Hindawi
Scientific Programming
Volume 2017, Article ID 8502691, 11 pages
https://doi.org/10.1155/2017/8502691

Research Article

Rigid Body Sampling and Individual Time Stepping for Rigid-Fluid Coupling of Fluid Simulation

Xiaokun Wang, Xiaojuan Ban, Yalan Zhang, and Xu Liu

University of Science and Technology Beijing, Beijing 100083, China

Correspondence should be addressed to Xiaojuan Ban; banxj@ustb.edu.cn

Received 4 August 2016; Accepted 14 February 2017; Published 2 March 2017

Academic Editor: Wenbing Zhao

In this paper, we propose an efficient and simple rigid-fluid coupling scheme with scientific programming algorithms for particle-based fluid simulation and three-dimensional visualization. Our approach samples the surface of rigid bodies with boundary particles that interact with fluids. It contains two procedures, that is, surface sampling and sampling relaxation, which insures uniform distribution of particles with less iterations. Furthermore, we present a rigid-fluid coupling scheme integrating individual time stepping to rigid-fluid coupling, which gains an obvious speedup compared to previous method. The experimental results demonstrate the effectiveness of our approach.

1. Introduction

Physically based fluid simulation is a popular issue in computer graphics and virtual reality while having a huge research and application demand in three-dimensional visualization and human- computer interactions. More realistic effects and higher simulation efficiency are the main goals; thus, reasonable, efficient, and scientific programming algorithms are needed to design and implement the animation. Two major schemes are employed for animating fluids: the grid-based Eulerian approach and particle-based Lagrangian approach. Eulerian method is particularly suited to simulate large volumes fluid, while being restricted by time step and computing time for small scale features. In contrast, Lagrangian method is suitable for capturing small scale effects such as spindrift and droplet. Among various particle-based approaches, Smoothed Particle Hydrodynamics (SPH) is the most popular method for simulating fluid due to computational simplicity and efficiency.

In reality, rigid-fluid interaction widely exists in many scenarios. As a result, the interesting fluid behaviors emerge once rigid objects are added to fluid simulation. While interaction between particle-based fluids and rigid objects seems to be straightforward, there are still several issues not well resolved. For one thing, rigid bodies must be sampled to particles in order to interact with particle-based fluids, but only a few rigid boundary sampling methods can be directly employed in rigid-fluid coupling simulation. For another thing, the computational expenses of rigid-fluid coupling are considerable. To deal with the increasing demands for more detailed fluids and high efficiency, we present rigid sampling and individual time stepping for rigid-fluid coupling and design a practical and easy rigid-fluid animation simulation scheme with our scientific programming algorithms.

2. Related Work

Desbrun and Gascuel introduced SPH to computer graphics for simulating deformable objects [1]. SPH became popular in computer graphics for various fluid phenomena. Monaghan addressed simulating free surface flows with SPH [2] that serves as a basis for SPH fluid simulation. Muller et al. [3] proposed using gas state equation with surface tension and viscosity forces for fluid simulation, which also bring compressibility issue. Becker and Teschner [4] proposed WCSPH employing Tait equation to reduce compressibility. It significantly increased realistic effects but the efficiency is limited by time step. As incompressibility expenses computation time, many improved algorithms were addressed to enhance the efficiency. Solenthaler and Pajarola [5] presented PCISPH using a prediction-correction scheme to determine particle's pressure and large time steps which significantly improved efficiency comparing to WCSPH. Another similar method that ensures incompressibility by iterative process is LPSPH [6]. Afterwards, Ihmsen et al. addressed a more efficient approach IISPH [7]. It carefully constructed pressure Poisson equation and solved the linear system using Relaxed Jacobi, which has a great improvement in stability as well as convergence speed and is particularly suitable for large-scale scene. Recently, a promising approach for impressible SPH has been proposed by Bender and Koschier [8]. It combines two pressure solvers which enforce low volume compression and a divergence-free velocity field and permits large time steps that yields a considerable performance gain.

Besides, adaptive method, either spatial resolution or time discretization, is another way to promote efficiency. They allot computing resources to regions with complex flow behavior. Space adaptive methods [9–11] adaptively sample particle and employ less particles to produce similar details. Large particles are divided into small particles if high resolution is needed, and vice versa. However, difficulties exist in reproducing quantity when refining particles, and neighbor searching is usually the bottleneck. As an alternate to adaptive spatial discretization, the time domain can be adaptively sampled as well. Globally adaptive time stepping methods [1, 12, 13] employ a single time step to adjust each step for all particles in consideration of CFL condition. Though each particle has the current smallest time step, it is not the most efficient way. Locally adaptive time stepping methods [9, 14, 15] use different time steps for particles. Desbrun and Cani proposed that each particle evaluates forces depending on its current individual time step [9]. He et al. [16] adopt this idea and implement stable simulation of stiff fluids. It updates position, velocity, and density for active particles and interpolates for inactive particles. In this paper, we integrate it to rigid-fluid coupling to reduce the computational time.

For boundary handling in SPH fluid simulation, distance-based penalty methods were commonly employed [17–19]. Nonetheless, these methods require large penalty forces which limit the time step and make particles stick to the boundary on account of lacking fluid neighbors. Frozen or ghost particles based models are used to solve the problem of sticking particles [20]. In order to avoid penetration, more than one layer of frozen particles were used [21], or the positions of penetrating particles should be corrected [13]. However, handling two-way interaction is troublesome since the elevated density near boundary in one phase affects particles in the other phase. For this reason and for the lack of fluid neighbors, Ghost SPH [22] solved those problem using a narrow layer of ghost particles and Akinci et al. employed boundary particles to correct the calculation of fluid density [23]. Because Ghost SPH is more time consuming, we use Akinci's boundary handling method which is simple and easy to achieve in this paper.

For rigid-fluid coupling, several approaches were presented up to now. In [24], the fluid is represented as rigid spheres and switching impulses with rigid bodies. In [25, 26], the pressure at the boundary is taken into consideration for two-way fluid-rigid coupling. Then, Oh et al. proposed an impulse-based scheme for two-way coupling of SPH fluids with rigid bodies [27]. Becker et al. presented direct forcing for rigid-fluid coupling [28] which employs a prediction-correction scheme to enforce particle positions and velocities to specific values. Akinci et al. presented a momentum-conserving two-way coupling approach based on hydrodynamic forces that use boundary particles to sample the surface of rigid bodies [23]. We present a rigid-fluid coupling scheme by integrating individual time stepping to Akinci's boundary handling method that gains an obvious speedup [29].

Rigid bodies sampling includes particle-based methods and polygonization-based methods. Turk repelled particles on surfaces to get a uniform sample [30] and also simplified a polygonization through reducing the number of polygons [31]. Witkin and Heckbert employed local repulsion to make particles spread uniform [32]. Nehab and Shilane [33] presented algorithm of stratified point sampling. Cook addressed stochastic sampling of Poisson disk distributions with blue noise [34]. Blue sampling has the ability to generate random points and get uniform distribution of sampling points set. Therefore, the following sampling methods always have blue noise characteristics. Corsini et al. sampled triangular meshes with blue noise properties [35]. Dunbar and Humphreys [36] modified Poisson disk sample using a spatial data structure. Bridson [37] simplified Dunbar's approach with rejection sampling and extending it to higher dimensions. Then, Schechter [22] modified Bridson's approach and employed it to Ghost SPH. Inspired by Schechter's approach, we address sampling method improved by SPH equation that is more efficient and easy to implement.

3. Particle-Based Fluid Simulation Framework

In particle-based fluid simulation, the forces acting on particles are derived from the Navier-Stokes equations. The conservation of mass and momentum are written as

$$\begin{aligned} \frac{d\rho_i}{dt} &= -\rho_i \nabla \cdot \mathbf{v}_i, \\ \rho_i \frac{D\mathbf{v}_i}{Dt} &= -\nabla p_i + \rho_i g + \mu \nabla^2 \mathbf{v}_i, \end{aligned} \tag{1}$$

where $\mathbf{v}_i$ is the velocity, ρ_i is the density, p_i is the pressure, μ is the viscosity coefficient, and g is the external force field.

SPH works by obtaining approximate numerical solutions of fluid dynamics equations by expressing fluids with particles. In SPH, the representation of a field variable A at location $\mathbf{x}_i$ is defined as

$$\langle A(\mathbf{x}_i) \rangle = \sum_j m_j \frac{A_j}{\rho_j} W(\mathbf{x}_i - \mathbf{x}_j, h), \tag{2}$$

where m_j and ρ_j represent particle mass and density, respectively, $W(\mathbf{x}_i - \mathbf{x}_j, h)$ is smoothing kernel, and h is smoothing radius.

It can be easily derived from the basic SPH equation by substituting fluid density ρ into (2), that is,

$$\rho_i = \sum_j m_j W(\mathbf{x}_i - \mathbf{x}_j, h). \tag{3}$$

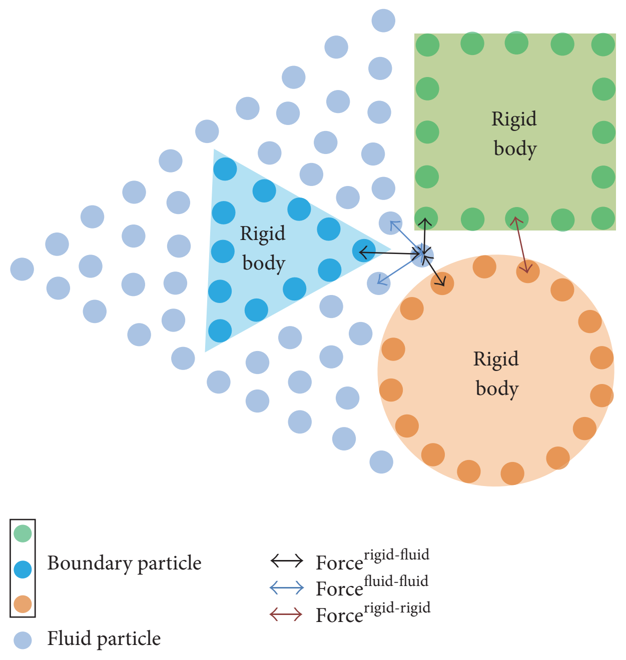


Figure 1: Forces between boundary particles and fluid particles.

Therefore, particles' pressure force $\mathbf{f}_i^P$ and viscous force $\mathbf{f}_i^v$ can be written as

$$\mathbf{f}_i^P = -\sum_j m_j \left(\frac{P_i}{\rho_i^2} + \frac{P_j}{\rho_j^2} \right) \nabla W_{ij}, \qquad \mathbf{f}_i^v = \mu \sum_j m_j \frac{\mathbf{v}_{ji}}{\rho_j} \nabla^2 W_{ij}. \tag{4}$$

In this paper, we use Tait equation [4] to calculate the pressure; that is, $p_i = (\rho_0 c_S^2/\gamma)((\rho_i/\rho_0)^\gamma - 1)$, where $\rho_0 = 1000$ is the rest density of the fluid, $\gamma = 7$ is stiffness parameter, and c_S is velocity of sound. We use the equation in [23] to compute viscous force.

4. Boundary Handling for Particle-Based Fluids

In rigid-fluid interaction, there are three types of forces among particles: the forces between fluid particles and boundary particles Force$^{\text{rigid-fluid}}$, the forces between fluid particles Force$^{\text{fluid-fluid}}$, the forces between boundary particles Force$^{\text{rigid-rigid}}$, which is shown in Figure 1. In our simulation, we sample rigid body to obtain boundary particles which is described in Section 5. Moreover, for the boundary handling way of rigid-fluid interaction, we implement our simulation based on the work of [23]. The following will briefly introduce boundary handling in this section.

Considering influence of boundary particles, density formula of fluid particles in (3) needs to introduce weighted summation influence of boundary particle [23], that is,

$$\rho_{f_i} = \sum_j m_{f_j} W_{ij} + \sum_k m_{b_k} W_{ik}, \tag{5}$$

where f_j and b_k denote fluid particle j and boundary particle k, respectively. The first summation calculates the affection of adjacent fluid particles, while the second summation computes the influence of adjacent boundary particles. This formula can overcome the problem of boundary defects in SPH fluid simulation to some extent.

Due to the use of boundary particle mass in (5), the density of fluid particles is incorrect or unstable when the boundary particle density is set unreasonably or unevenly distributed. Hence, consider the contribution of boundary particles to fluid particle by the volume of boundary particles is

$$\Psi_{b_i}(\rho_0) = \rho_0 V_{b_i} \tag{6}$$

where ρ_0 denotes the remaining density of fluid and V_{b_i} is the estimation value of boundary area volume of corresponding boundary particles. Applying $\Psi_{b_i}(\rho_0)$ to replace the boundary particle mass can guarantee the stability.

Thus, (5) can be rewritten as

$$\rho_{f_i} = \sum_j m_{f_j} W_{ij} + \sum_k \Psi_{b_k}(\rho_{0i}) W_{ik}. \tag{7}$$

The most important interaction between fluid particles and boundary particles is the pressure. The pressure acceleration generated by boundary particles to fluid particles can be computed as

$$\frac{d\mathbf{v}_{f_i}}{dt} = -\frac{k p_{f_i}}{\rho_{f_i}^2} \sum_k \Psi_{b_k}(\rho_{0i}) \nabla W_{ik}, \tag{8}$$

where $p_{f_i} > 0$ takes $k = 2$. When $p_{f_i} < 0$, boundary particles and fluid particles attract each other; then, we can adjust parameter k $(0 \le k \le 2)$ to realize different adsorption effects, and we choose $k = 1$ in our experiment.

To simulate the friction between fluid and container wall or the interaction of rigid body and fluid, we have to compute the friction of boundary particles with fluid particles. The friction is calculated by the artificial viscosity; that is,

$$\frac{d\mathbf{v}_{f_i}}{dt} = -\sum_k \Psi_{b_k}(\rho_{0i}) \Pi_{ik} \nabla W_{ik}, \tag{9}$$

where $\Pi_{ik} = -\nu({\mathbf{v}_{ik}}^T \mathbf{x}_{ik}/({\mathbf{x}_{ik}}^2 + \varepsilon h^2))$, $\nu = 2\alpha h c_s/(\rho_k + \rho_j)$.

On account of (8) and (9) that listed the forces for fluid particles, we can get the forces of boundary particles using Newton's third law. The forces generated by fluid particles to boundary particles are

$$\mathbf{F}_{b_k} = \sum_i \left(\frac{k p_{f_i}}{\rho_{f_i}^2} + \Pi_{ik} \right) m_{f_i} \Psi_{b_k}(\rho_{0i}) \nabla W_{ik}, \tag{10}$$

where i denotes the fluid neighbors of boundary particle k. It is the counteracting force of (8) and (9).

For a rigid body, the total force and torque need to be calculated. This can be separately written as

$$\mathbf{F}_{\text{rigid}} = \sum_k \mathbf{F}_{b_k}, \qquad \tau_{\text{rigid}} = \sum_k \left(\mathbf{x}_k - \mathbf{x}_{\text{rigid}}^{\text{cm}} \right) \times \mathbf{F}_{b_k}, \tag{11}$$

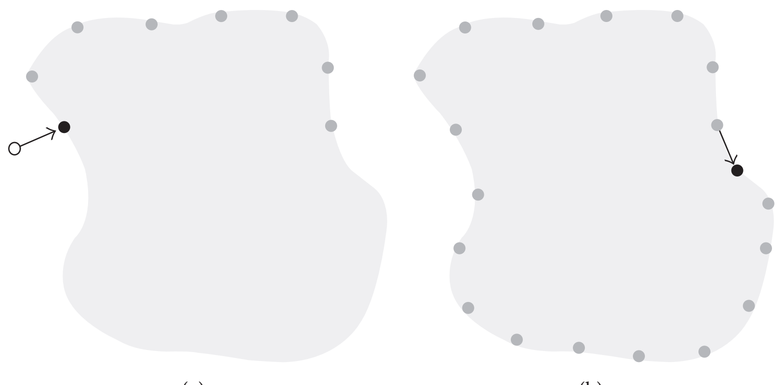


Figure 2: Surface sampling and relaxation. (a) Surface sampling. (b) Surface relaxation. Black points: newly added sample. Gray points: surface sampling points. White points: exterior points before being projected to the surface.

```
Input: Level set $\phi$, radius $r$, count $t$, constant $e$
Output: Sample set $S$
(1) for all grid cells $C$ that $\phi$ changes sign do
(2)     for each $t$ do
(3)         Generate random point $\mathbf{p}$ in $C$
(4)         Project $\mathbf{p}$ to surface of $\phi$
(5)         if $\mathbf{p}$ satisfies Poisson Disk criterion in $S$ then
(6)             $S \leftarrow S \cup \{\mathbf{p}\}$
(7)             Break
(8) if no point was found in $C$ then
(9)             Continue
(10)    while new samples are found do
(11)        Generate random tangential direction $\mathbf{d}$ of surface at $\mathbf{p}$
(12)        $\mathbf{q} \leftarrow \mathbf{p} + \mathbf{d} \cdot e \cdot r$
(13)        Project $\mathbf{q}$ to surface of $\phi$
(14)        if $\mathbf{q}$ satisfies the Poisson Disk criterion in $S$ then
(15)                $S \leftarrow S \cup \{\mathbf{q}\}$
(16)                $\mathbf{p} \leftarrow \mathbf{q}$
```

Algorithm 1: Surface sampling.

where $\mathbf{x}_k$ is the location of boundary particle k and $\mathbf{x}_{\text{rigid}}^{\text{cm}}$ denotes the mass center of a rigid body. The total force and torque will be transmitted to the physics engine to handle the motion of rigid bodies.

5. Rigid Boundary Sampling

Rigid body sampling is the first issue in rigid-fluid coupling which we have to handle. We propose a rigid body sampling algorithm which is an extension of Poisson disk method and sampling method in [22] for rigid-fluid coupling.

For rigid objects sampling, boundary particles are used to sample the surface of rigid bodies that has several merits. For one thing, using particles permits us to get a rigid model that can handle different shapes even with complex geometry structure. For another thing, the use of boundary particles successfully alleviates sticking artifacts and makes sampling uniform.

There are two components in our sampling: surface sampling and surface relaxation. As shown in Figure 2, it first samples the surface of rigid object image, and then it improves initial sampling with surface relaxation. In order to realize the first procedure, it needs fast projection of points to the surface. Hence, level set method is employed to express surface geometry with $\phi > 0$ and $\phi < 0$ denoting exterior and interior of rigid objects, respectively, while $\phi = 0$ denotes surface of rigid objects.

After obtaining the surface geometry of signed distance function, we use surface sampling method proposed in [22] (as shown in Algorithm 1), first, searching seed points on the surface by checking each grid cell intersecting with the surface; the details are as follows: projecting random points from the cell to the surface and stopping when the point satisfies the Poisson disk criterion, which operates k attempts in a cell; when obtaining a seed sample, continuing to sample it and taking a step of size $e \cdot r$ from the previous samples along a random tangential direction $\mathbf{d}$; then projecting to the surface and checking the Poisson disk criterion again. Parameters were chosen as $k = 30$ and $e = 1.085$.

```
Input: sample set $S$, Level set $\phi$, radius $r$, count $t$, constant $f$
Output: relaxed sample set $S$
(1) for each $t$ do
(2)     for each $p_i \in S$ do
(3)         compute density $\rho_i(t)$, average density $\overline{\rho}(t)$
(4)         compute density gradient $\nabla\rho_i(t)$
(5)         $d \leftarrow r \cdot |(\rho_i(t) - \overline{\rho}(t))/\overline{\rho}(t)| \cdot f$
(6)         $p^{\text{new}} \leftarrow p + d \cdot \nabla\rho_i(t)$
(7)           if $p^{\text{new}}$ outside $\phi$ or came from surface sample
(8)                 Project $p^{\text{new}}$ to surface of $\phi$
(9)         if $p^{\text{new}}$ satisfies the Poisson Disk criterion in $S$
(10)        $p \leftarrow p^{\text{new}}$
```

Algorithm 2: Modified surface relaxation.

In order to optimize the position of sample points, reduce noises, and get a uniform distribution set of sampling points, we need to further improve sampling set with a surface relaxation step. Inspired by the relaxation algorithm proposed in [22] and SPH interpolation method, surface relaxation algorithm is presented in Algorithm 2. Unlike employing random testing way in [22], we compel particles to move according to the density gradient. This ensures that the particles move to sparse place, so as to insure uniform distribution of particles.

It starts with the initial particles seed in Algorithm 1 and attempts to reposition each sample through density gradient. Next it computes density $\rho_i(t)$ and density gradient $\nabla\rho_i(t)$ of each surface particles and employs deviation of density $\rho_i(t)$ and average density $\overline{\rho_i(t)}$ as a coefficient to tune distance d. Then, it employs $d \cdot \nabla\rho_i(t)$ to adjust particle locations. Surface sample candidates are projected to the surface once again and are reserved which satisfies the Poisson disk criterion. Parameter t is iterations and f is distance coefficient. According to SPH gradient formula, particle's density gradient can be written as

$$\langle \nabla\rho_i \rangle = \sum_{j=1}^{N} m_j \nabla W \left(\left| \mathbf{x}_i - \mathbf{x}_j \right|, h \right). \tag{12}$$

On the basis of signed distance field ϕ, it is fairly convenient for calculation of steps (7) and (8). If $\phi(p^{\text{new}}) \neq 0$, it means p^{new} is not on the surface. While for projecting particles to surface, we compute signed distance field gradient $\nabla\phi(p^{\text{new}})$. Projection formula is

$$p^{\text{new}} = p + d \cdot \nabla\rho_i(t) - \nabla\phi\left(p^{\text{new}}\right). \tag{13}$$

We have done the experiment in 2 dimensions contrast to the relaxation algorithm in [22]. We randomly generate 100 points in a 0.1×0.1 square shown in Figure 3. The red points represent that it does not meet the conditions of Poisson disk.

Figure 4 shows the relaxation results of Figure 3; the first row is our method and the second row is the method in [22]. The column (a) reveals the distribution of points after relaxation of two algorithms and the red points mean it does not satisfy Poisson disk condition. Each algorithm iterates 100 times, respectively, while column (b) illustrates the number

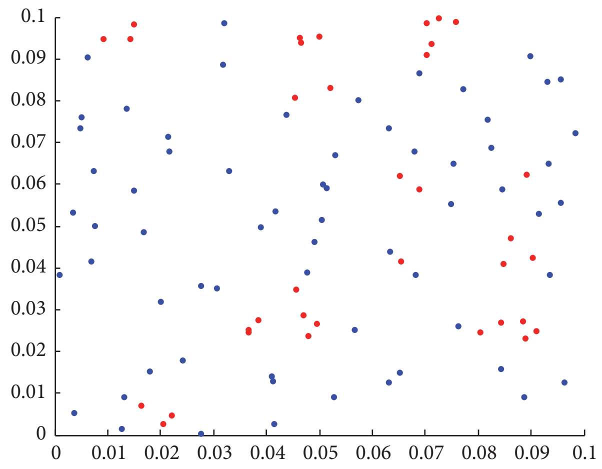


Figure 3: Initial samples in a 0.1×0.1 square.

of points that do not satisfy Poisson disk conditions for each iteration. It is obvious that our method can get a better effect with a slight concussion. Compared to the method in [22], the optimization effect is basically the same after 30 times' iteration using our method while it is the optimal results of fast Poisson disk method. In addition, our method is more efficient. In MATLAB environment, all parameters are the same as mentioned in [22]; our method takes 2.44691 s while relaxation method in [22] costs 56.44153 s for 100 iterations.

6. Individual Time Stepping for Rigid-Fluid Coupling

In this section, we propose a rigid-fluid coupling method employing individual time stepping. As a result, larger time steps can be used comparing to previous methods and the overall computation time is reduced. In particle-based fluids, particles only interact with their neighbors. So permitting particles to have different time steps is more efficient than using a global time step for all particles. The individual time stepping computes time step for each particle and updates time step asynchronously.

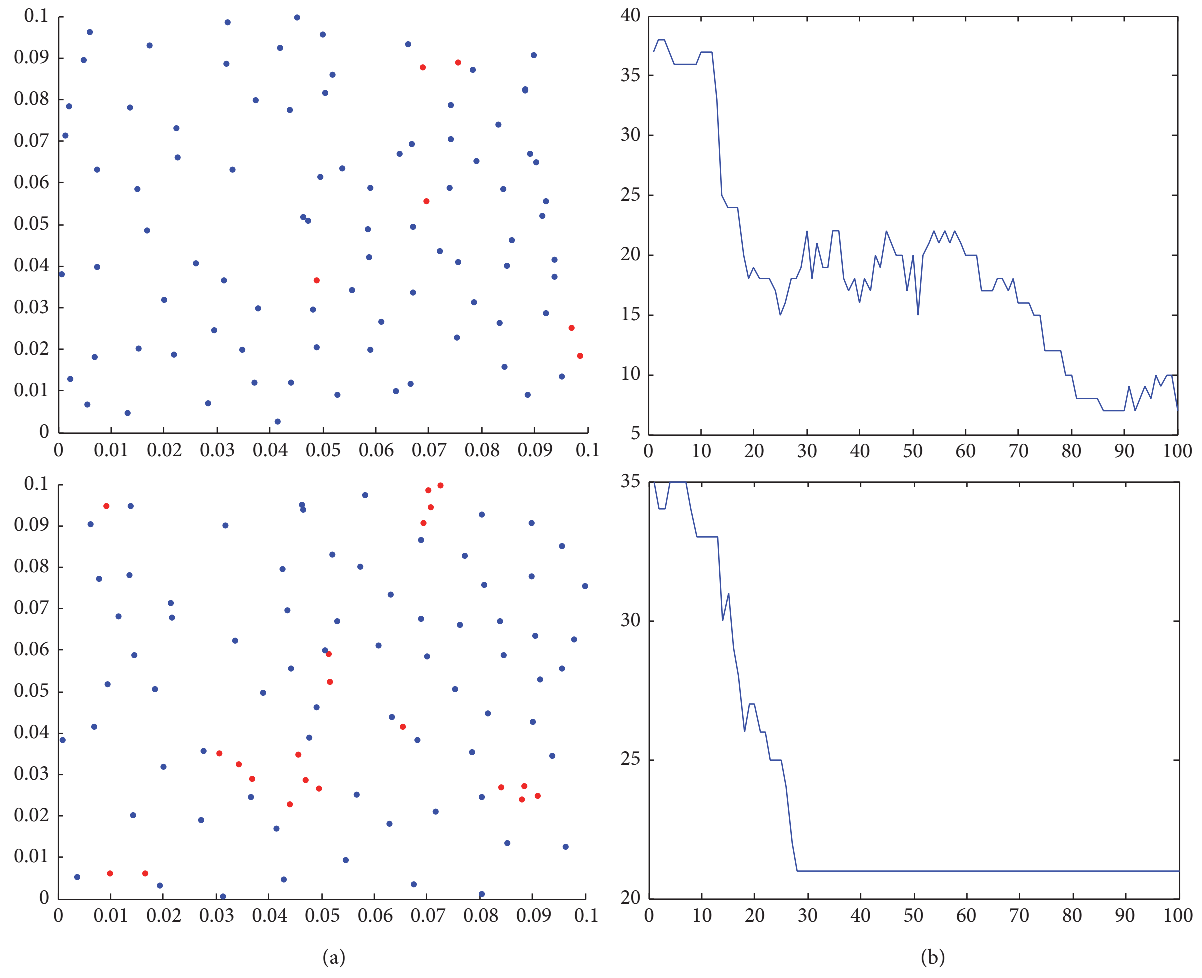


Figure 4: Relaxation results comparison of our method with fast Poisson disk. (a) Relaxation results. (b) The relationship between iteration times and points not conforming to the conditions.

6.1. Time Steps. For particle-based fluids, the time step must satisfies Courant-Friedrich-Levy (CFL) condition for numerical stability, that is,

$$\Delta t_{\mathrm{CFL}} \leq \lambda_v \left(\frac{h}{v_{\max}} \right), \qquad (14)$$

where $v_{\max} = \max_i \|\mathbf{v}_i\|$ is the maximum velocity of particles and coefficient $\lambda_v < 1$. In addition, it also has to consider particles' maximum acceleration. Thus, the time step must also meet the condition

$$\Delta t_f \leq \lambda_f \left(\frac{h}{f_{\max}} \right), \qquad (15)$$

where $f_{\max} = \max_i \|d\mathbf{v}_i/dt\|$ denotes the maximum force per unit mass of particles and $\lambda_f < 1$. In [13], $\lambda_v = 0.4$ and $\lambda_f = 0.25$ are used for PCISPH, while we use $\lambda_v = 0.1$ and $\lambda_f = 0.05$ for WCSPH. Instead of using a constant time step, we adjust time step dynamically as

$$\Delta t \leq \min \left(\Delta t_{\mathrm{CFL}}, \Delta t_f \right). \qquad (16)$$

Thus, the time step for each particle i is

$$\Delta t_i = \min_j \left(\lambda_v \frac{h}{\left\| \mathbf{v}_j \right\|}, \lambda_f \sqrt{\frac{h}{\left\| d\mathbf{v}_j/dt \right\|}} \right), \qquad (17)$$

where j denotes that it iterates all the neighbors.

However, the presented algorithm demands small coefficients for asynchrony, so we choose $\lambda_v = 0.05$ and $\lambda_f = 0.025$.

6.2. Asynchronism. Since each particle has individual time step, we enforce asynchronous time integration for the update. To save computing resources, the system time step is chosen as the minimized individual time step:

$$\Delta t = \min_i \left(\Delta t_i \right), \qquad (18)$$

where Δt_i is computed by (17).

The particle i will be updated if it satisfies the condition

$$t_i^{\mathrm{last}} + \Delta t_i < t, \qquad (19)$$

where t_i^{last} denotes the last update time of particle i and t is the system time. It means that if system time is larger than

```
(1) while animating do
(2)     select active
(3)     for each active fluid particle $i$ do
(4)         find fluid and boundary neighbors
(5)     for each fluid particle $i$ do
(6)         if active then
(7)             compute $\rho_i(t)$, $p_i(t)$
(8)         else
(9)             interpolate $\rho_i(t)$, $p_i(t)$ using $d\rho_i(t_i^{\text{last}})/dt$
(10)    for each active fluid particle $i$ do
(11)        compute $d\mathbf{v}_i(t)/dt$
(12)        compute $d\rho_i(t)/dt$
(13)        compute time step condition $\Delta t_i'$ (Eq. (15))
(14)        $t_i^{\text{last}} = t$
(15)    for each boundary particle $k$ do
(16)        compute forces (Eq. (10))
(17)    for each fluid particle $i$ do
(18)        compute time step $\Delta t_i = \min_j(\Delta t_i')$
(19)    $\Delta t = \min_i(\Delta t_i)$
(20)    for each rigid body do
(21)        compute total forces, torques (Eq. (11))
(22)        pass forces and torques to physics engine
(23)        update rigid body
(24)        update boundary particles of rigid body
(25)    for each fluid particle $i$ do
(26)        $\Delta t' = t + \Delta t - t_i^{\text{last}}$
(27)        $\mathbf{v}_i(t_i^{\text{last}} + \Delta t') = \mathbf{v}_i(t_i^{\text{last}}) + \Delta t'(d\mathbf{v}_i(t_i^{\text{last}})/dt)$
(28)        $\mathbf{x}_i(t_i^{\text{last}} + \Delta t') = \mathbf{x}_i(t_i^{\text{last}}) + \Delta t'\mathbf{v}_i(t_i^{\text{last}} + \Delta t')$
(29)    $t = t + \Delta t$
```

Algorithm 3: Individual time stepping for rigid-fluid coupling.

the individual time step, particle i will be set as active particle and be updated.

Semi-implicit Euler integration is generally used in SPH simulation. To accommodate for asynchronism, the semi-implicit Euler integrations can be expressed as

$$\begin{aligned}\mathbf{v}_i\left(t_i^{\text{last}} + \Delta t'\right) &= \mathbf{v}_i\left(t_i^{\text{last}}\right) + \Delta t'\mathbf{a}_i\left(t_i^{\text{last}}\right),\\ \mathbf{x}_i\left(t_i^{\text{last}} + \Delta t'\right) &= \mathbf{x}_i\left(t_i^{\text{last}}\right) + \Delta t'\mathbf{v}_i\left(t_i^{\text{last}} + \Delta t'\right),\end{aligned} \tag{20}$$

where $\Delta t'$ is an independent integral time step which is different from the global time step Δt. For inactive particles, (20) is equivalent to interpolation, while, for active particles, they are semi-implicit Euler integration equations.

6.3. Algorithm. The individual time stepping for rigid-fluid coupling algorithm is shown in Algorithm 3. In this algorithm, particle i has several extra variables; that is, $d\rho_i(t)/dt$ denotes density derivative, Δt_i is time step, $\Delta t_i'$ is individual condition time step, and t_i^{last} is last updated time. In the algorithm, t is the system time and Δt is system update time step. Particle i is active if $t_i^{\text{last}} + \Delta t_i < t$.

In order to analyse the proposed algorithm, we implement the breaking dam with obstacles experiment. The setting of this experiment is shown in Table 1.

Table 1: The setting of breaking dam with obstacles.

Item	Value
Simulation domain size	12 m × 12 m × 8 m
Fluid particles	153600
Boundary particles	73585
Smoothing kernel function	Cubic splines
Smoothing radius	0.2 m
Fluid particle width	0.1 m

Table 2: Comparison of experimental results of breaking dam with obstacles.

Method	Total comp. time	Avg. Δt (avg. active pct)	Speedup
Constant steps	175 min	0.11 ms	—
Globally adaptive	41 min	0.46 ms	—
Individual stepping	27 min	0.23 ms (31%)	1.5 (6.4)

We compare individual stepping method to adaptive stepping and constant stepping method in breaking dam with obstacles scene. The rending results are shown in Figure 5 and the time statistics are listed in Table 2. From Figure 5, the fluid simulation results are almost not different using three methods, while in Table 2, we can find that our method gains 1.5 and 6.4 times speedup comparing to globally adaptive stepping method and constant stepping method, respectively. In addition, the average active particles percent of individual stepping method is 31%.

7. Implementation and Results

All the experiments in this paper are implemented on Intel 3.50 GHz CPU with 4 cores. The simulation algorithms (Algorithms 1, 2, and 3) and surface reconstruction [38, 39] are actualized with C++ language and multithreading technology. Bullet is used to simulate rigid objects while OpenMP served as parallelization. Images were rendered with Blender.

To implement fluid-rigid coupling animation efficiently and scientifically, we design a fluid-rigid coupling programming simulation scheme shown in Figure 6. We firstly initialize rigid and fluid particles configuration. Then, we do neighbor search using spatial hashing algorithm for each particle. Next, the governing equations of fluid, rigid boundary sampling, and boundary handling are solved as described in the previous sections. Then, total force and torque of rigid are calculated and provided to Bullet. After total force acting on particles is computed, particles are integrated to next time step using asynchronous update scheme introduced in Section 6.

In order to demonstrate the validity of the entire fluid-rigid coupling simulation system, we designed a scene of dropping multiple small squares into water. The setting and statistics are shown in Table 3, and the experimental results

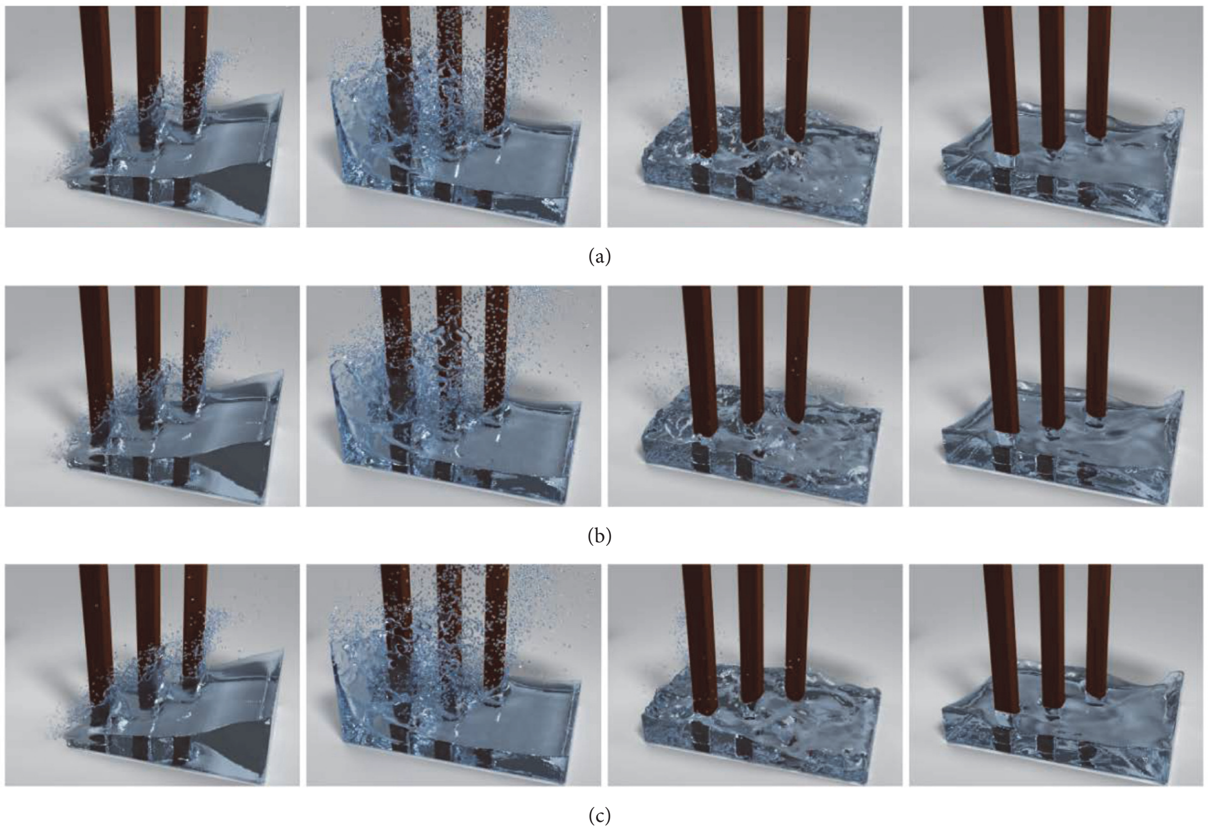


Figure 5: Rendering results of breaking dam with obstacles. (a) Individual stepping method. (b) Globally adaptive stepping method. (c) Constant stepping method.

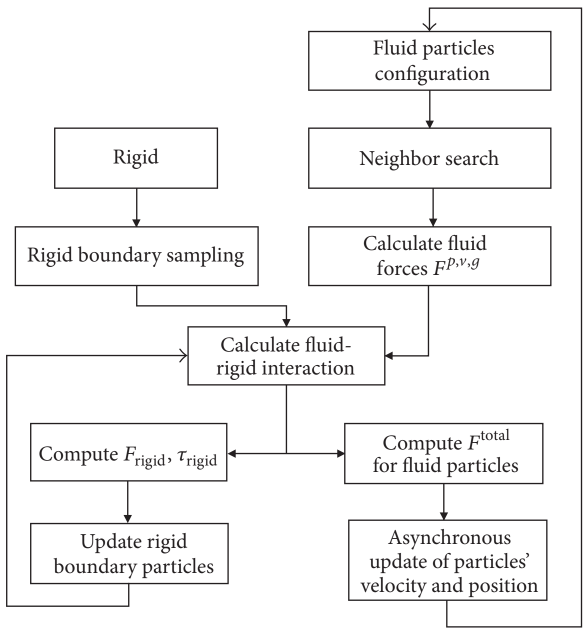

Fluid particles configuration
Rigid
Neighbor search
Rigid boundary sampling
Calculate fluid forces $F^{p,v,g}$
Calculate fluid-rigid interaction
Compute F_{rigid}, τ_{rigid}
Compute F^{total} for fluid particles
Update rigid boundary particles
Asynchronous update of particles' velocity and position

Figure 6: Flowchart of the programming scheme.

are displayed in Figure 7. It can be seen from the diagram that small squares fall into water and splash water while they are rotated and inclined by water. Finally, small squares are force balance and floating on the water.

We realize another fluid-rigid coupling experiment displayed in Figure 8. Figure 8(a) is the results in particle view while Figure 8(b) is the rendering results. The setting and statistics are illustrated in Table 3. In this scenario, the breaking dam of water hits the sculpture which is knocked down and pushed for some distances due to kinetic energy of water. The motions of sculpture are in line with expectations which proved that the simulation and calculation of fluid-rigid coupling system accord with physics laws.

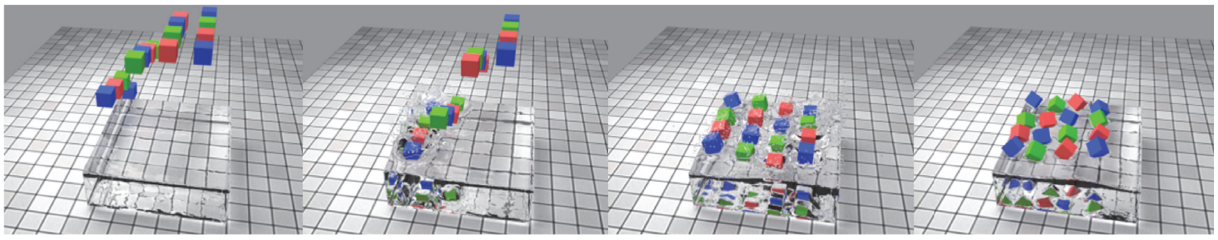
This experiment further proved that our method can implement vivid fluid-rigid coupling animation simulation system with high realistic effects. It can be expected that this animation system can be used in virtual reality domain and special effects in film and game.

8. Conclusion

We proposed an efficient and simple rigid-fluid coupling scheme for particle-based fluid simulation. It samples rigid surface with boundary particles which are used to interact with fluids. It insures uniform distribution of particles which requires less iterations. In addition, we present an efficient rigid-fluid coupling approach combining individual time stepping with rigid-fluid coupling. Neighbors and forces of

Table 3: The setting and statistics of fluid-rigid coupling.

Item	Cubes fall into water	Water shock sculpture
Simulation domain size	12 m × 12 m × 12 m	10 m × 8 m × 5 m
Fluid particles	320 k	873 K
Boundary particles	47 K	156 K
Smoothing kernel function	Cubic splines	Cubic splines
Smoothing radius	0.05	0.1
Artificial viscosity coefficient	0.05	0.05
Surface tension coefficient	0	0
Rigid body mass	65 kg	1100 kg
Rigid body volume	0.125 m^{-3}	18.4 (4 × 2 × 2.3) m^{-3}
Fluid rest density	1000 $kg \cdot m^{-3}$	1000 $kg \cdot m^{-3}$
SPH computing time (1 frame)	0.9008 min	8.46 min
Surface reconstruction time (1 frame)	74.0 s	261 s
Rendering time (1 frame)	13.85 min	8.86 min


(a)

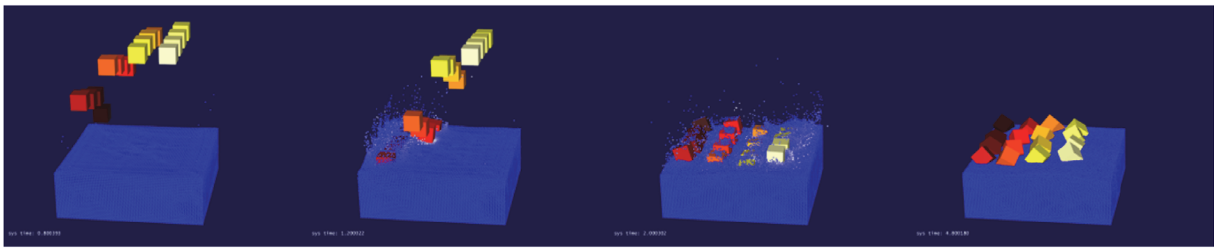
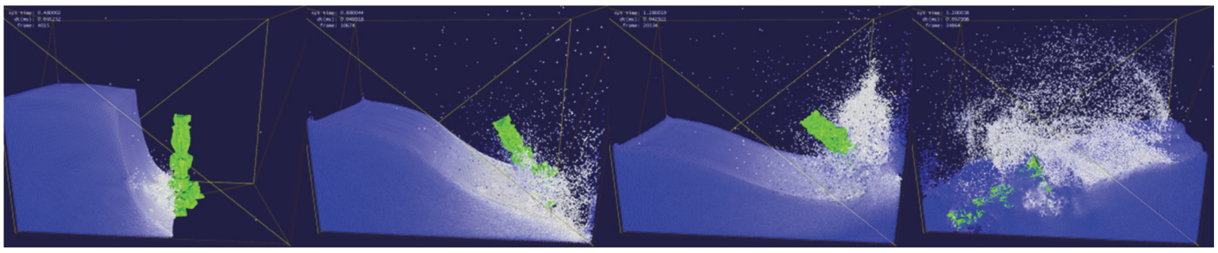

(b)

Figure 7: 16 cubes fall into water. (a) Simulation in particle view; (b) rendering results.


(a)

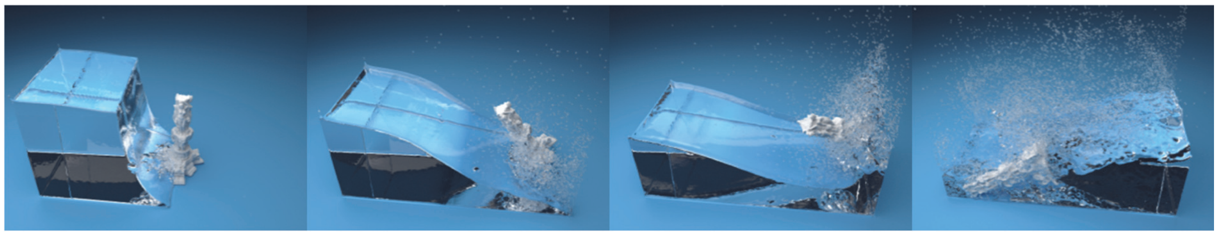

(b)

Figure 8: Water shock sculpture. (a) Simulation in particle view; (b) rendering results.

particles are updated only when needed, while computing resources are allocated to complex regions. It obtains an obvious speedup compared to previous methods. Besides, this scheme was integrated with rigid body coupling simulation with several scenes which has a good sense of visual reality. Overall, our method is efficient to compute while the sampling and coupling algorithm can be applied to other particle-based simulation or relevant approaches. Future work would be extending the proposed method to IISPH [7] or DFSPH [8] as well as large-scale scenarios.

Competing Interests

The authors declare that they have no competing interests.

Acknowledgments

This work was supported by National Natural Science Foundation of China (nos. 61272357, 61300074, and 61572075).

References

[1] M. Desbrun and M. P. Gascuel, “Smoothed particles: a new paradigm for animating highly deformable bodies,” in *Computer Animation and Simulation ’96*, pp. 61–76, Springer Vienna, 1996.

[2] J. J. Monaghan, “Simulating free surface flows with SPH,” *Journal of Computational Physics*, vol. 110, no. 2, pp. 399–406, 1994.

[3] M. Muller, D. Charypar, and M. Gross, “Particle-based fluid simulation for interactive applications,” in *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, pp. 154–159, Eurographics Association, San Diego, Calif, USA, July 2003.

[4] M. Becker and M. Teschner, “Weakly compressible SPH for free surface flows,” in *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, pp. 209–217, San Diego, Calif, USA, August 2007.

[5] B. Solenthaler and R. Pajarola, “Predictive-corrective incompressible SPH,” *ACM Transactions on Graphics*, vol. 28, no. 3, article no. 40, 2009.

[6] X. He, N. Liu, S. Li, H. Wang, and G. Wang, “Local poisson SPH for viscous incompressible fluids,” *Computer Graphics Forum*, vol. 31, no. 6, pp. 1948–1958, 2012.

[7] M. Ihmsen, J. Cornelis, B. Solenthaler, C. Horvath, and M. Teschner, “Implicit incompressible SPH,” *IEEE Transactions on Visualization and Computer Graphics*, vol. 20, no. 3, pp. 426–435, 2014.

[8] J. Bender and D. Koschier, “Divergence-free smoothed particle hydrodynamics,” in *Proceedings of the 14th ACM SIGGRAPH/Eurographics Symposium on Computer Animation (SCA ’15)*, pp. 147–155, Los Angeles, Calif, USA, August 2015.

[9] M. Desbrun and M. P. Cani, *Space-Time Adaptive Simulation of Highly Deformable Substances*, INRIA, 1999.

[10] B. Adams, M. Pauly, R. Keiser, and L. J. Guibas, “Adaptively sampled particle fluids,” *ACM Transactions on Graphics*, vol. 26, no. 3, Article ID 1276437, 2007.

[11] B. Solenthaler and M. Gross, “Two-scale particle simulation,” *ACM Transactions on Graphics*, vol. 30, no. 4, article 81, 2011.

[12] J. J. Monaghan, “Smoothed particle hydrodynamics,” *Annual Review of Astronomy and Astrophysics*, vol. 30, no. 1, pp. 543–574, 1992.

[13] M. Ihmsen, N. Akinci, M. Gissler, and M. Teschner, “Boundary handling and adaptive time-stepping for PCISPH,” in *Proceedings of the Workshop on Virtual Reality Interaction and Physical Simulation*, pp. 79–88, Eurographics Association, Copenhagen, Denmark, November 2010.

[14] P. Goswami and C. Batty, “Regional time stepping for SPH,” in *Eurographics 2014*, pp. 45–48, Eurographics Association, 2014.

[15] P. Goswami and R. Pajarola, “Time adaptive approximate sph,” in *Proceedings of the 8th Workshop on Virtual Reality Interactions and Physical Simulations (VRIPHYS ’11)*, pp. 19–28, December 2011.

[16] L. He, X. Ban, X. Liu, and X. Wang, “Individual time stepping for SPH fluids,” in *Proceedings of the 36th Annual Conference of the European Association for Computer Graphics (Eurographics ’15)*, Eurographics Association, Zurich, Switzerland, May 2015.

[17] M. Müller, S. Schirm, M. Teschner, B. Heidelberger, and M. Gross, “Interaction of fluids with deformable solids,” *Computer Animation and Virtual Worlds*, vol. 15, no. 3-4, pp. 159–171, 2004.

[18] T. Harada, S. Koshizuka, and Y. Kawaguchi, “Smoothed particle hydrodynamics on GPUs,” *Structure*, vol. 4, no. 4, pp. 671–691, 2007.

[19] J. J. Monaghan and J. B. Kajtar, “SPH particle boundary forces for arbitrary boundaries,” *Computer Physics Communications*, vol. 180, no. 10, pp. 1811–1820, 2009.

[20] X. Y. Hu and N. A. Adams, “A multi-phase SPH method for macroscopic and mesoscopic flows,” *Journal of Computational Physics*, vol. 213, no. 2, pp. 844–861, 2006.

[21] R. A. Dalrymple and O. Knio, “SPH modelling of water waves,” in *Proceedings of the 4th Conference on Coastal Dynamics*, pp. 779–787, ASCE, June 2001.

[22] H. Schechter and R. Bridson, “Ghost SPH for animating water,” *ACM Transactions on Graphics*, vol. 31, no. 4, article 61, 2012.

[23] N. Akinci, M. Ihmsen, G. Akinci, B. Solenthaler, and M. Teschner, “Versatile rigid-fluid coupling for incompressible SPH,” *ACM Transactions on Graphics*, vol. 31, no. 4, article 62, 2012.

[24] S. Clavet, P. Beaudoin, and P. Poulin, “Particle based viscoelastic fluid simulation,” in *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, pp. 219–228, ACM, Los Angeles, Calif, USA, July 2005.

[25] G. Oger, M. Doring, B. Alessandrini, and P. Ferrant, “Two-dimensional SPH simulations of wedge water entries,” *Journal of Computational Physics*, vol. 213, no. 2, pp. 803–822, 2006.

[26] R. Keiser, B. Adams, P. Dutre, L. Guibas, and M. Pauly, “Multiresolution particle-based fluids,” *ETH Department of Computer Science*, vol. 31, no. 6, pp. 1797–1809, 2006.

[27] S. Oh, Y. Kim, and B.-S. Roh, “Impulse-based rigid body interaction in SPH,” *Computer Animation and Virtual Worlds*, vol. 20, no. 2-3, pp. 215–224, 2009.

[28] M. Becker, H. Tessendorf, and M. Teschner, “Direct forcing for Lagrangian rigid-fluid coupling,” *IEEE Transactions on Visualization and Computer Graphics*, vol. 15, no. 3, pp. 493–503, 2009.

[29] R. Li and X. Wang, “Individual time-stepping for rigid-fluid coupling of particle based fluids,” in *Proceedings of the International Conference on Cyberworlds (CW ’16)*, pp. 235–238, Chongqing, China, September 2016.

[30] G. Turk, "Generating textures on arbitrary surfaces using reaction-diffusion," *ACM SIGGRAPH Computer Graphics*, vol. 25, no. 4, pp. 289–298, 1991.

[31] G. Turk, "Re-tiling polygonal surfaces," *ACM Transactions on Graphics*, vol. 26, no. 2, pp. 55–64, 1992.

[32] A. P. Witkin and P. S. Heckbert, "Using particles to sample and control implicit surfaces," in *Proceedings of the 21st Annual Conference on Computer Graphics and Interactive Techniques*, pp. 269–277, ACM, Orlando, Fla, USA, July 1994.

[33] D. Nehab and P. Shilane, "Stratified point sampling of 3D models," in *Proceedings of the 1st Eurographics conference on Point-Based Graphics*, pp. 49–56, Eurographics Association, Zurich, Switzerland, June 2004.

[34] R. L. Cook, "Stochastic sampling in computer graphics," *ACM Transactions on Graphics (TOG)*, vol. 5, no. 1, pp. 51–72, 1986.

[35] M. Corsini, P. Cignoni, and R. Scopigno, "Efficient and flexible sampling with blue noise properties of triangular meshes," *IEEE Transactions on Visualization and Computer Graphics*, vol. 18, no. 6, pp. 914–924, 2012.

[36] D. Dunbar and G. Humphreys, "A spatial data structure for fast poisson-disk sample generation," in *Proceedings of the ACM (SIGGRAPH '06)*, pp. 503–508, ACM, Boston, Mass, USA, August 2006.

[37] R. Bridson, "Fast poisson disk sampling in arbitrary dimensions," in *Proceedings of the ACM SIGGRAPH Sketches (SIGGRAPH '07)*, San Diego, Calif, USA, August 2007.

[38] J. Yu and G. Turk, "Reconstructing surfaces of particle-based fluids using anisotropic kernels," *ACM Transactions on Graphics*, vol. 32, no. 1, article no. 5, 2013.

[39] X. Wang, X. Ban, X. Liu, Y. Zhang, and L. Wang, "Efficient extracting surfaces approach employing anisotropic kernels for SPH fluids," *Journal of Visualization*, vol. 19, no. 2, pp. 301–317, 2016.