water

MDPI

Article

Simulation of Fluid and Complex Obstacle Coupling Based on Narrow Band FLIP Method

Changjun Zou and Yong Yin *

Navigation College, Dalian Maritime University, Dalian 116026, China; zoucj2006@163.com
* Correspondence: bushyin@163.com; Tel.: +86-0411-8472-3925

Received: 1 May 2018 ; Accepted: 11 June 2018; Published: 19 June 2018

Abstract: With the continuous development of fluid simulation theory and technology, there are increasingly higher requirements for simulation of complex fluid interaction. Fluid simulation based on the Eulerian method is limited by the grid resolution, and the sawtooth phenomenon occurs near the obstacle boundary. To enhance the fluid interaction performance with complex obstacle, an advanced fluid interaction method was proposed based on NBFLIP. Improved from FLIP method, the NBFLIP method combines the advantages of Euler method and Lagrangian method. The SDF method is proposed in complex obstacle discretion, with an expectation to facilitate the processing with obstacle boundary and efficiency improvement. Compared with FLIP method, particle number in NBFLIP method is reduced by 86.2% and the average running time per frame is reduced by 36.1%.

Keywords: Narrow Band FLIP; fluid simulation; free surface; complex obstacle; SDF

1. Introduction

Fluid interaction simulation is not only of great importance in computer graphics applications and research fields, but also is also a hot topic in the computer graphics community. It is widely used in film special effects, computer games, virtual military exercises, virtual education and other fields. With the upgrade of computer hardware, it has become an urgent requirement on the measures taken how to simulate fluid and its interaction with obstacles more efficiently and realistically. Traditional fluid modeling methods include Eulerian method and Lagrangian method. In the Euler method, it is easier to reconstruct the free surface and more straight forward to implement of Navier-Stokes Equation (NSE) discretization. However, it is not suitable for simulating fluid of large deformation. The advantages of Particle-based Lagrangian method lie in energy conservation and small-scale details preserving, but the surface reconstruction is difficult and the searching for neighboring particles is costly.

Hence, the approach of combining Euler method and Lagrangian method in fluid simulations has been explored by many scholars. The FLIP (Fluid Implicit Particle) method was firstly introduced into fluid simulation by Brackbill [1,2], who used a complete Lagrangian method for convective terms. The grid velocity is updated from the accumulation of the grid velocity interpolated from the particles inside. In this way, the FLIP method almost eliminates all the numerical dissipation and ensures mass conservation , thus being widely used in fluid simulation. In the FLIP method, all grids are filled with particles for velocity advection, but many of them make no contribution to the surface reconstruction. Based on this fact, Ferstl [3] proposed a fluid simulation method called the Narrow Band FLIP (NBFLIP) method, which reduces the computational effort by resampling the particle in the narrow band without degrading the details of fluid simulation. The improved NBFLIP method has shown great potential in fluid simulation [3]. To simulate the interaction between fluid and complex obstacles as well as free surfaces tracking, a fluid simulation method based on NBFLIP which discretizes the complex obstacle boundary based on Signed Distance Function (SDF) is proposed in this paper. The free surface

is tracked by LevelSet method based on SDF, thus rich phenomena of free surface and fluid interaction with complex obstacle are achieved.

2. Related Research

The FLIP method was firstly introduced into the field of fluid simulation by Brackbill [1,2]. It interpolates the grid velocity change into the particle, almost eliminating all the numerical dissipation. Zhu [4] introduced the FLIP and PIC (Particle in Cell) methods into the computer graphics community in sand simulation, in which the sticky effect of sand flow was reproduced successfully. Batty [5] proposed a fluid-solid coupling method based on variational principles and solved problems on interaction with slim obstacles. Cornelis [6] and Zhu [7] combined FLIP method with Smoothed Particle Hydrodynamics (SPH) method for incompressible fluid simulation with an expectation of getting more details. Boyd [8] and Nan [9] utilized MultiFLIP method in simulating the multiphase flow by applying FLIP method for incompressible two-phase fluid simulation and have successfully simulated the interaction of liquid and air bubbles. With FLIP method, it is necessary to capture the particle-to-particle interface in the grid. However, a large number of particles make no contribution to the interface reconstruction. To overcome this disadvantage, Ferstl [3] proposed a NBFLIP method for fluid modeling by resampling particles within a Narrow Band near interface instead of arranging the particles in the entire grid. The NBFLIP method has a wide application prospect since it can reduce computational complexity without reducing the details of the simulation.

In addition to the above hybrid methods, there are still grid-based Euler method and particle-based Lagrange method. For grid methods, Stam [10] proposed a semi-implicit method to solve the convective term in the NS equation, thus realizing the unconditional stability of the numerical simulation. It has become a very popular method for Euler fluid simulation. Foster [11] pioneered to use the finite difference method in solving three-dimensional NS equations in the field of computer graphics, and achieved relatively rich 3D fluid simulation results.

For the meshless method, Smoothed Particle Hydrodynamic (SPH) and Position Based Fluids (PBF) are the most popularly used. The SPH [12] method, originally proposed by Monaghan, is a meshless method developed over the past 20 years. The basic idea of this method is to describe the continuous fluid (or solid) with the particle of interaction. Each material point is loaded with various physical quantities, including mass, velocity and pressure. The mechanical behavior of the whole system is obtained by solving the kinetic equation of the particle and tracking the motion of each particle. The method is pushed forward and then widely applied to the simulation of foam and spatter effect, free surface and multiphase flow.

The PBF [13,14] method, which is different from the previous mechanical method, is for updating the object directly according to the constraints between the position of the object. Therefore, it will not result in error accumulation from the gradual integration in the traditional method. However, the constraints of the vertex position and the particle in this method belong to the category of geometry, does not belong to the physical simulation in the strict sense.

3. Proposed Method

3.1. NBFLIP Method

The main idea of the NBFLIP method is the same as that of FLIP. specifically, fill particles in the grid and use these particles to treat advection and interact with obstacle boundary. The NSE is solved in projection method, NSE is shown in Equation (1)

$$\begin{cases} \frac{\partial V}{\partial t} + (V \cdot \nabla)V - \nabla\left(\frac{p}{\rho}\right) + \mu\nabla^2 V = f \\ \nabla \cdot V = 0 \end{cases} \tag{1}$$

As shown in Table 1, V is the velocity vector; p is pressure; ρ is fluid density; f is external force; μ is viscosity; The upper one is momentum equation, which includes the advection term, pressure term, diffusion term and external force term from second term; the lower one is continuity equation, which defines the divergence-free velocity field in irrotational fluid simulation.

Table 1. Nomenclature in this paper.

Symbols	Meaning
V	Velocity vector
∇	Nabla Operator
ρ	Fluid density
μ	Fluid viscosity
f	External force
t	Time
p	Pressure
r	Radius
$h, \delta x$	Grid width
x, y, z	Position
i, j	Grid position index

For pressure projection

$$\frac{\partial V}{\partial t} = \nabla \left(\frac{p}{\rho} \right) \tag{2}$$

For Equation (2), take divergence for left the right, and for

$$\nabla \cdot V^{n+1} = 0 \tag{3}$$

In addition, for single phase fluid simulation, the density ρ could be constant , we set $\rho = 1$, yield

$$\nabla(\frac{-V}{\Delta t}) + \nabla^2 p^{n+1} = 0 \tag{4}$$

The projection step is shown in Equation (4). In advection step, the attributes of the particles, such as velocity and pressure need to be interpolated into the grid. Advection step is the main difference between FLIP and NBFLIP method. Next, diffusion and pressure projection are solved on grid, which is the same as in Euler method. Then the velocity and pressure are interpolated back to the particles to update the particle attributes after the pressure solution convergence. The solver for pressure projection step is Preconditioning Conjugate Gradient (PCG) algorithm. PCG is famous in linear system solving.

The flow chart of specific algorithm is shown in Figure 1. The transition from gird to particles are handled by sharp transition as shown in Equation (5). The reader are suggested to refer to reference [3] for more detailed discussion. u_F^* is the final velocity for advection; u_F^p is the velocity from FLIP method outside the narrow band; $u_F^{'}$ is the velocity from after advection in the narrow band.

In the narrow band, the interpolation method has little influence on the simulation result. We can hardly identify the difference from the visual performance. To make the result more intuitive, we choose the Kinetic Energy (KE) dissipation of different interpolation method for explanation. The KE dissipation has slight difference in different interpolation method, but they all rest in the final for both FLIP and NBFLIP method. As we can see that the tendency of the KE dissipation agrees well with the result from reference [3].

$$u_F^* = \begin{cases} u_F^p, & if \quad SD \geq -r \\ u_F', & else \end{cases} \tag{5}$$

Another main difference between NBFLIP method and FLIP method is that the NBFLIP needs to resample the particles in the Narrow Band near the boundary. In this way particles are limited to the Narrow Band only, thus reducing the number of particles and improving computational efficiency. In this paper, SDF [15] is used for distance measurement. Signed Distance (SD) is the distance from a point to a triangular which can be calculated by CPM (Closest Point Method) [16], which will be discussed in the next section in details. Furthermore, self-sampling method is used to calculate the free liquid surface distance. As shown in Figure 2, region [−R, 0] is defined as Narrow Band, [−r, 0] as Combination band and [−R, −h] as Resampling Band. h represents the grid width, in normal conditions R = 3h , r = 2h. To avoid modifying to the free surface, particle resampling is only performed in the Resampling Band area. Only the particles in the Combination Band are interpolated.

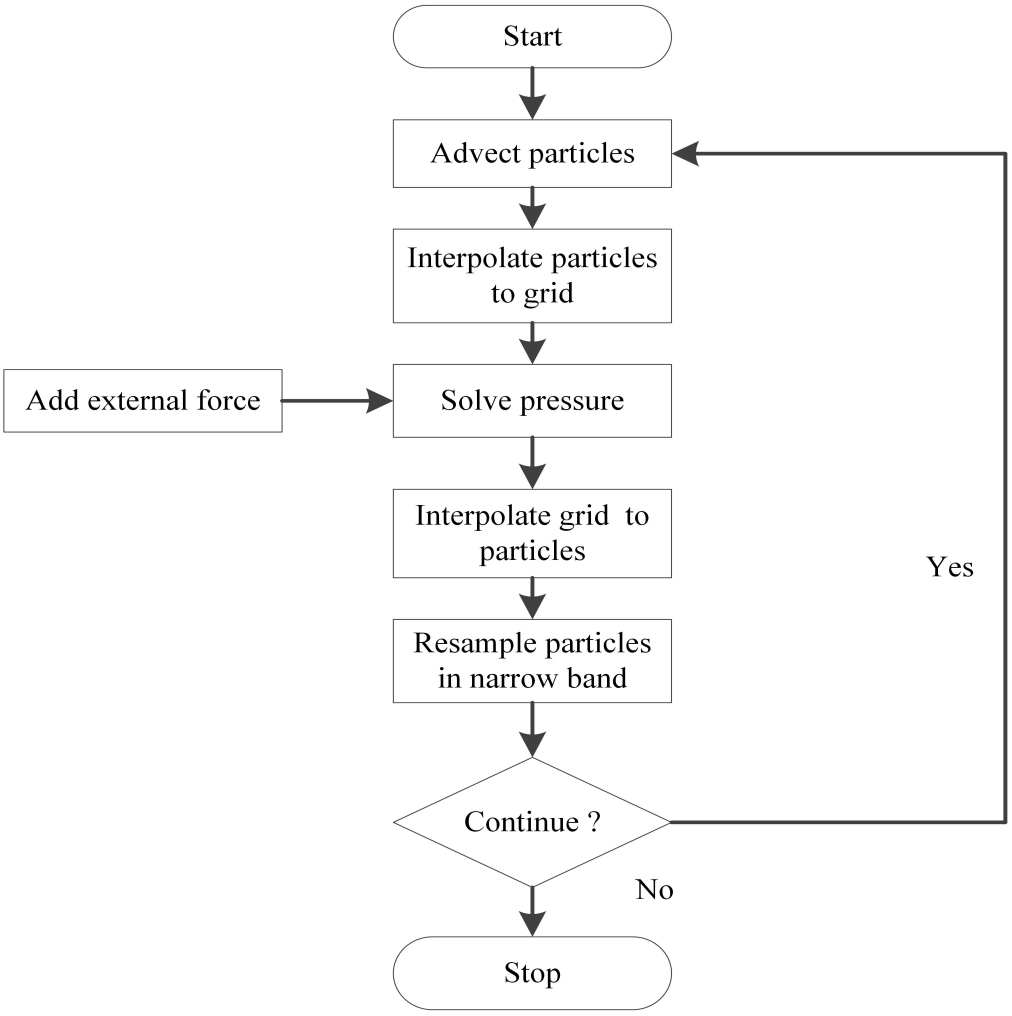


Figure 1. NBFLIP Flow Chart.

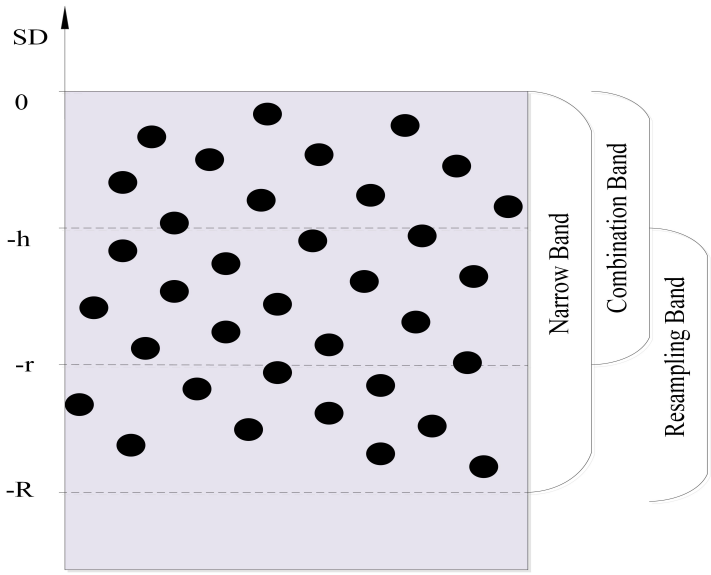


Figure 2. Breakdown of the Narrow Band [3].

3.2. Complex Obstacle Representation Based on SDF

The boundary treatment of complex obstacle is one of the important issues in fluid simulation. The reproduction of realistic fluid interaction with complex obstacle boundaries is still a challenging subject. The first problem to be solved in achieving fluid interaction is to discrete obstacles before the interaction. The existing methods for dealing with obstacle boundary interaction mainly include bounding box (ball), collision detection, LevelSet method and so on. The bounding box (ball) method does not accurately represent the obstacle boundary, artifact will happen near the boundary area; collision detection method requires huge collision test, and the fluid may penetrate through the boundary.

$$\begin{cases} \varphi(x,y,z) < 0, & Inside \quad \Gamma \\ \varphi(x,y,z) = 0, & \text{On} \quad \Gamma \\ \varphi(x,y,z) > 0, & Outside \quad \Gamma \end{cases} \tag{6}$$

In LevelSet method, SDF needs to be obtained before the boundary interface tracking. As can be seen from Equation (6), it is the definition of SDF, φ is the value of SDF. SDF < 0 if it is inside the boundary; SDF $= 0$ if it is on the boundary; SDF > 0 if it is outside the boundary. SDF can be calculated from the definition, the method can be described in Algorithm 1 and Figure 3.

Algorithm 1: SDF Calculation Procedure.

Step1. Initial Smin=Inf. Inf is a big value used to initialize the Smin which will be replaced by the real distance in the next.
Step2. Calculate the distance from grid (i,j) center to triangular S, the distance is set as s1.
Step2.1 Compare Smin and s1, if Smin>=s1, set Smin=s1.
Step2.2 Else if Smin<s1, Smin remains the same.
Step3. Repeat Step2, go through all the grids until finished

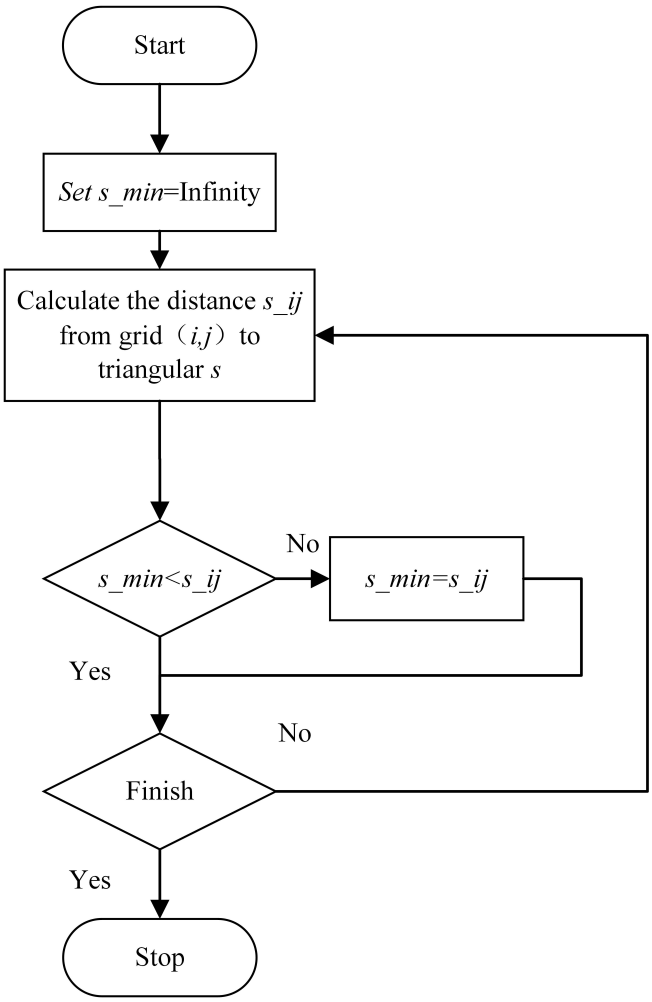


Figure 3. Flow Chart for SDF Calculation.

In our algorithm, we suppose the obstacle is a triangular mesh, as triangular mesh is quite common in computer graphics area and for other meshes it will be almost the same. In Algorithm 1, Smin means the minimum distance from grid (*i*,*j*) to the obstacle mesh triangular. With this method, the distance from the grid to the triangular is essential. The distance could be calculated from the grid position to the triangular if the projection of the grid position is inside the triangular; In this case, the problem equals to how to calculate the distance from a point to a plane, this could be solved easily in solid geometry method. Or else the distance is the minimum distance to the three edges. Once we get the distance from one grid to the triangular, we could calculate the distance to the boundary obstacle in triangular mesh. Furthermore, we could get the whole minimum distance field from all grids to the obstacle triangles. We will be able get the SDF of the obstacle in this way .

Figure 4 is the original bunny mesh; Figure 5 is the SDF calculated from Algorithm 1; Figure 6 is the contour line of SDF on obstacle profile. As can be seen from Figures 4 and 5, the SDF calculated agrees well with the original mesh. The profile in Figure 6 gives the outline of the bunny, in which, the thickened line in the contour is the zero contour line to show the position of the obstacle. Once we get the SDF of the obstacle, we could employ it in fluid interaction in the next step.

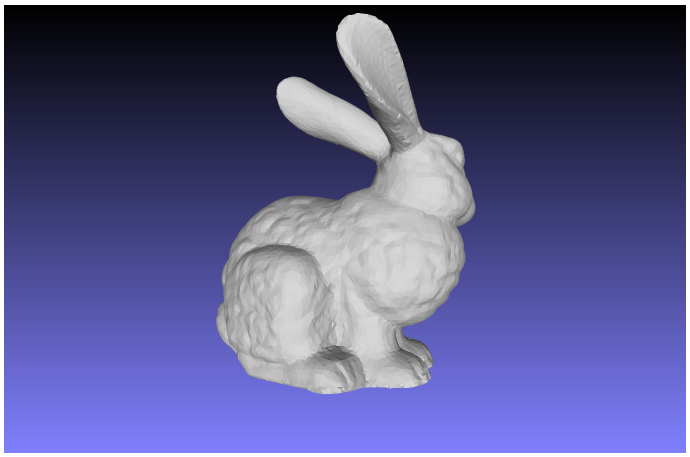


Figure 4. Rendering of original obstacle.

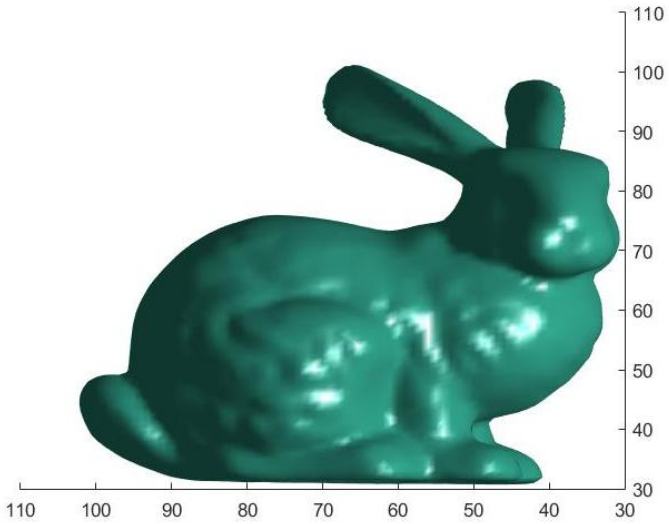


Figure 5. Zero contour surface result of 3D obstacle SDF.

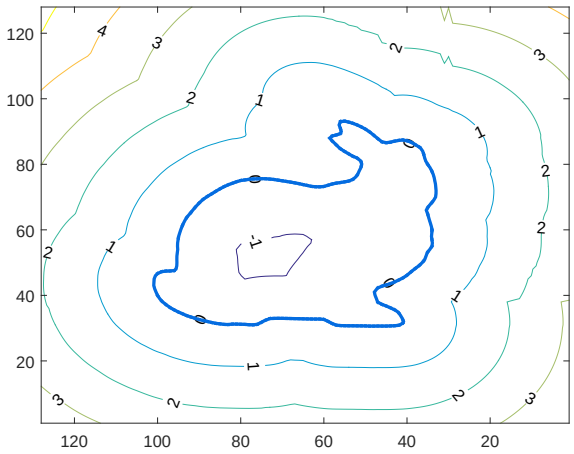


Figure 6. Contour line of SDF on obstacle profile.

3.3. Free Surface and Complex Obstacle Interaction

The obstacle boundary could be represented by SDF prescribed above, but how could the obstacle boundary condition enter the system is not yet discussed. In computer graphics community, fluid simulation is mainly based on NSE, as shown in Equation (1).

The boundary condition could be set during the projection step as shown in Equation (7). Taking 2D as example, if grid (i,j) is fluid, while the neighbouring grid might be solid. In this case, the pressure projection could be discrete as

$$\frac{u^n_{i+1,j} - u^n_{i,j}}{\Delta x \Delta t} + \frac{v^n_{i,j+1} - v^n_{i,j}}{\Delta y \Delta t} = \frac{p^{n+1}_{i+1,j} - 2p^{n+1}_{i,j} + p^{n+1}_{i-1,j}}{\Delta x^2} + \frac{p^{n+1}_{i,j+1} - 2p^{n+1}_{i,j} + p^{n+1}_{i,j-1}}{\Delta y^2} \tag{7}$$

The implementation of the above discretion in case of solid boundary could be described in Algorithm 2 and Figure 7.

Algorithm 2: Free surface and solid obstacle boundary treatment in projection.

```
scale=1.0/(dt*dx);
if(cell(j,i)! = solid)
rhs=-(U(j,i+1)-U(j,i)+V(j+1,i)-V(j,i))/(dt*dx);
if(cell(j,i-1)==solid)
rhs=rhs-scale*(U(j,i)-Usolid(j,i));
end
if(cell(j,i+1)==solid)
rhs=rhs+scale*(U(j,i+1)-Usolid(j,i+1));
end
if(cell(j-1,i)==solid)
rhs=rhs-scale*(V(j,i)-Vsolid(j,i));
end
if(cell(j+1,i)==solid)
rhs=rhs+scale*(V(j+1,i)-Vsolid(j+1,i));
end
pl=1; pr=1; ptop=1; pbot=1;
if(cell(j,i+1)==solid) or if(cell(j,i+1)==empty)
pr=0;
end
if(cell(j,i-1)==solid) or if(cell(j,i-1)==empty)
pl=0;
end
if(cell(j+1,i)==solid) or if(cell(j+1,i)==empty)
ptop=0;
end
if(cell(j-1,i)==solid)or if(cell(j-1,i)==empty)
pbot=0;
end
```

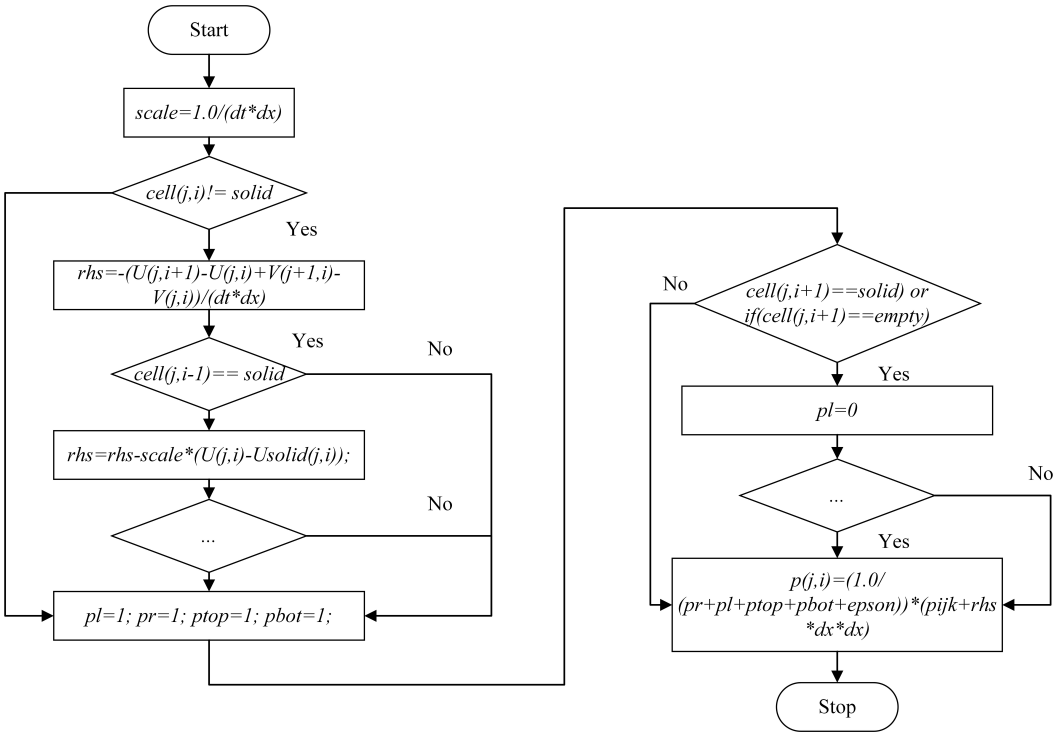


Figure 7. Solid Obstacle Process.

In this algorithm, the cell is the marker cell for the obstacle boundary indicating that the grid is full of fluid, solid or empty; Full means the cell is occupied by fluid. Solid means the cell is occupied by obstacle. Empty means the cell is occupied by air indicating that it is empty and the free surface is the boundary between the fluid and the air. The marker cell could be updated from SDF by determining the sign of the SDF easily. pr, pl, ptop, ptop mean the pressure coefficiency in the right, left, top, bottom grid directions respectively. In case of free surface, we utilize the LevelSet method to track the movement of free surface. The surface tracking algorithm is base on Equation (8)

$$\frac{\partial \varphi}{\partial t} + \nabla \cdot (\varphi V) = 0 \tag{8}$$

V is the current field from NSE; we could start tracking the free surface after initialization of the free surface.

For the empty cells, the pressure is set to zero as shown in Algorithm 2 as if (cell(j,i+1)==empty), which means if the cell is empty, the coefficiency for pressure is set zero. After that, we will keep tracking the free surface.

For solid cells, the pressure is treated the same as empty cell in the first. However, there are still extra treatments for the divergence, In rhs=rhs-scale*(U(j,i)-Usolid(j,i)), Usolid means the horizontal velocity of the solid, which is the correction of divergence considering the solid boundary.

In this way, the solid boundary condition is introduced into the linear system. As we can put the variable for time step $n + 1$ on the left side and the variable for time step n on the right side, we can build a linear system for updating pressure. The coefficient for pressure is shown in Algorithm 2, Hence we can update the pressure for time step $n + 1$ with PCG or other method, the effect of free surface as well as obstacle in fluid could be simulated in fluid.

4. Interactions Simulation of Fluid with Complex Obstacle

4.1. Dam Break Simulation Results

Dam break is a benchmark problem for fluid simulation. The initial setting is as shown in Figure 8. The simulation is carried out in a square container with unit width and unit height. The initial water wall has a height of 0.5 and a width of 0.25. The water wall on the left starts to fall once the experiment launches. The water falls from the left to the right. The front rolls back when it hits the wall boundary on the right as shown in Figures 9d and 10d. The simulation results for 0.2 s, 0.4 s,0.6 s and 0.8 s from the reference [17] are shown in Figure 9, and our simulation results of the same period can be seen in Figure 10. As shown in the two results, our simulation results are in good agreement with the result from reference.

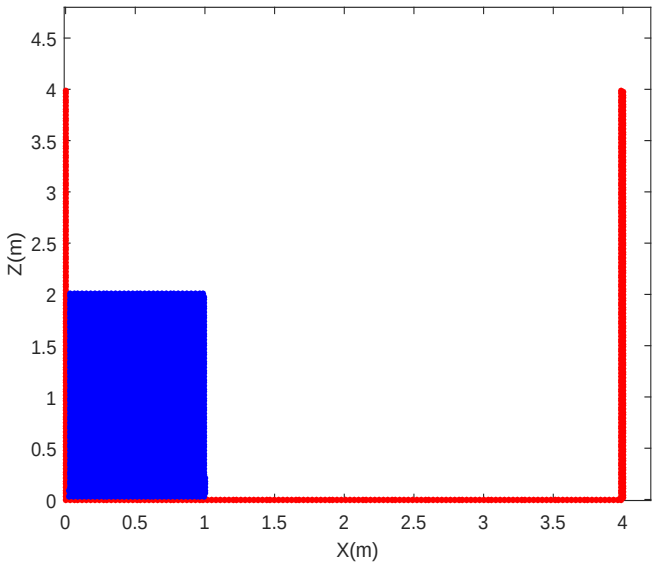
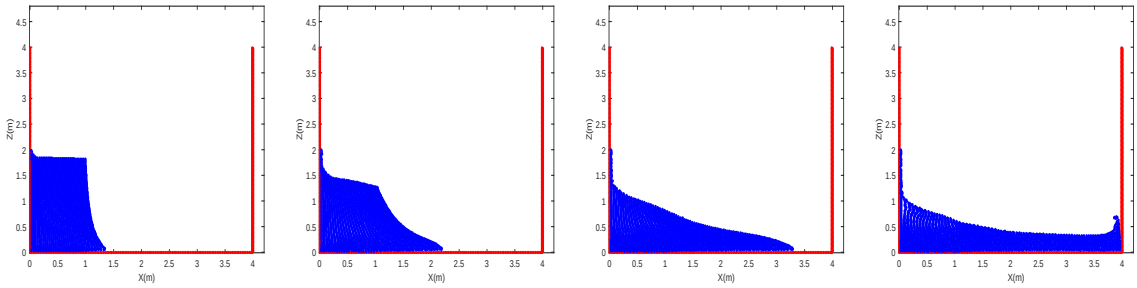
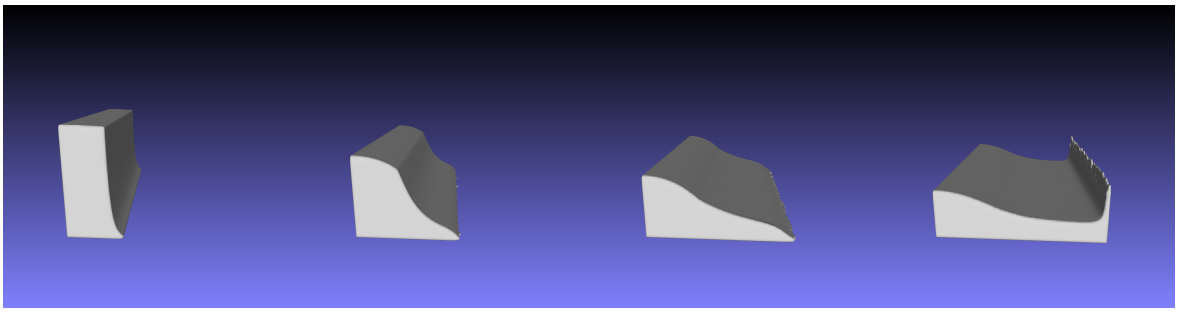


Figure 8. Dam Break Initial Setting.



(**a**) 0.2 s (**b**) 0.4 s (**c**) 0.6 s (**d**) 0.8 s

Figure 9. Dam Break Results from Reference [17].



(**a**) 0.2 s (**b**) 0.4 s (**c**) 0.6 s (**d**) 0.8 s

Figure 10. Our Simulation Result.

4.2. Free Surface Simulation Results

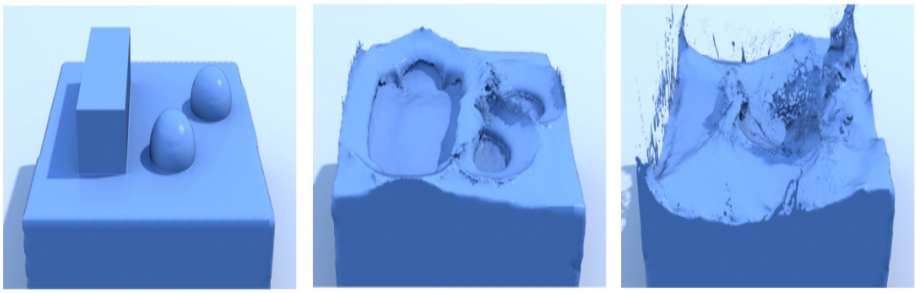
To further verify the simulation results of NBFLIP method, the paper chooses a test case used in the reference [18] for comparison. Fluid surface is reconstructed using Fast Marching Method (FMM) [19].

In Figure 11,we present the simulation result from reference [18]. Figure 12 is our simulation result. It can be seen that our simulation results are very close to the simulation results from reference. With the dropping of the liquid sphere and liquid rectangle, the liquid collided with the fluid in the bottom. In the middle picture, three holes were created when the drop hit the bottom surface. The holes soon were filled with fluid after that, and the splash flied high into the air. The middle images in Figures 11 and 12 show the same result. In particularly, the free surface splash is consistent with the reference results.

While regarding details of the droplet splash, the results with our proposed method are not as good as that in the literature being limited by grid resolution.

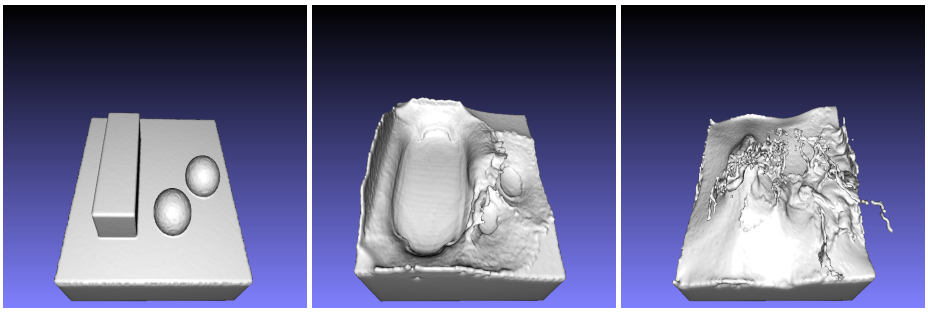
In case of more complex fluid interactions, for example, the liquid bunny falling into the fluid as shown in Figure 13, the bottom liquid has a height of 0.33. This figure shows the result of 0.2 s, 0.4 s, 0.6 s and 0.8 s. This result has certain similarities to that of Figure 12. The difference is that the liquid sphere and rectangle are replaced by more complex liquid bunny. Once the liquid hits the surface, the shape of water crown is likely to be created in the simulation. This is a very common phenomenon such as such as the dropping water on the lake surface.

However, the comparison results show that the simulation results with the method proposed in this paper are very close to the results of the literature, thereby proving the validity of this new method.



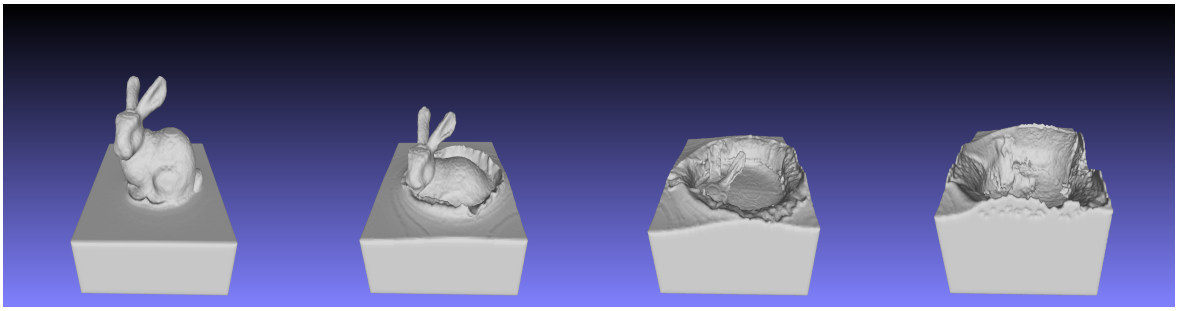
(**a**) 0.0 s (**b**) 0.4 s (**c**) 0.8 s

Figure 11. Simulation Result from Reference [18].



(**a**) 0.0 s (**b**) 0.4 s (**c**) 0.8 s

Figure 12. Our Simulation Result.



(**a**) 0.2 s (**b**) 0.4 s (**c**) 0.6 s (**d**) 0.8 s

Figure 13. Bunny Drop on Surface.

4.3. Simulation of Fluid Interactions with Complex Obstacle

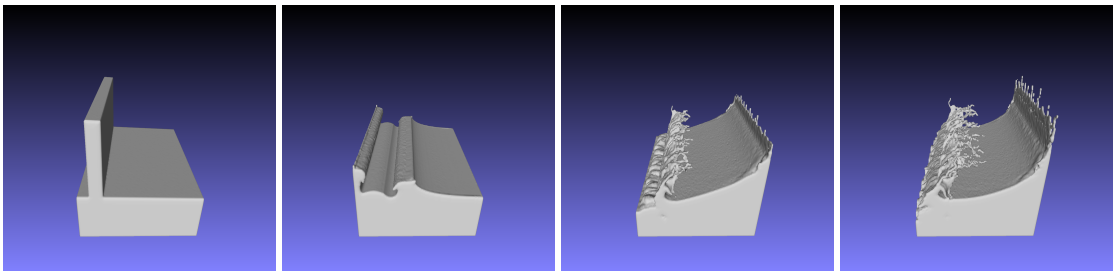
The solid wall boundary is processed by the method proposed in Section 3.3. Figures 14 and 15 show the fluid interaction with and without obstacles.

The graphs from left to right in both Figures 14 and 15 are the initial setting, the others are the first 200th frame, 400th frame, the right picture is the result of 600th frame.

The water wall on the left will fall at the starting of the simulation. The falling down of the water wall on the left caused the water rolls on the left and right as shown in Figure 14b. In addition, the water front on the right continued to propagate from left to the right. The rolls on the left rebound from the left boundary and the front on the right kept climbing the right boundary as shown in Figure 14c. Then the rolls on the left and right splashed into the air.

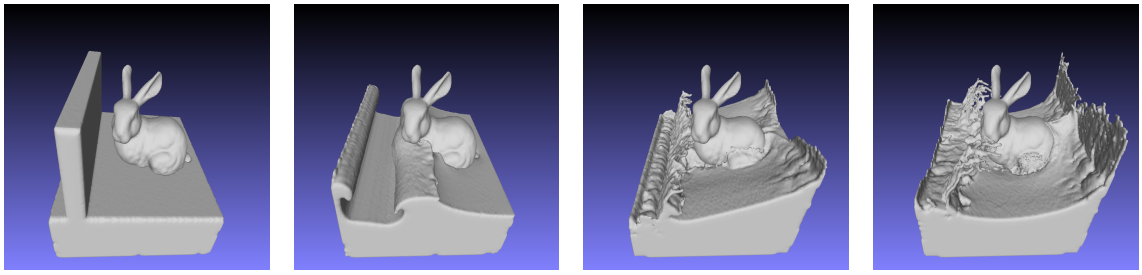
While as Figure 15 indicates, the wall of falling water will cause water splash and interact with the obstacle bunny on the right, as seen in the middle on the left, the water was pushed aside where the water wall falls. The water rolled back on the left boundary while the water hit the bunny on the right. After that, the water on the left boundary rolled back and fell down again.

Figure 16 on the left and right are the close-up of the 200th frame with and without obstacle, which is the same as the one on the left middle in Figure 15b. We can see that the method effectively deals with free surface and the complex solid boundary situation. As a result the splash flies into the sky and the solid obstacle is not penetrated, so the free surface tracking and the interaction between the solid and fluid are successfully demonstrated.



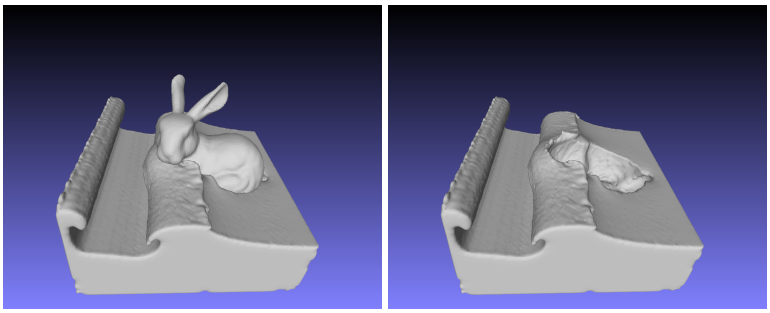
(**a**) 0.0 s (**b**) 0.4 s (**c**) 0.8 s (**d**) 1.2 s

Figure 14. Simulation without Obstacle.



(**a**) 0.0 s (**b**) 0.4 s (**c**) 0.8 s (**d**) 1.2 s

Figure 15. Fluid Simulation with Obstacle.

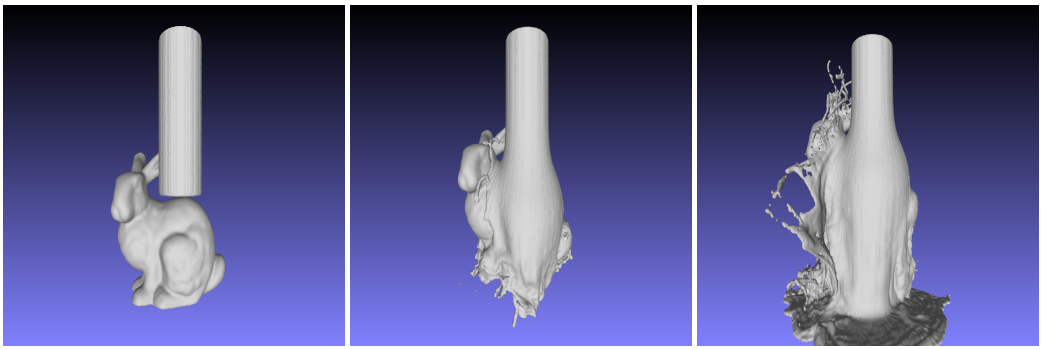


(**a**) Rendering with Obstacle (**b**) Rendering without Obstacle

Figure 16. Rendering with and without Obstacle.

4.4. Interaction with Moving Obstacle

As for the fluid and moving obstacle interaction, the handling method has been shown in Algorithm 2. Actually, this algorithm describes the method for processing static and moving obstacles. In the algorithm Usolid and Vsolid stand for the obstacle velocity in X and Y directions. Obviously, Usolid and Vsolid will be zero for static obstacle. This is a unit method for dealing with free surface, static and moving obstacle. As shown in Figure 17, it shows a bunny moving upward and interact with fluid flowing downward in frame 10, 30 and 50. The time step is 0.02 s. As shown in frame 10, the bunny starts contact with the fluid. When Bunny passes through a fluid column, the fluid flows around the bunny and almost completely envelops the bunny. In the water column, the bunny shape of the fluid is presented. At the same time, as Bunny moves upwards, the outline of bunny in the water column also moves upwards. All these phenomena reflect the effect of interaction between moving obstacle and fluid. It also explains the reliability of moving obstacle and fluid interaction method proposed in this paper.



(**a**) 0.2 s (**b**) 0.6 s (**c**) 1.0 s

Figure 17. Fluid Interaction with Moving Obstacle.

5. Result Analysis

5.1. Particle Number Comparison

In this set of experiments, the system runs 500 frames for each experiment. With NBFLIP method, as can be seen from Figure 18, the total energy and total number of particles are 2.6692×10^7 and 51,872,474, respectively, so the average energy of particles is 0.51. With FLIP method, the total energy and total number of particles are 1.1016×10^8 and 376,245,524, respectively, resulting in the average energy of particles of 0.29. Besides, the total particle number with FLIP method is 7.25 times higher of that with NBFLIP method. In other words, the particle number with NBFLIP method is reduced by 86.2% compared with that with FLIP method. Meanwhile, it can be seen from Figure 19 that the NBFLIP method dissipates less than the FLIP method, which is consistent with the results of Ferstl [3] and Boyd [8]. In other words, NBFLIP method decreases energy dissipation.

Unlike the FLIP method, NBFLIP method limits the particles near the boundary, so as to reduce the number of sampled particles and improve calculation efficiency. It can be seen from Figures 20 and 21 that the left is the particle distribution in the NBFLIP method. It shows that particles are distributed near the boundary of the free surface and the obstacle boundary, which reflects the idea of reducing the number of particles without reducing the details.

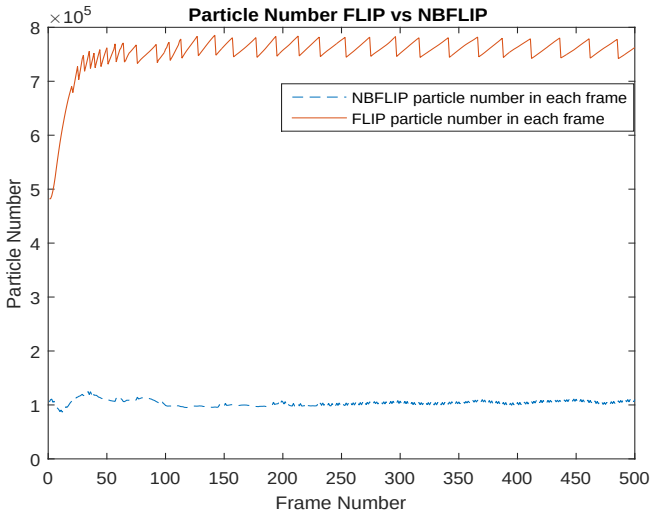


Figure 18. Particle Number Comparison.

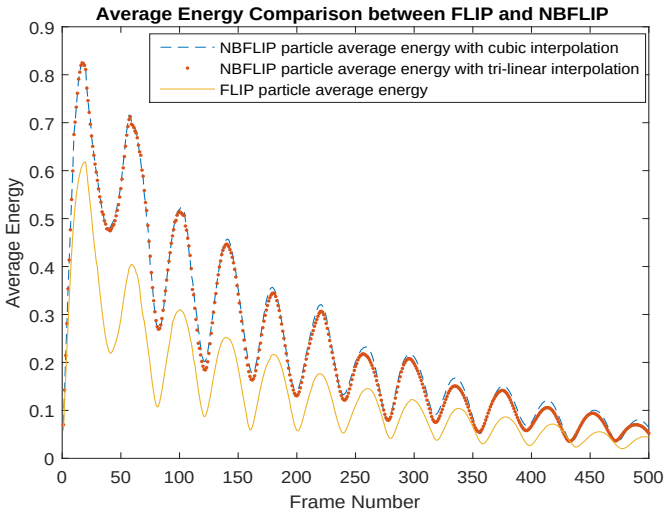


Figure 19. Energy Comparison.

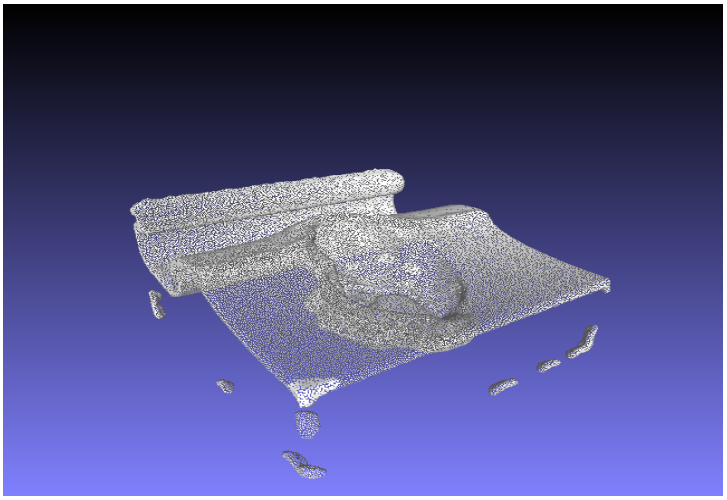


Figure 20. Particles in NBFLIP.

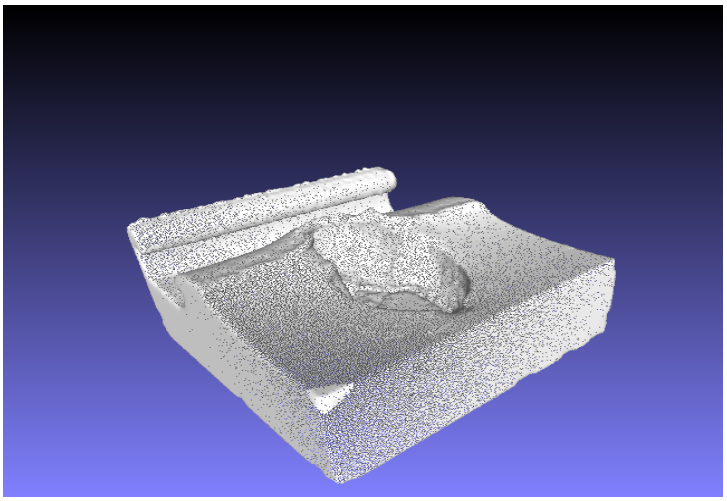


Figure 21. Particles in FLIP.

5.2. Runtime Comparison

In this experiment, we take the first 250 frames result for the running time comparison. During the start-up phase (the first 5 frames) of NBFLIP, the running time per frame is longer than that of FLIP. After that, the running time per frame is significantly less than that of FLIP. It is analyzed that the whole scene needs to be entirely initialized, and the running time is gradually stabilized after the construction of Narrow Band, thus archiving a higher operational efficiency. As shown in Figure 22, particles need to be added into the grid, so the overall running time of each frame is high at the beginning with NBFLIP method, then the number of particles almost remains stable after that.

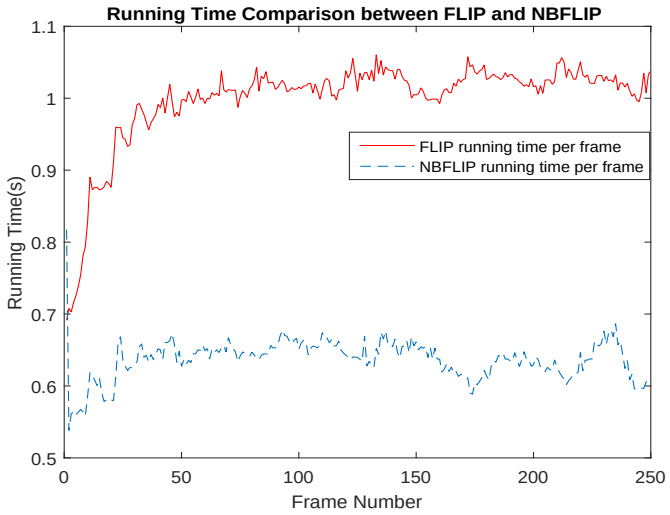


Figure 22. Running Time Comparison

As shown in Figure 22, the total running time is 159 s, the average running time is 0.638 s per frame with NBFLIP method; the total running time for FLIP method is 249.5 s and the average running time per frame is 0.998 s. Compared with the FLIP method, the average running time per frame is decreased by 36.1% with NBFLIP method. This demonstrates the advantages of NBFLIP method in reducing particle number and running time per frame, which means that NBFLIP has better efficiency in free surface tracking and interaction with boundary obstacle.

6. Conclusions and Future Work

Based on the NBFLIP method, this paper proposed a free surface tracking and complex obstacle boundary interaction processing method using SDF method. The main difference between FLIP method and NBFLIP method is that the NBFLIP method limits particles in the vicinity of the boundary (Narrow Band), thereby reducing the number of particles to improve the computational efficiency. Compared with the traditional FLIP method, it is found that the average number of particles per frame is reduced by 86.2% and the average running time per frame is reduced by 36.1%. The energy dissipation is smaller in NBFLIP method, fully verifying the superiority of NBFLIP method in complex fluid simulation. However, this paper only deals with the solid boundary obstacle and does not consider the deformable elastic obstacle boundary. In the future, the simulation of elastic deformation obstacle based on NBFLIP method will be considered in order to further enrich fluid simulation performance.

However, there are still limitations for the NBFLIP method. For most of the cases, the NBFLIP method has better performances than the FLIP method. However, for some situations with thin structures (smaller than 3r), the NBFLIP method is slower than the FLIP method. The reason for this could be that almost all of the particles are within the Narrow Band, hence very few particles are deleted in the resampling process. As a result, the resampling slows down the final efficiency.

Author Contributions: All authors contributed equally to the paper.

Funding: The author would like to acknowledge the support from the National High Technology Research and Development Program of China ("863"Program) [No. 2015AA016404], Marine public welfare industry research [No. 201505017-4] , the Fundamental Research Funds for the Central Universities [No. 3132016310] and Manoeuvre Simulation of Inland River Vessel of Yunnan (YunJiaoKe2013(A)01, 851333).

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

NSE	Navier-Stokes Equation
FLIP	Fluid Implicit Particle
NBFLIP	Narrow Band FLIP
KE	Kinetic Energy
SDF	Signed Distance Function
SD	Signed Distance
CPM	Closest Point Method

References

1. Brackbill, J.U.; Kothe, D.B.; Ruppel, H.M. Flip (fluid-implicit-particle): A low-dissipation, particle-in-cell method for fluid flow. In Proceedings of the Workshop on Particle Methods in Fluid Dynamics and Plasma Physics, Los Alamos, NM, USA, 13 April 1987; pp. 25–38.
2. Brackbill, J.U.; Kothe, D.B.; Ruppel, H.M. Flip: A low-dissipation, particle-in-cell method for fluid flow. *Comput. Phys. Commun.* **1988**, *48*, 25–38. [CrossRef]
3. Ferstl, F.; Ando, R.; Wojtan, C.; Westermann, R.; Thuerey, N. Narrow band flip for liquid simulations. *Comput. Graphics Forum* **2016**, *35*, 225–232. [CrossRef]
4. Zhu, Y.; Bridson, R. Animating sand as a fluid. *ACM Trans. Graph.* **2005**, *24*, 965–972. [CrossRef]
5. Batty, C.; Bertails, F.; Bridson, R. A fast variational framework for accurate solid-fluid coupling. *ACM Trans. Graph.* **2007**, *26*, 100. [CrossRef]
6. Cornelis, J.; Ihmsen, M.; Peer, A.; Teschner, M. Iisph-flip for incompressible fluids. *Comput. Graphics Forum* **2014**, *33*, 255–262. [CrossRef]
7. Zhu, B.; Yang, X.; Fan, Y. Creating and preserving vortical details in sph fluid. *Comput. Graphics Forum* **2010**, *29*, 2207–2214. [CrossRef]
8. Boyd, L.; Bridson, R. Multiflip for energetic two-phase fluid simulation. *ACM Trans. Graph.* **2012**, *31*, 1–12. [CrossRef]
9. Nan, C. Research on Enhancing Details of Fluid Simulation. Master's Thesis, Changan University, Xi'an, China, 2015; pp. 1–67. (In Chinese)
10. Stam, J. Stable fluids. In Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques, Los Angeles, CA, USA, 8–13 August 1999; pp. 121–128.
11. Foster, N.; Metaxas, D. Controlling fluid animation. In Proceedings of the Conference on Computer Graphics International, Diepenbeek, Belgium, 23–27 June 1997; pp. 178–188.
12. Monaghan, J. Simulating Free Surface Flows with SPH. *J. Comput. Phys.* **1994**, *110*, 399–406. [CrossRef]
13. Charypar, D.; Gross, M. Particle-based fluid simulation for interactive applications. In Proceedings of the ACM Siggraph/Eurographics Symposium on Computer Animation, San Diego, CA, USA, 26–27 July 2003; Eurographics Association: Geneva, Switzerland, 2003; pp. 154–159.
14. Müller, M.; Solenthaler, B.; Keiser, R.; Gross, M. Particle-based fluid-fluid interaction. In Proceedings of the ACM Siggraph/Eurographics Symposium on Computer Animation, Los Angeles, CA, USA, 29–31 July 2005; pp. 237–244.
15. Calakli, F.; Taubin, G. SSD: Smooth signed distance surface reconstruction. *Comput. Graphics Forum* **2011**, *30*, 1993–2002. [CrossRef]
16. Macdonald, C.B.; Ruuth, S.J. Levelset equations on surfaces via the closest point method. *J. Sci. Comput.* **2008**, *35*, 219–240. [CrossRef]

17. Crespo, A.J.C.; Dominguez, J.M.; Gesteira, M.; Barreiro, A.; Rogers, B.D. 2011. User Guide for the DualSPHysics Code v1.0. Available online: http://www.dual.sphysics.org (accessed on 1 May 2018).
18. Ando, R.; Wojtan, C. Highly adaptive liquid simulations on tetrahedral meshes. *ACM Trans. Graph.* **2013**, *32*, 1–10. [CrossRef]
19. Sethian, J.A. Levelset methods and fast marching methods. *J. Comput. Inf. Technol.* **1999**, *11*, 400–410.