hence, c can be varied according to depth and described by a velocity map. The same applies to damping and a damping map may describe the damping according to the type of ground.

2.2 Discrete wave equation

The discretization of the wave equation (Equation 1) using a finite difference method can be carried out using a 2D map $z_{ij}=f(i,j)$, with $i,j \in [0,size]$ and step size $h = 1/size$. Using central differences, the discretization leads to:

$$z_{i,j}^{t+1} = a \cdot \sum_{\substack{k=i\pm1,l=j;\\ k=i,l=j\pm1}} z_{k,l}^t + (2-4a)\cdot z_{i,j}^t - z_{i,j}^{t-1} \quad (2)$$

with $a = \frac{c^2 \Delta t^2}{h^2}$ and time step Δt. Hence, the wave equation is solved for a rectangular area.

For smooth movement of the simulation grid in

https://dc-china-simulation.researchcommons.org/journal/vol29/iss10/25
DOI: 10.16182/j.issn1004731x.joss.201710025

surface space, we define a translation $\mathbf{t} \in R^2$. Note that a translation with non-integer numbers would result in large numerical diffusion due to the necessary dissipation step. The translated grid elements have to be refiltered onto the discrete grid values within each time step, resulting in diverging wave trains. To overcome this problem, we split the translation $\mathbf{t}$ into the integral part $\mathbf{t}_{int} = (t_{int}^x, t_{int}^y)^T \in N^2$ and the fractional part $\mathbf{t}_{frac} \in R^2$ with $0 \leqslant t_{frac}^x, t_{frac}^y \leqslant 1$. Since the numeric solution of Equation 2 depends on the current and the previous time step, the integral translation $\mathbf{t}_{int}^{old}$ of the previous time step is stored for reuse. Consequently, the discrete wave equation is then changed to

$$z_{i,j}^{t+1} = a \cdot \sum_{\substack{k=i\pm1,l=j;\\ k=i,l=j\pm1}} z_{k,l}^t + (2-4a) \cdot z_{m,n}^t - z_{o,p}^{t-1} \quad (3)$$

with $m = i - t_{int}^x$, $n = j - t_{int}^y$, $o = m - t_{int}^{xold}$ and $p = n - t_{int}^{yold}$. The simulation grid is accessed in surface coordinates, adding the fractional translation $\mathbf{t}_{frac}$. Hence, numerical dissipation is suppressed, due to the use of a discrete grid. Transformation within the grid is carried out on integer basis. However, the grid can still be moved smoothly, because access of coordinates works on a fractional basis. To scale the surface in x or y direction by s_x and s_y, the translation vector is scaled by $(1/s_x, 1/s_y)^T$.

The described simulation method can be executed efficiently on a GPU. Using a 2D grid, the method can be implemented within a fragment shader using textures. A general purpose GPU framework, such as CUDA, can be used too. Yet the texture-based point of view lets us set the boundary conditions efficiently by rendering directly into the simulation grid.

Stability we used an explicit approach for solving the wave equation, we got the following stability constraints according to explicit finite difference simulation methods. The simulation becomes unstable, if condition $c^2 \Delta t^2 / h^2 \leqslant 0.5$ is not adhered and hence, the height field grows exponentially. Yet, the maximum wave propagation speed can be determined by the given condition.

2.3 Multiple Grids

The above method allows for placing the simulation grid at an arbitrary position on the infinite water surface. The grid can follow moving objects and the simulation grid can follow the view frustum. We describe the use of several simulation grids for adaptive simulation below.

2.4 Different Locations

In some scenarios, it may be of importance to simulate the surrounding of more than one object or the area within the view frustum only. Our method can be expanded to support multiple moving grids. Even more, each grid can be moved independently as depicted in Figure 1. Final height-field values can be determined by superposition of all overlapping grids. Simulation grid sizes are adapted according to the importance of the actual simulation. Hence, waves are simulated where necessary. As a result, we achieve a fast, environment specific simulation. For example, different moving boats can have their own simulation grid and, hence, full interaction with surrounding liquid. Simulation grid sizes may vary according to the importance of details within that grid.

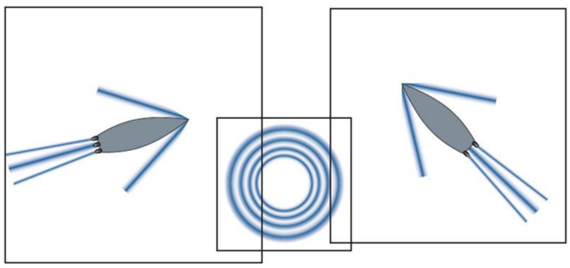


Fig. 1 Several simultaneous wave simulations are used within the infinite ocean surface.

Interference of waves is a fundamental physical principle and result in superposition of amplitudes (not intensities). Since waves are interfering without any interaction, different grids can be simulated independently—wave interaction between different grids does not occur. Hence, objects in different grids have to be coupled to their belonging simulation grid only and the final height field can then be determined by superposition. Take note that the independence of simulation grids is important for efficient execution, since no data has to be transferred between simulation grids. Height field values are scaled down smoothly to zero at the boundaries of moving grids to guarantee a soft transition to the height values of ambient waves or interfering simulation grids. A static grid can be used, if the wave propagation within a static region is of importance.

2.5 Different Scales

Our approach can not only be used for adaptive simulation at different locations, but also at different scales. Since multiple grids can be used, one grid can be used to represent high details if the camera comes close to the liquid's surface. For example, in a multiple grid simulation, one tiled grid simulates the detailed wave propagation of a rain drop hitting the water surface and another grid simulates the wave propagation of a moving boat (see Figure 1). If different scales are used, the simulation grid's translations have to be normalized: They are divided by their resolution—otherwise different grids are moving with different velocities.

Another possibility is the use as an LOD-Simulation. Depending on the distance to the camera, simulation grid size can be decreased, resulting in less details and faster simulations for objects far away.

3 Rigid body dynamics simulation

To simulate objects' motion on water surface, it is important to calculate the rotation. When the angular momentum and angular velocity are in different directions, even if no force is applied, there will still be a rigid body angular acceleration which is called precession in the field of physics. Based on the condition that the angular momentum and the angular velocity are in the same direction, which is called non-precession motion, we proposed a novel efficient rigid body dynamics simulation method.

According to the rigid body angular momentum theorem, the rotation of a single rigid body can be expressed using the angular acceleration formula as follows:

$$\frac{d\omega}{dt} = I^{-1}[N_{net} - \omega \times (I\omega)] \tag{4}$$

where I is the rigid body inertia tensor, N_{net} denotes the combined effect of rigid body moment, and ω is the angular velocity. Assume that the angular momentum and the angular velocity of the rigid body are identical in direction; then, the item $\omega\times(I\omega)$ will be zero. Thus, the calculation of the angular acceleration can be simplified as the following:

$$\frac{d\omega}{dt} = I^{-1}N_{net}, I^{-1} = RI_p^{-1}R^{-1} \tag{5}$$

As the composition of the spindle orthonormal basis vectors R (spindle coordinate system) represents the local space, the inertia tensor matrix can be expressed as $I_p = \begin{pmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{pmatrix}$, where I_{xx}, I_{yy}, I_{ss} are three main moments of inertia for I_p.

A solution of the above equation is $I_{pe}=I_{xx}=I_{yy}=I_{zz}$. As I_{pe} is a scalar, we can get that $I^{-1} = I_p^{-1} = I_{pe}^{-1}$. So the Formula 5 can be further adapted as

https://dc-china-simulation.researchcommons.org/journal/vol29/iss10/25
DOI: 10.16182/j.issn1004731x.joss.201710025

$$\frac{d\omega}{dt} = I^{-1}N_{net} = N_{net}/I_{pe} \tag{6}$$

Under the condition that the angular momentum and the angular velocity are in the same direction, that is $\omega\times(I\omega)=0$, we solve the geometry values of the rigid body in a backward induction way. Next, we will detail on our calculation strategy.

For a rigid body with constant density like a boat, it is similar in geometry with the symmetrical objects such as square, sphere, cylinder and gyropendulum. Due to the symmetry axis of these rigid bodies, their products of inertia are zero. So the symmetry axes of them can be regarded as the spindle.

In this paper, we adopt the assumption that the object's geometry can be approximated using a hollow cylindrical geometry, and then replace the object with a hollow cylindrical body as an applied force object. Eventually, the calculated parameters for the applied force object will be used for the simulation object, making them have the same velocity and acceleration. There are several force contact points on the surface of the hollow cylinder with the same distances to its center of mass, symmetrically distributed on both sides of the hollow cylinder, as shown in Fig. 2.

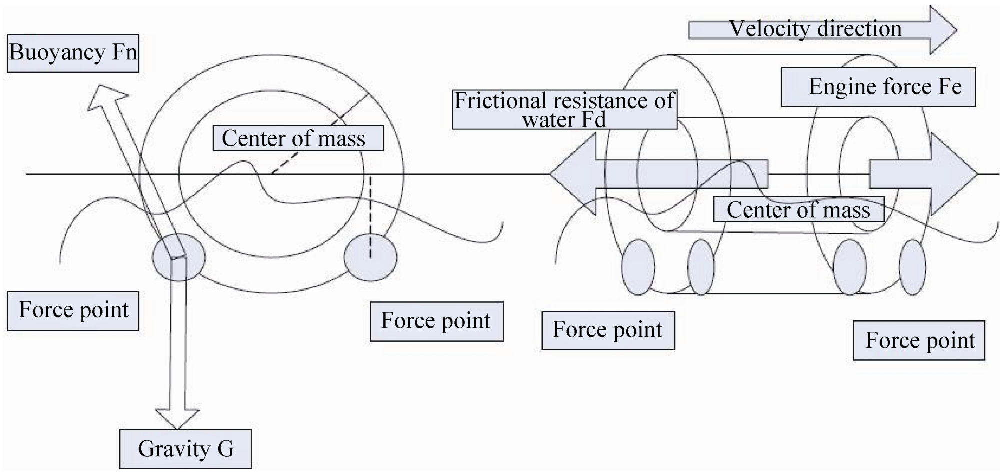


Fig. 2 Force analysis for the rigid body contact points

We applied a linear engine force Fe on the center of mass to speed the rigid body, which is in the same direction with its velocity. It can maintain the boat's movement. If there are n force points on the rigid body, the combined force for rigid body is

$$F = \sum_{i=1}^{n}(Fn_i + m_i g + Fd_i + Fc_i) + Fe \tag{7}$$

where Fd is the frictional force of water, m is the mass of the rigid body. Then, we can calculate the displacement acceleration as $a = F/M\ (M = \sum_{i=1}^{n} m_i)$. The total resultant moment is

$$N_{net} = \sum_{i=1}^{n}((Fn_i + m_i g + Fd_i + Fc_i)\times r_i) \tag{8}$$

where r_i is the distance between the center of mass and each force point.

Note that the ratio of the inner to outer radius of the hollow cylindrical is μ; the main moment of inertia for the hollow cylinder can be calculated as follows:

$$I_{xx} = I_{yy} = \frac{1}{12}ML^2 + \frac{1}{4}MR^2(1+\mu^2) \tag{9}$$

$$I_{zz} = \frac{1}{2}MR^2(1+\mu^2) \tag{10}$$

with Eqs. (6), (9) and (10), we can determine that the outer radius of the hollow cylinder is R=0.5L, which meets the torque-free precession condition. The moment of inertia can thus be calculated as

I_{pe}=0.905MI^2. Finally, the angular acceleration can be obtained with the Formula (6). The simulation efficiency is much improved due to the simplification of the calculation of angular acceleration.

4 Simulation results

Based on the above model, we have generated various scenes of water interacting with moving boats. All experiments were run on a PC with Intel 3.10 GHz CPU, Nvidia GeForce GT440 GPU. The software platform is based on DirectX 9.0c. The sizes of the projected grid, the shift grid and the FFT grid are 128×128, 512×512 and 1 024×1 024, respectively. The resolution of the rendering images is 1,440*900. We have tested the time efficiency of our method and compared it with previous methods. Fig. 3 shows the scene of rigid body dynamics in the testing experiment.

Fig. 3 Scene of the rigid body dynamics testing experiment

Fig. 4 shows the efficiency comparison between Cords and Staadt[19] and our method, from which we can see that our algorithm is more efficient. When the number of the simulation rigid bodies (Gizmos) is more than ten thousand, our algorithm achieves a more obvious advantage over that in Cords and Staadt[19]. In comparison with the method in Cords and Staadt[19], our method is more stable and achieves more plausible simulation result as we adapted the PML boundary layer method to eliminate the boundary shear instability and reduce the wave reflection of the grid boundary.

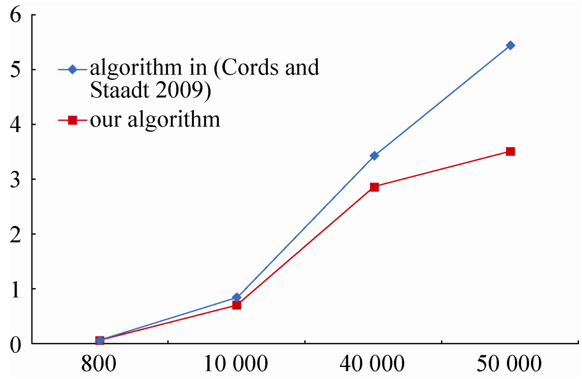


Fig. 4 Algorithm efficiency comparison of simulation time (ms) of rigid body dynamics

5 Conclusion and future work

This paper has proposed a novel efficient method for simulation of the dynamic interactions between rigid bodies and water. This method realized the simulation of the motion of water surface, the interactions between rigid bodies and water, and realistic shading of water surface, etc. It can achieve high rendering rates on common PCs, with potentials in interactive or real-time applications such as video games and on-line training system. This method is easy to implement, which can be imported to the popular simulation engines such as OGRE, Hydrax and Caelum Plug-ins. Our method achieves high rendering rates at the cost of simplifying some details including spray and splashing effects. In the future work, we will introduce particle-based method into our simulation framework to generate scenes with more details. We plan to include the effects of spray, break waves, etc., into our simulation in order to add more realism of the result. To extend the pre-rigid body method to objects with non-symmetrical geometry is another goal of our future work. We will also improve the rendering speed of our method in the future work. To integrate with the multi-core

https://dc-china-simulation.researchcommons.org/journal/vol29/iss10/25
DOI: 10.16182/j.issn1004731x.joss.201710025

technique and improve the collaboration of CPU-GPU might be a feasible way.

Reference:

[1] Foster N, Metaxas D. Realistic animation of liquids [J]. Graphical models and image processing (S1077-3169), 1996, 58(5): 471-483.

[2] Bridson R. Fluid simulation for computer graphics [M]. London: CRC Press, 2015.

[3] Ihmsen M, Orthmann J, Solenthaler B, et al. SPH fluids in computer graphics [J]. Eurographics 2014 - State of the Art Reports (S1017-4656), 2014,2(1):21-42.

[4] Peachey D R. Modeling waves and surf [C]// ACM Siggraph Computer Graphics. USA: ACM, 1986: 65-74.

[5] Tessendorf J. Simulating ocean water [J]. Simulating nature: realistic and interactive techniques, SIGGRAPH (S2092-6731), 2001, 1(2):1-27.

[6] Johanson C. Real-time water rendering - introducing the projected grid concept [D]. Sweden: Lund University, 2004.

[7] Darles E, Crespin B, Ghazanfarpour D, et al. A survey of ocean simulation and rendering techniques in computer graphics [C]// Computer Graphics Forum. USA: Blackwell Publishing Ltd, 2011: 43-60.

[8] Kass M, Miller G Rapid. Stable fluid dynamics for computer graphics [C]// ACM SIGGRAPH Computer Graphics. USA: ACM, 1990: 49-57.

[9] Stam J. Stable fluids [C]// Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques. USA: ACM Press/Addison-Wesley Publishing Co., 1999: 121-128.

[10] O'brien J F, Hodgins J K. Dynamic simulation of splashing fluids [C]// Computer Animation'95, Proceedings. USA: IEEE, 1995: 198-205.

[11] Yuksel C, House D H, Keyser J. Wave particles [C]// ACM Transactions on Graphics (TOG). USA: ACM, 2007: 99:1-8.

[12] Yan H, Wang Z, He J, et al. Real-time fluid simulation with adaptive SPH [J]. Computer Animation and Virtual Worlds (S1546-4261), 2009, 20(2/3): 417-426.

[13] Baraff D. An introduction to physically based modeling: rigid body simulation I—unconstrained rigid body dynamics [J]. SIGGRAPH Course Notes (S2092-6731), 1997, 1:1-32.

[14] Wejchert J, Haumann D. Animation aerodynamics [C]// ACM SIGGRAPH Computer Graphics. USA: ACM, 1991: 19-22.

[15] Wei X, Zhao Y, Fan Z, et al. Blowing in the wind [C]// Proceedings of the 2003 ACM SIGGRAPH /Eurographics Symposium on Computer Animation. France: Eurographics Association, 2003: 75-85.

[16] Fedkiw R, Stam J, Jensen H W. Visual simulation of smoke [C]// Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques. USA: ACM, 2001: 15-22.

[17] Yngve G D, O'Brien J F, Hodgins J K. Animating explosions [C]// Proceedings of the 27th annual conference on Computer Graphics and Interactive Techniques. USA: ACM Press/Addison-Wesley Publishing Co., 2000: 29-36.

[18] Gomez M. Interactive simulation of water surfaces [M]// Game Programming Gems. USA: Charles River Media, 2000, 1: 187-194.

[19] Cords H, Staadt O G. Real-Time Open Water Environments with Interacting Objects [C]// NPH. Munich: The Eurographics Association, 2009: 35-42.

[20] Carlson M, Mucha P J, Turk G. Rigid fluid: animating the interplay between rigid bodies and fluid [J]. ACM Transactions on Graphics (TOG) (S0730-0301), 2004, 23(3): 377-384.

[21] Génevaux O, Habibi A, Dischler J M. Simulating Fluid-Solid Interaction [C]// Graphics Interface, 2003. Canada: Canadian Information Processing Society, 2003: 31-38.