Hindawi Publishing Corporation
International Journal of Computer Games Technology
Volume 2014, Article ID 806095, 8 pages
http://dx.doi.org/10.1155/2014/806095

Research Article

An Efficient Sleepy Algorithm for Particle-Based Fluids

Xiao Nie,[1,2] Leiting Chen,[1,2] and Tao Xiang[1]

[1] *School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu 611731, China*
[2] *Provincial Key Laboratory of Digital Media, Chengdu 611731, China*

Correspondence should be addressed to Xiao Nie; niexiao2008@gmail.com

Received 30 June 2014; Accepted 15 October 2014; Published 12 November 2014

Academic Editor: Yiyu Cai

We present a novel Smoothed Particle Hydrodynamics (SPH) based algorithm for efficiently simulating compressible and weakly compressible particle fluids. Prior particle-based methods simulate all fluid particles; however, in many cases some particles appearing to be at rest can be safely ignored without notably affecting the fluid flow behavior. To identify these particles, a novel sleepy strategy is introduced. By utilizing this strategy, only a portion of the fluid particles requires computational resources; thus an obvious performance gain can be achieved. In addition, in order to resolve unphysical clumping issue due to tensile instability in SPH based methods, a new artificial repulsive force is provided. We demonstrate that our approach can be easily integrated with existing SPH based methods to improve the efficiency without sacrificing visual quality.

1. Introduction

With an observable development of physically based methods, fluid simulations have popularized themselves in many graphics applications such as feature films and computer games. There are various physics-based fluid simulation techniques available in the context of computer graphics, but here we focus on SPH based methods due to their simplicity and flexibility.

In SPH based methods, the simulation and visual quality are largely influenced by the resolution of the fluid, that is, the number of particles. Nevertheless, the higher the resolution is, the more computing power is required. In general, only a few milliseconds can be used for physics computation in computer games. As a consequence, low resolution fluid simulation is used in games to get acceptable frame rates, for example, in NVIDIA PhysX engine [1] SPH fluid with less than 60 k particles is typically used for real time applications. In SPH based methods, regardless of its usage in off-line simulations or real time 3D games, reducing the total number of fluid particles can lead to a lower computational overhead. To this end, adaptively sampled SPH method [2] has been introduced to allocate more computing resources to specific regions such as the visually important fluid surface. Similarly, our goal is to develop an efficient fluid simulation method based on Lagrangian SPH model, which is able to lower physical simulation times without compromising the resulting visual quality.

Inspired by the sleepy strategy frequently used in rigid body simulations, we apply this idea in the context of SPH based methods. However, unlike adaptive sampling techniques which also concentrate computational resources only on visually interesting regions, we only use a single resolution. We begin by identifying different particle types according to physical or geometrical criteria. Next, we perform only parts of the whole physical simulation pipeline for different particle types. Although we show that our model can reduce computational overhead compared to standard WCSPH approach while still preserving plausible visual results, our algorithm can be also integrated with standard SPH method commonly applied in real time physics engines.

Additionally, we note that traditional SPH based methods usually lead to particle clumping as a result of density underestimation near the fluid surface where negative pressures are generated that causes the particles to cluster, which is known to be tensile instability issue. We avoid this problem

by introducing a positive repulsive force to naturally separate them apart. The experiment results show that our repulsive force can eliminate particle clumping issue that occurred on the fluid surface in previous methods.

The main contributions of this work are as follows:

(1) an efficient SPH based algorithm grounded on a novel sleepy strategy, which can significantly enhance the performance of fluid simulation;

(2) a new artificial repulsive force to resolve tensile instability issues on account of inaccurate density estimation, which can nicely avoid particle clumping near the fluid surface.

In the remainder of this paper, we first briefly review the related work in Section 2. In Section 3, we give an overview of the standard SPH and WCSPH framework. Section 4 continues with a discussion of our sleepy method and a new artificial repulsive force for resolving tensile instability issue in SPH fluids. Implementation details, experimental results, and analysis are given in Section 5. Finally, conclusion of the work is presented in Section 6.

2. Related Work

SPH was first independently developed by Monaghan and Gingold [3] and Lucy [4] to solve astrophysical problems. It was introduced to the graphics community by Stam and Fiume [5] to simulate fire. Muller et al. [6] used this method to interactively simulate compressible fluids. Since then, various fluid effects have been simulated using this technique in computer graphics, such as fluid-deformable interaction [7, 8], fluid-fluid interaction [9, 10], fluid-solid coupling [11, 12], melting [13, 14], surface tension and cohesion [15, 16], weakly compressible fluids [17, 18], and incompressible fluids [19–22].

Most of the work mentioned above adopts a single resolution model both in terms of space and time, allocating computational resources uniformly to each particle which is not always necessary. In order to simulate large-scale fluid scenarios in reasonable time on a desktop computer, several adaptive sampling techniques have been introduced.

One way that employs the concept of adaptive sampling is adaptive particle refinement, which devotes most of the computational resources to visually or physically important fluid areas. Desbrun and Cani [23] used an adaptive scheme in the context of SPH where particle sampling and time sampling are adapted according to a given compromise between accuracy and efficiency. Adams et al. [2] have presented a spatial adaptive particle fluids approach in which particles are subdivided in geometrically complex and visually important areas and merged in regions of low geometric complexity and of low visual interest, and a smooth particle level distribution is also maintained to keep the physical simulation stable. This method can significantly reduce computational cost, making it possible to simulate large-scale and more complex scenarios in acceptable time. However, one drawback of this technique is that the neighbor search is a major performance bottleneck since hierarchical data structures are expensive to construct and maintain. In addition, particle splitting and merging may lead to inaccurate density and force profiles which are important for the accuracy and stability. Zhang et al. [24] extended this method to simulate weakly compressible fluids; they achieve interactive frame rates by executing both the simulation and rendering fully on the GPU. Yan et al. [25] added a physical importance criterion for adaptively splitting or merging particles and also used GPUs to accelerate the simulation. Orthmann and Kolb [26] applied temporal blending scheme to reduce the total number of particles while still being capable of using large time steps. They can simulate up to a million of particles at interactive speed by exploiting the massively parallel computation capabilities of GPUs.

Another way towards adaptive sampling is adaptive time discretization, which uses different time steps for each simulation frame. Desbrun and Cani [23] automatically adjusted the individual time step for each particle with respect to the Courant-Friedrichs-Lewy (CFL) condition; thus instability problem is avoided. Ihmsen et al. [27] have proposed an adaptive time-stepping method for predictive-corrective incompressible SPH (PCISPH), which adaptively evaluates appropriate time steps regardless of the scenario. By utilizing this temporal adaptivity, the overall computational overhead can be largely reduced compared to constant time-stepping techniques.

Recently, Solenthaler and Gross [28] used a multiscale sampling technique that uses two coexisting simulations with different resolution levels to cut down the overall computational overhead, which does not need to split or merge particles, and can stably simulate two simulations of largely different resolutions. This two-scale particle approach can improve the simulation efficiency linearly to the reduction of the total particle count while still achieving similar surface fluid flow behavior. However, the mass conservation is lost because particles need to be dynamically added or removed in the fine resolution region. Horvath and Solenthaler [29] extended this method to simulate high resolution incompressible fluids with multiple levels, which further reduces the computational costs and also conserves the total mass in high resolution area.

3. Background

SPH is a particle-based interpolation method which discretizes continuous field using the properties defined in all particles. As for SPH fluids, this interpolation can be described as

$$A(\mathbf{x}) = \sum_j \frac{m_j}{\rho_j} A_j W\left(\mathbf{x}_{ij}, h\right). \tag{1}$$

The value $A(\mathbf{x})$ denotes the physical property, ρ_i the density, m_j the mass, $\mathbf{x}_{ij}$ the distance between the evaluated particle i and its neighbor j, W the weighting kernel, and h the smoothing length.

In standard SPH, the pressure is computed from the density fluctuations of the fluid particles through ideal gas equation [6], which leads to perceivable compressibility artifact. To resolve this issue, WCSPH is introduced into

```
while simulating do
  for all particles do
    neighbor search
  for all particles do
    density computation according to (2)
    pressure computation according to (3)
  for all particles do
    pressure force computation according to (4)
    viscosity force computation according to (5)
    other forces computation
  for all particles do
    time integration
```

Algorithm 1: Standard SPH/WCSPH algorithm.

graphics community [18]. In WCSPH, the Tait equation is applied to enforce incompressibility of the fluids. While Tait equation is essentially an equation of state (EOS), it uses large stiff values to generate huge penalty force to avoid much smaller distance between fluid particles usually appearing in the compressible case. The basic SPH/WCSPH method is summarized in Algorithm 1. At the beginning of each physics simulation step, the neighboring particles need to be found and then used to calculate particle's physical attributes, such as density, pressure, and various forces acting on them. Particle's density is evaluated through the following equation:

$$\rho_i = \sum_j m_j W\left(\mathbf{x}_{ij}, h\right). \tag{2}$$

The pressure is calculated from Tait equation:

$$p_i = \frac{k\rho_0}{\gamma}\left(\left(\frac{\rho_i}{\rho_0}\right)^{\gamma} - 1\right), \tag{3}$$

where p_i is the pressure at particle i, k is a stiffness parameter to allow small density fluctuation, ρ_0 is the reference density, and γ is a user defined value which is usually set to 1 for SPH and 7 for WCSPH.

The pressure force and viscosity force on each particle are given by the following equations:

$$\mathbf{F}_i^{\text{pressure}} = -\frac{m_i}{\rho_i}\sum_j \frac{m_j}{\rho_j}\frac{p_i + p_j}{2}\nabla W\left(\mathbf{x}_{ij}, h\right), \tag{4}$$

$$\mathbf{F}_i^{\text{viscosity}} = \mu\sum_j \frac{m_j}{\rho_j}\left(\mathbf{v}_j - \mathbf{v}_i\right)\nabla^2 W\left(\mathbf{x}_{ij}, h\right), \tag{5}$$

where $\mathbf{v}_i$ and $\mathbf{v}_j$ denote individual velocities of particle i and j.

4. Sleepy Method

In standard WCSPH, the physical quantities of all particles are being traced and updated at each time step to obtain a correct and stable simulation, whereas this is not always necessary in terms of visual plausibility. Considering the rigid body simulations, some rigid bodies are regarded as sleepy if they satisfy specific criteria, meaning that we do not need to compute their physical properties. Similarly, we observe that in WCSPH, there are usually many particles that can be treated as quasi static, which means their moving states can be safely ignored during several consecutive time steps; thus lots of computational resources can be saved. In the following, we will discuss the sleepy strategy used in our framework as well as the solution to tensile instability problem in the SPH based methods.

4.1. Sleepy Strategy. In adaptively sampled SPH method, particles are categorized to different levels according to their radii. Similarly, we have partitioned the whole fluid particles into three different types, that is, nonsleeping particles, intermediate particles, and sleeping particles, as shown in Figure 2. However, we still use a regular particle sampling which is better at reproducing physical properties and more efficient at neighbor searching compared to spatial adaptive sampling method. This is because that the latter uses hierarchical data structures such as kd-trees to perform neighbor search which are expensive to construct.

The processing of nonsleeping particles is the most computationally demanding part of the overall simulation. The computing demand of intermediate particles is second only to nonsleeping ones since they are needed to model the virtual boundary conditions for nonsleeping particles. The sleeping particles mark the most important ones in our method because it is precisely on the sleeping particles that a large amount of computational overhead can be saved.

Similar to its concept in rigid body simulation, a sleepy particle does not move, nor does its current state imply that it means to move in the instant future. Thus computational resources are saved by ignoring calculation of their physical properties. Intuitively, these sleepy particles should be deep inside the fluid or under a slow motion, while the nonsleeping particles normally lie in physically complex or visually important regions such as the fluid surface.

As for the heuristic to determine nonsleepy particles, we choose the following criteria:

$$\begin{gathered} e_i \geq E_{\text{threshold}} \quad \text{or} \quad p_i \geq P_{\text{threshold}} \quad \text{or} \quad d_i \geq D_{\text{threshold}}, \\ e_i = \frac{1}{2}mv_i^2, \\ d_i = \frac{\sum_j \mathbf{x}_{ij} m_j}{\sum_j m_j} \geq D_{\text{threshold}}, \end{gathered} \tag{6}$$

where e_i is the kinetic energy of particle i, p_i is the local pressure, d_i is the distance from fluid particle to the normalized center of mass of its neighbors, and $E_{\text{threshold}}$, $P_{\text{threshold}}$, and $D_{\text{threshold}}$ are user defined values. Here we adopt three criteria for detecting nonsleepy particles:

(i) kinetic energy criterion $e_i \geq E_{\text{threshold}}$ means that the particles in fast moving fluid region should be marked as nonsleepy;

(ii) pressure criterion $p_i \geq P_{\text{threshold}}$ means that the particles in fluid region with high turbulence should be marked as nonsleepy;

(iii) distance criterion $d_i \geq D_{\text{threshold}}$ means that the particles near the fluid surface or boundary should be marked as nonsleepy.

Now that the nonsleepy particles can be identified through (6), the other two particle types can also be easily identified afterwards as will be discussed in Section 4.3

4.2. Tensile Instability. In standard SPH based methods, when fluid particles are under tensile stress state, their motions become unstable which appears in the form of either particle clumping or complete blowup. This instability is usually referred to as tensile instability. This is due to the negative pressures of the surface particles that have fewer neighbor particles than needed to maintain the rest density (see Figure 3(a)).

In SPH literature, truncating the pressure to nonnegative is a commonly used scheme to avoid tensile instability, but with the side effect that the particle cohesion will be reduced. In this paper, we introduce an artificial repulsive force to solve this problem:

$$\mathbf{F}^{\text{replusive}} = \sum_j \frac{km}{\rho_0} \frac{\left(|p_i| + |p_j|\right)}{2} \left(\frac{W\left(\mathbf{d}_{ij}, h\right)}{W\left(\overline{\mathbf{d}_{ij}}, h\right)} \right)^4 \nabla W\left(\mathbf{x}_{ij}, h\right), \tag{7}$$

where k is the repulsive coefficient, p_i and p_j are pressures of particle i and j, $\mathbf{d}_{ij}$ is the distance between these two particles, and $\overline{\mathbf{d}_{ij}}$ is the average distance from its neighbors to particle i.

When the pressures of fluid particles become negative, we use this artificial repulsive force instead of pressure force (4). Thus, unphysical attraction force as a result of the negative pressure becomes a positive repulsive force which can effectively remove the tensile instability. Figure 3 demonstrates that particle clumping problem that arises because of attraction forces between surface particles with negative pressures is avoided.

4.3. Sleepy SPH/WCSPH Algorithm. Algorithm 2 is our sleepy algorithm for compressible and weakly compressible fluid simulations. Initially, we determine a subset of the fluid as nonsleepy particles according to (6). After that, we apply a flood fill technique to these particles. Specifically, we first perform neighbor search for nonsleepy particles. Then, those neighboring particles of the nonsleepy ones with a magnitude of kinetic energy larger than a user defined threshold are flood filled to be nonsleepy. Last, we mark the remaining neighbor particles as intermediate. Thus, similar to the reactivation concept in rigid body simulation, the sleepy fluid particles could be reactivated again by simply updating their velocities. Afterwards, the sleepy particles can be easily identified by marking all the rest particles as sleepy. After all particle types have been identified, we perform neighbor search, density, and pressure computation as well as the time integration for only a part of fluid particles.

We notice that in each time step, neighbor search, density, and pressure calculation are only needed for nonsleepy and intermediate particles; force calculation and time integration are only necessary for nonsleepy particles. Hence, lots of computational overhead can be saved compared to the standard SPH/WCSPH algorithm.

```
while simulating do
  determine different particle types
  for intermediate particles do
    neighbor search
  for non-sleepy & intermediate particles do
    density computation according to (2)
    pressure computation according to (3)
  for non-sleepy particles do
    if pressure is non-negative
      pressure force computation according to (4)
    else
      repulsive force computation according to (7)
    viscosity force computation according to (5)
    other forces computation
  for non-sleepy particles do
    time integration
```

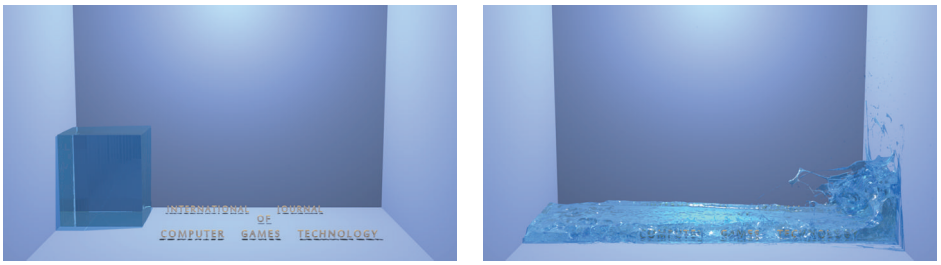
Algorithm 2: Sleepy SPH/WCSPH algorithm.

5. Implementation and Results

5.1. Implementation Details. In our implementation, we use cubic spline kernel as described in [30] for all calculations of physical attributes of fluid particles. The smoothing length is set to twice the rest distance between particles to make sure there are usually 30 to 40 neighbors for most particles. For the neighbor searching, a uniform grid is used to accelerate the process since it is a major performance bottleneck of the whole physics simulation. The time step is chosen to be the largest value satisfying CFL conditions. For the complex boundary handling such as letters as shown in Figure 1, we first sample the surface of static solids with boundary particles using the method from [31] and then adopt approaches from [11] to evaluate physical properties. While for simply shaped boundaries such as walls, we apply wall weight function idea from [32] since it can reduce the computational overhead compared to [11] in this case.

For our sleepy algorithm, we perform neighbor search for nonsleeping particles together with particle type determination at the beginning of each simulation step. Thus, the computation waste of another loop for type determination can be avoided. Our code is implemented in C++, and all experiments have been performed on an Intel Core i7 3.4G CPU with 8 GB of RAM. For surface reconstruction and fluid rendering, the technique from [12] is used.

5.2. Visual and Performance Results. In this section, we present several scenes simulated with our sleepy method in the context of WCSPH, which show that our algorithm can obtain similar visual results and is more efficient than standard WCSPH. Moreover, our technique can be also easily integrated with existing SPH method usually used in game physics engines.



(a) (b)

Figure 1: Corner dam break around static obstacles, 500 k fluid particles. Compared with standard weakly compressible smoothed particles hydrodynamics (WCSPH), our approach can achieve similar visual results at a higher efficiency.

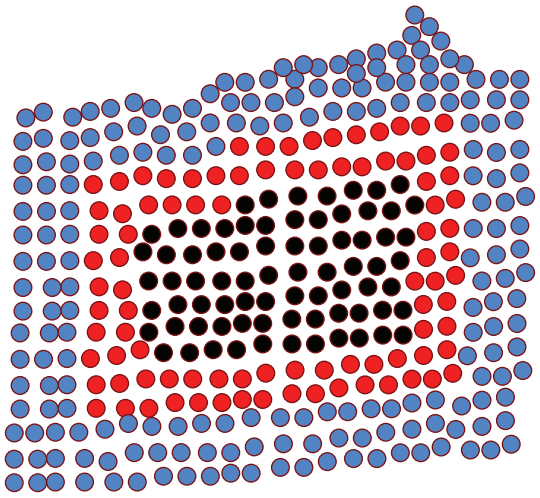
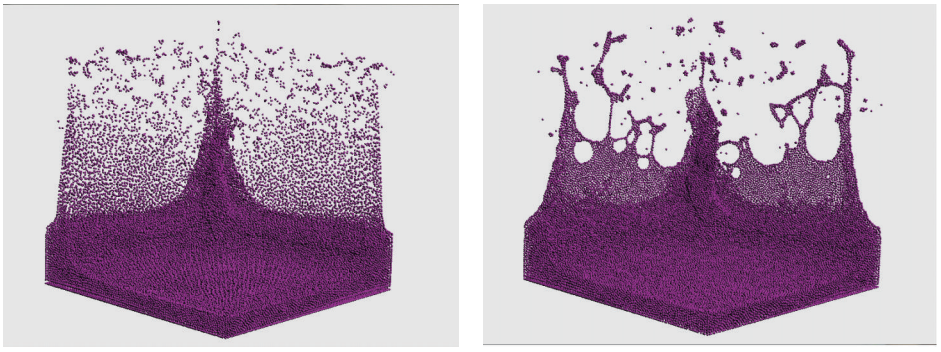


Figure 2: The fluid particles are partitioned into three different types: nonsleeping particles (blue), intermediate particles (red), and sleeping particles (black). The update of physical properties of sleeping particles can be saved in our method, leading to a performance boost without affecting visual fidelity.

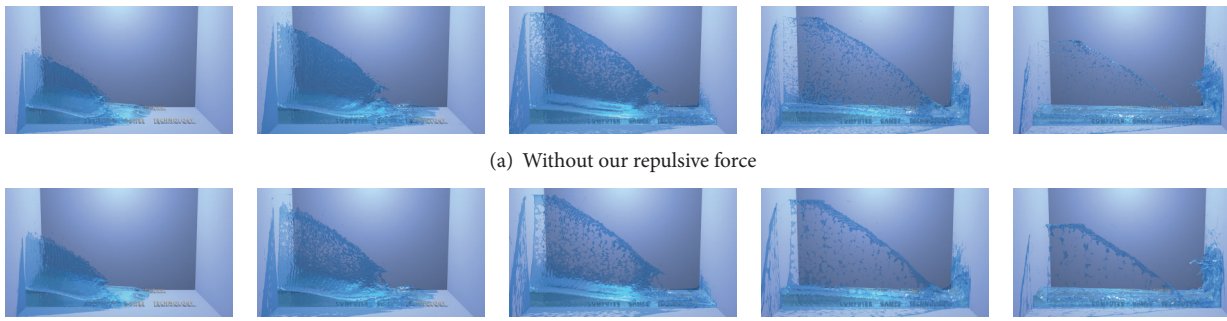


(a) (b)

Figure 3: Tensile instability. (a) Without our repulsive force, particles cluster because of the negative pressures. (b) With our repulsive force, particles distribute more uniformly with our artificial repulsive force. This example also shows that our method can generate surface tension like effects, as can be seen from (b).

Table 1: Performance comparison of our method with standard WCSPH for Figure 1.

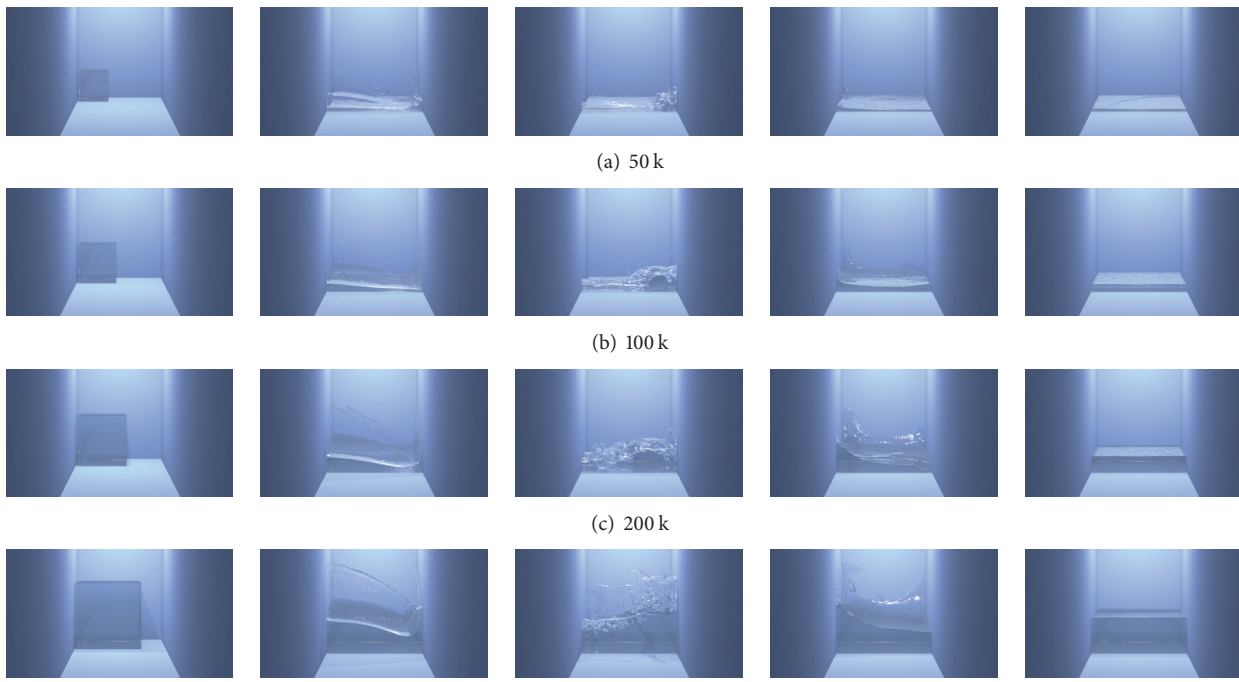
Scene	Number of fluid particles	Simulation time (minutes)		Speedup
		Standard WCSPH	Sleepy WCSPH	
Water around letters	500 k	31059	10183	3.05



(a) Without our repulsive force

(b) With our repulsive force

Figure 4: Corner breaking dam (a) without repulsive force and (b) with repulsive force. These scenarios are simulated using our sleepy WCSPH. Note that the fluid surface is now more naturally reconstructed and surface tension like effects can be achieved in the lower figures.



(a) 50 k

(b) 100 k

(c) 200 k

(d) 400 k

Figure 5: Corner dam break using (a) 50 k, (b) 100 k, (c) 200 k, and (d) 400 k fluid particles, respectively. This example shows that the greater the proportion of nonsleepy particles is, the larger the speed up can be achieved.

Figure 1 shows weakly compressible water with 500 k fluid particles flow around static letters sampled with 9 k boundary particles; the corner breaking dam collides with static letters and then comes to rest. This example is created using sleepy WCSPH together with boundary handling methods from [11, 32]. Note that there is little difference between the resulting visual quality of our method and the standard WCSPH, while our approach is typically faster. In Table 1, we make a performance comparison between standard WCSPH and our sleepy method. From the table, we can note that a speedup of 3.05 can be achieved using our method compared to standard approach.

In Figure 4, we make a visual comparison between the final effects with and without our repulsive force model based on sleepy WCSPH. From the figures we can see that the tensile instability issue is avoided and surface tension like visual effects is obtained as well as the particle distribution has been improved due to the proposed repulsive force model. Therefore, the splash effects are more naturally captured with our method as can be seen from Figure 4(b).

To illustrate the scaling of our method with the portion of the sleepy particles, we animate corner dam break scenes with different number of particles (see Figure 5) and compare their performance results in Table 2. As shown, when simulating scenarios with most particles under fast moving state as is the case in Figure 5(a), the speedup of our technique is small. This is because, under this circumstance, not too much computational overhead could be saved since portion of sleepy particles is relatively small. On the other hand, if the percentage of sleepy particles is larger as is the case in

Table 2: Speedup comparison of four different particle counts for Figure 5.

Scene	Number of fluid particles	Simulation time (minutes)		Speedup
		Standard WCSPH	Sleepy WCSPH	
Corner dam break	50 k	858	608	1.41
Corner dam break	100 k	2149	1149	1.87
Corner dam break	200 k	7121	2707	2.63
Corner dam break	400 k	24109	6150	3.92

Figure 5(d), a higher performance boost can be obtained. On the basis of this analysis, we can draw a conclusion that our method performs better if a larger portion of the whole fluid particles are under sleeping state.

6. Conclusion and Future Work

In this paper, we have proposed an efficient method for taking advantage of the sleepy state of particles in a compressible and weakly compressible fluid simulator that allows focusing computational resources only on parts of the whole fluid volume. Experimental results show that our method can capture similar realistic fluid effects with less computational needs. Our sleepy technique is also suitable for particle-based fluids widely applied in current generation game physics engines to get a further speedup. Additionally, we have introduced a new artificial repulsive force to avoid particle clumping as a result of neighbor deficiencies near the fluid surface.

As future work, we plan to speed up our algorithm by using multicore CPUs and many-core GPUs and by integrating adaptive time step techniques.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgments

The authors thank the reviewers for their helpful comments. This work is supported by National High Technology Research and Development Program of China (no. 2012AA011503) and the preresearch project (no. 51306050102).

References

[1] NVIDIA, "NVIDIA GameWorks PhysX Overview," 2014, https://developer.nvidia.com//gameworks-physx-overview.

[2] B. Adams, M. Pauly, R. Keiser, and L. J. Guibas, "Adaptively sampled particle fluids," *ACM Transactions on Graphics (TOG)*, vol. 26, no. 3, article 48, pp. 1–7, 2007.

[3] J. J. Monaghan and R. A. Gingold, "Smoothed particle hydrodynamics-theory and application to nonspherical starts," *Monthly Notices of the Royal Astronomical Society*, vol. 181, pp. 375–389, 1977.

[4] L. B. Lucy, "A numerical approach to the testing of the fission hypothesis," *The Astronomical Journal*, vol. 82, pp. 1013–1024, 1977.

[5] J. Stam and E. Fiume, "Depicting fire and other gaseous phenomena using diffusion processes," in *Proceedings of the 22nd Annual ACM Conference on Computer Graphics and Interactive Techniques*, pp. 129–136, ACM, August 1995.

[6] 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, 2003.

[7] 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.

[8] N. Akinci, J. Cornelis, G. Akinci, and M. Teschner, "Coupling elastic solids with smoothed particle hydrodynamics fluids," *Computer Animation and Virtual Worlds*, vol. 24, no. 3-4, pp. 195–203, 2013.

[9] M. Müller, B. Solenthaler, R. Keiser, and M. Gross, "Particle-based fluid-fluid interaction," in *Proceedings of the ACM SIGGRAPH/ Eurographics Symposium on Computer Animation*, pp. 237–243, July 2005.

[10] B. Solenthaler and R. Pajarola, "Density contrast SPH interfaces," in *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, pp. 211–218, 2008.

[11] N. Akinci and B. Solenthaler, "Versatile rigid-fluid coupling for incompressible SPH," *ACM Transactions on Graphics (Proceedings SIGGRAPH)*, vol. 30, no. 4, pp. 72:1–72:8, 2012.

[12] B. Solenthaler, J. Schläfli, and R. Pajarola, "A unified particle model for fluid-solid interactions," *Computer Animation and Virtual Worlds*, vol. 18, no. 1, pp. 69–82, 2007.

[13] M. Muller, R. Keiser, A. Nealen, M. Pauly, M. Gross, and M. Alexa, "Point based animation of elastic, plastic and melting objects," in *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation (SCA '04)*, pp. 141–151, Eurographics Association, Aire-la-Ville, Switzerland, 2004.

[14] K. Iwasaki, H. Uchida, Y. Dobashi, and T. Nishita, "Fast particle-based visual simulation of ice melting," *Computer Graphics Forum*, vol. 29, no. 7, pp. 2215–2223, 2010.

[15] N. Akinci, "Versatile surface tension and adhesion for SPH fluids," *ACM Transactions on Graphics*, vol. 32, no. 6, article 182, 2013.

[16] H. Schechter and R. Bridson, "Ghost SPH for animating water," *ACM Transactions on Graphics—SIGGRAPH Conference Proceedings*, vol. 31, no. 4, article 61, 2012.

[17] T. Lenaerts, B. Adams, and P. Dutré, "Porous flow in particle-based fluid simulations," *ACM Transactions on Graphics*, vol. 27, no. 3, article 49, 2008.

[18] 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, 2007.

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

[20] 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.

[21] J. Cornelis, M. Ihmsen, A. Peer, and M. Teschner, "IISPH-FLIP for incompressible fluids," *Computer Graphics Forum*, vol. 33, no. 2, pp. 255–262, 2014.

[22] M. Macklin and M. Muller, "Position based fluids," *ACM Transactions on Graphics*, vol. 32, no. 4, article 104, 2013.

[23] M. Desbrun and M. P. Cani, "Space-time adaptive simulation of highly deformable substances," Tech. Rep. 3829, INRIA, 1999.

[24] Y. Zhang, B. Solenthaler, and R. Pajarola, "Adaptive sampling and rendering of fluids on the GPU," in *Proceedings Symposium on Point-Based Graphics*, pp. 137–146, 2008.

[25] H. Yan, Z. Wang, J. He, X. Chen, C. Wang, and Q. Peng, "Real-time fluid simulation with adaptive SPH," *Computer Animation and Virtual Worlds*, vol. 20, no. 2-3, pp. 417–426, 2009.

[26] J. Orthmann and A. Kolb, "Temporal blending for adaptive SPH," *Computer Graphics Forum*, vol. 31, no. 8, pp. 2436–2449, 2012.

[27] M. Ihmsen, N. Akinci, M. Gissler, and M. Teschner, "Boundary handling and adaptive time-stepping for pcisph," in *Proceedings of the 7th Workshop on Virtual Reality Interactions and Physical Simulations (VRIPHYS '10)*, pp. 79–88, November 2010.

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

[29] C. J. Horvath and B. Solenthaler, "Mass preserving multi-scale SPH," Pixar Technical Memo 13-04, Pixar Animation Studios, 2013.

[30] J. J. Monaghan, "Smoothed particle hydrodynamics," *Reports on Progress in Physics*, vol. 68, no. 8, pp. 1703–1759, 2005.

[31] N. Bell, Y. Yu, and P. J. Mucha, "Particle-based simulation of granular materials," in *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, pp. 77–86, July 2005.

[32] T. Harada, S. Koshizuka, and Y. Kawaguchi, "Smoothed particle hydrodynamics on GPUs," in *Proceedings of the Computer Graphics International*, pp. 63–70, 2007.