Eurographics/ ACM SIGGRAPH Symposium on Computer Animation (2012)
N.N. and N.N. (Editors)

Energy-Centric Adaptive Particle Fluid Simulation based on Wavelet Analysis

paper1077

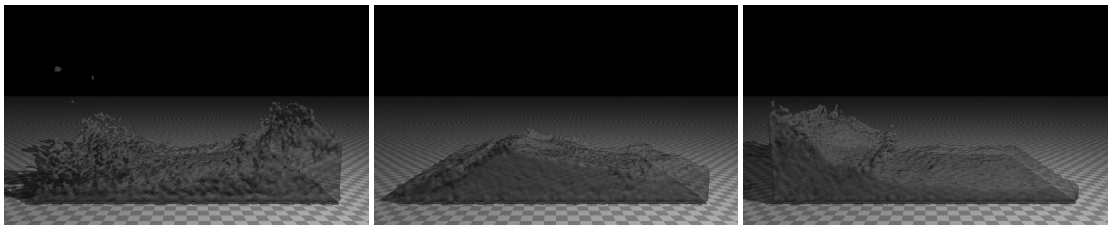

Figure 1: *Fluid simulation with our wavelet energy based approach*

Abstract

Simulation of SPH fluid with adaptive, multiple-sized particles could improve the performance of simulation while still retaining dynamic details with high-fidelity visual effects. One key element in adaptive SPH is the criterion guiding particle splitting and merging. In this paper, we propose a new criterion for adaptively-sampled particle fluid. Our method is energy-centric. By the analysis of the energy of each particle in the wavelet-transformed frequency domain, we could detect the turbulent region effectively and accurately because our approach is strictly physically based and the analysis is not only in spatial domain but also in frequency domain. We split particles in high turbulent region and merge particles in non-turbulent region dynamically. Since wavelet analysis is an efficient way to detect turbulent region, unlike previous methods, geometry information such as distance from particle to fluid surface could be completely ignored in the interest of time performance. By the demonstration, we show our algorithm could effectively sample regions where more details of flow are required.

Categories and Subject Descriptors (according to ACM CCS): I.3.7 [Computer Graphics]: Computer Graphics—Three-Dimensional Graphics and Realism

1. Introduction

Fluid simulation necessarily requires very high discretization resolution to preserve dynamic details and generate appealing visual effects. Towards these goals, a high resolution grid is required in Eulerian grid method, and a very large number of particles are demanded in Lagrangian Smoothed Particle Hydrodynamics (SPH) method. However, increasing the resolution unavoidably gives rise to heavy computational cost. Therefore, it is much more desirable to only allocate computing resources to regions where complex flow behavior emerges. With the help of octree data structure, [LGF04] makes highly realistic water behavior, by adaptively solving the full three-dimensional Navier-Stokes equations in a grid.

There are also some works addressing this problem in Lagrangian method, especially, the SPH method. By dynamically and adaptively splitting large particles into small particles and merging small ones into large ones, we could use large particles to simulate the gross shape of fluid but small particles to animate turbulent features like splash and wave. We could sample the turbulent areas according to geometry [APKG07], physics [HHK08] or both [YWH*09].

In this paper, we propose a new criterion guiding particle splitting and merging. Our method is strictly physically based. It may be noted that, [HHK08] is also based on physics. In [HHK08], Hong et al. used the concept of Reynolds number as part of criterion. But this Reynolds number is only computed in spatial domain. In [HHK08], to detect turbulent region, they should also take the distance from particle to surface into consideration to assist guiding. In our method, we use the wavelet-transformed energy as criterion. Wavelet analysis is an efficient and accurate mathematical tool to analyze data in different scales. In both [KTJG08] and [MGT*11], it is used as tool to detect and compensate lost energy to preserve details. To the best of our knowledge, the work proposed in this paper is the first attempt to utilize wavelet analysis as a tool to help guide particle splitting and merging. Since our analysis is also in frequency domain, we have a more global knowledge about the overall energy and its distribution. We do not need to take the surface distance into consideration. We just take the surface particles into consideration and this may not obviously decrease performance. In [APKG07], [HHK08] and [YWH*09], a marching method is always used to compute the distance from particle to surface. This process is time consuming. In our method, this process is removed.

Since the merging and splitting of particles in different level of scales may overburden any desktop animation system by introducing heavy computational cost, we simulate the fluid with particles in only two scales. Since we only have two scales of particles in our simulation, attributes like kernel length and mass are fixed for particles in each layer. It is now possible for us to cache these attributes in constant memory in GPU to facilitate the implementation in CUDA.

Our method has the following contributions:

- We propose a new criterion to guide particle splitting and merging. This represents the first attempt to use wavelet analysis to guide particle splitting and merging in adaptive SPH fluid simulation. This is mathematically rigorous because it is based on energy and it is also conducted in frequency domain. Our method is laid upon a solid foundation in both mathematics and physics.
- We remove the geometry information (i.e., the distance from particle to fluid surface) from the criterion popularly used to guide adaptive SPH. Instead, we simply use the surface particle to assist the splitting and merging. Hence, our method is totally physically based and it is physically correct.

2. Related Works

In this part, we will talk about the previous works. Since our method is a particle based method, we should primarily discuss the particle based methods that have been developed previously.

Generally, there are two physically based ways to simulate fluids, Eulerian method and Lagrangian method. They are all based on Navier-Stokes equation. Eulerian method solves the Navier-Stokes equation over a grid. [FM96] were the first to apply the Navier-Stokes equation to 3D liquid simulation in computer graphics field for incompressible fluid. But this algorithm was not stable enough. Later, [Sta99] designed an algorithm to guarantee the stability of fluid simulation by utilizing a semi-Lagrangian scheme to handle velocity advection. This algorithm is always stable for arbitrary time steps. Because of the computational cost of Eulerian method, it is hard to simulate fluid with a grid in real-time. [CM11] proposed a new Eulerian fluid simulation method in real-time by utilizing a hybrid grid representation composed of regular cubic cells on top of a layer of tall cells.

Because of the numerical dissipation introduced in Eulerian method, the particle based Lagrangian methods play a more and more important role in recent research and the most promising approach among them are SPH method. [MCG03] were the first to introduce the SPH method to the simulation of water. Later, the SPH method was extended to the simulation of interaction between different fluids [MSKG05] and animation of viscoelastic fluid [CBP05]. There is also research about parallel implementation of SPH algorithm [HKK07].

Besides fluids, people also simulate smoke by the combination of particle system and grid system method, like FLIP [ZB05]. By generating some votex particles in the grid and coupling the particles with grid based on vorticity confinement, [SRF05] could improve the turbulent effects for water, smoke and explosion. [GLHB09] could simulate multiple speed smoke by simulating the low speed smoke in grid and high speed smoke in particle system. [CTG10] proposed an interactive system featuring fluid-driven animation that could respond to moving objects. It would generate a grid coupled with the moving object and simulate the smoke near the object in the grid system. For particles out of the gird, the smoke would be animated in pure particle system, obscuring the point at which the fluid simulation ends.

Because energy driven approach is always strictly physics based in computer graphics, energy based fluid and smoke simulation is also a hot topic. We could compensate the lost energy back to the simulation system to preserve the details. [KTJG08] analyzes the wavelet-transformed energy in the low resolution grid system in frequency domain and then synthesizes the wavelet turbulence energy to the high resolution grid to preserve turbulence features. Instead of grid system, [MGT*11] applied the algorithm in [KTJG08] to the particle system to conserve the wavelet turbulence energy in SPH. In [KTJG08], we do not need to transfer the particle attribute to the auxiliary grid but directly work in the point cloud. Instead of isotropic noise, [PTC*10] added the procedural approach based turbulence by generating and advecting turbulence particles based on anisotropic noise. Simula-

tion based on anisotropic noise could make a more appealing visual effect than simulation based on isotropic noise.

In particle system, the computing efficiency could be improved by constructing the particle system adaptively. It can make the distribution computing resources reasonable and reduce the complexity of calculation. For regions that do not need to preserve details, fewer particles would be enough; while for physically and visually important region, a larger number of particles are required. [DG96] first applied the adaptive particle system to the simulation of highly deformable models. [APKG07] analyzed the particle system based on the extended local feature size to decide when to split or merge particles. This concept of extended local feature size is geometry based, but the turbulent regions computed with this method obey the rules in physics. Unlike [APKG07], which simulates compressible fluid, [HHK08] could simulate incompressible fluid by the application of the FLIP [ZB05] algorithm. This method computes the advection with SPH but analyzes the deformability of fluid and conserves incompressibility by transferring the particle attribute value to the auxiliary grid system. [YWH*09] sampled the turbulent region based on both geometry and physics. They introduced the non-uniform particle system and proposed a generalized distance field function. They utilized the pressure to evaluate the turbulence of fluid based on physics. Thus, the auxiliary grid is not required. [SG11] presented a two-scale particle method based on the idea to simulate distinct particle sizes in individual but coupled simulations. Since traditional adaptive SPH method is not hardware based, [ZSP08] presented a new GPU-friendly algorithm for weakly compressible adaptive SPH.

Compared to other topic, research works on adaptive particle fluid simulation is not many. Therefore, we propose energy based adaptive SPH method in this paper. Our method is physically based. We also separate the concepts of kernel and radius. We use particle kernel to handle interaction between particles and particle radius to handle collision with boundary and obstacles.Unlike [HHK08], it is not only in spatial domain but also in frequency domain. Figure 1 is the example generated with our approach.

3. SPH Model

We first introduce the basic knowledge of SPH fluid simulation. SPH is an interpolation method for particle system. In SPH, the fluid is composed of a set of particles with interparticle forces such as pressure and viscosity computed at the position of a particle by a smoothing kernel. We define field quantities, such as velocity and density, at discrete particle locations anywhere in space. Instead of original attribute value, we use the smoothed attribute value to guide particle behavior. We can interpolate a scalar quantity A at location r by a weighted sum of contributions from all neighboring particles with equation (1):

$$A_S(r) = \sum_j \frac{A_j}{\rho_j} W(r - r_j, h) \qquad (1),$$

where j is the index of all neighboring particles, m is the mass of particle, r is its position, ρ the density, A the field quantity at position r and h is the kernel length of the particle.

Function $W(r,h)$ is the radial kernel function which is used to compute the smoothed field quantity according to the contribution from neighbors. For more details about SPH method and its kernel functions, please check [MCG03].

Generally, the governing force equation of SPH is as follows:

$$F_{Total} = F_{Pressure} + F_{Viscosity} + F_{External} \qquad (2).$$

The total force applied on each particle is the sum of forces from pressure, viscosity and other external forces like gravity and surface tension. After we get the total force, we compute the acceleration of each particle with the following equation:

$$a_i = \frac{F_{Total}}{\rho_i} \qquad (3).$$

According to [MCG03], the equation to compute density is:

$$\rho_i(r) = \sum_j m_j W(r - r_j, h) \qquad (4).$$

Based on the constant gas equation, we could yield the pressure for each particle:

$$p = \kappa(\rho - \rho_0) \qquad (5),$$

where ρ_0 is the rest density of fluid. The equations for pressure force and viscosity force are:

$$F_i^{Pressure} = -\sum_j m_j \frac{p_i + p_j}{2\rho_j} \nabla W(r - r_j, h) \qquad (6),$$

$$F_i^{Viscosity} = \mu \sum_j m_j \frac{v_j - v_i}{\rho_j} \nabla^2 W(r - r_j, h) \qquad (7).$$

4. Adaptive Particle Fluid Simulation

In this section, we will describe our algorithm to simulate fluid with adaptive two scale particles.

4.1. Wavelet Energy

In [APKG07], the splitting and merging of particles are based on geometry. In [HHK08] this process is based on physics, that is, the deformability of fluid. Our method is also based on physics. In previous adaptive SPH paper, people primarily use their algorithm to detect the region with

high deformability. The highly deformable regions are always near the surface of fluid. We would like also detect turbulent region in the inner fluid.

In such scenarios, details are everywhere. Therefore, we propose a new adaptive SPH method based on wavelet-transformed energy in our paper. By the combination of particle energy and surface particle together, we propose new criteria for particle splitting and merging.

In the filed of physics, we use the following equation to compute the energy of particle:

$$E_i = \frac{1}{2}mv^2 \qquad (8).$$

However, this equation is only in spatial domain. To detect the region where the turbulence will occur, we analyze the energy in both spatial and frequency domain. Fourier transform is one solution to obtain information in both the spatial and frequency domains. [KTJG08] solved this problem by utilizing a wavelet transform to compute the wavelet-transformed energy of each grid cell. [MGT*11] analyzed the energy distribution in a particle system in both spatial and frequency domain with wavelet-transformed energy. In the particle system, a grid does not exist that can explicitly define the neighborhood value. They directly determine the wavelet transform $\hat{u}$ of the velocity field u and the energy spectrum $\hat{e}$ of e, at a scale s by taking a weighted sum of neighboring particles and using the SPH method. We could use the following equation in [MGT*11] to compute the transformed velocity in frequency domain:

$$\hat{u}_i = \frac{1}{\sqrt{s}\psi_{sum}} \sum_j u_j \psi\left(\frac{x_i - x_j}{s}, \frac{y_i - y_j}{s}, \frac{x_i - z_j}{s}\right) \quad (9),$$

where $\hat{u}$ is the wavelet transform of velocity u, s is the wavelet scale, ψ is the mother wavelet function and

$$\psi_{sum} = \sum_j \psi \qquad (10).$$

We use Mexican hat wavelet as the mother wavelet. Figure 2 is the graph of Mexican Hat wavelet function. We are the first attempt to apply this equation to guide adaptive particle fluid simulation. The reason we use wavelet is that it has very solid foundation in mathematics and physics and it is proved that wavelet analysis can provide us with accurate global knowledge of data.

In [MGT*11], the computation of wavelet-transformed velocity is only with particles of one scale. But we have particles in two scales in our work. The equation to compute wavelet-transformed velocity in this paper is the same as in [MGT*11]. We use different scale s for particles in different size. s is usually the same as the kernel length h. After generating the transformed velocity, we compute the energy in frequency domain with:

$$\hat{e}_i = \frac{1}{2}m_i\hat{u}_i^2 \qquad (11).$$

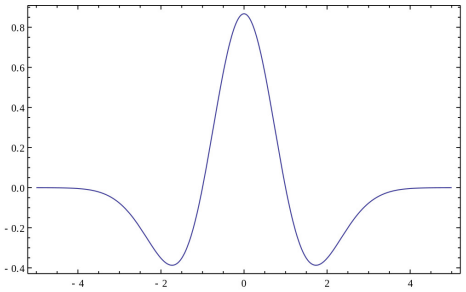

Figure 2: *Mexican Hat Wavelet*

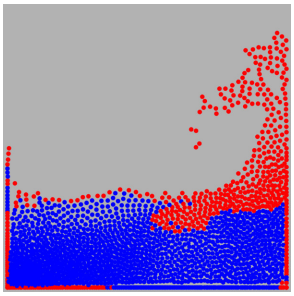

Figure 3: *Energy distribution in 2D*

Figure 3 is the energy distribution generated with the equation (11). The red particle is in high energy level and blue one is in low energy level. Figure 4 is the energy distribution in 3D. As both figures show, the wavelet-transformed energy in frequency domain could indicate the turbulent region.

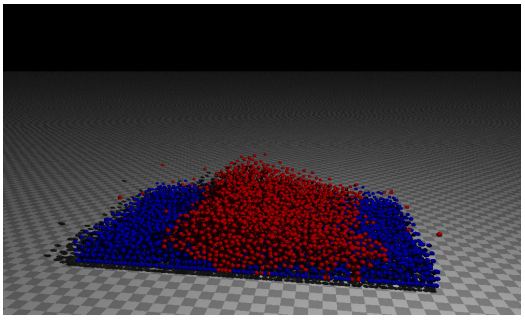

Figure 4: *Energy distribution in 3D*

Fluid surface is the place where details always occur. Since energy could accurately help detect where is turbulent, we do not need to take the surface distance into consideration. We just use particles on the fluid surface to assist particle splitting and merging. We directly apply the method in [MCG03] to our algorithm. In [YWH*09], Yan et al. first compute the surface particle and then calculate the depth value for all particles with marching method. Since this is

time consuming and we could detect turbulent regions with our energy-centric analysis, we only take the surface particles into consideration.

As in [MCG03], the surface of fluid can be founded by using an additional field quantity which is 1 at particle location and 0 the everywhere else. We call this the color field. The equation for the smoothed color field c is:

$$c_i = \sum_j \frac{m_j}{\rho_j} W(r_i - r_j, h) \qquad (12).$$

The gradient of the smoothed color field is:

$$n = \nabla c_i \qquad (13),$$

and its normalized scalar field is $|n|$. Given a threshold l, if $|n|$ is larger than the threshold l, the particle is on the fluid surface.

Since we are able to detect both high energy particles and surface particles, we could establish our criterion to guide particle splitting and merging. By combining the high energy particles and surface particles together, we get the following equation:

$$D_i = \alpha \cdot \hat{e} + \beta \cdot |n_i| \qquad (14),$$

α and β are user defined coefficients. For each particle i, if D is larger than a threshold k_1, we may split the particle into small particles. If D is smaller than another threshold k_2, we may merge small particles into one large particle. We use different threshold to guide particle splitting and merging because we hope to avoid the unstable oscillation between merging and splitting. $k_1 > k_2$.

4.2. Two Scale SPH Equation

Since we now have particles in two scales in our particle system, we could not use the original SPH interpolation equation in [MCG03]. We extend the interpolation equation in [APKG07] to our algorithm. We use the same equation in [MCG03] to compute the density for the particle. However, we modified the constant gas equation as follows to calculate pressure:

$$\rho = \kappa((\frac{\rho}{\rho_0})^3 - 1) \qquad (15)$$

Since two particles in different scales have different kernel length and mass, we modify equations in [APKG07] as follows:

$$F_i^{Pres} = -\sum_j m_j m_i (\frac{p_i}{\rho_i^2} + \frac{p_j}{\rho_j^2})(\frac{\nabla W(x_{ij}, h_i) + \nabla W(x_{ij}, h_j)}{2}) (16),$$

and

$$F_i^{Visc} = \mu \sum_j V_i V_j (v_j - v_i) \frac{\nabla^2 W(x_i j, h_i) + \nabla^2 W(x_i j, h_j)}{2} (17),$$

with $V = \frac{m}{\rho}$ the particle volume.

4.3. Merging and Splitting

We use similar method as in [HHK08] to merge and split particles. However, in this paper, we separate the concepts of kernel length and radius of particles. We use kernel length to compute the interaction between particles, such as SPH interpolation and wavelet transform. To handle boundary condition, we use the value of particle radius. Based on our experiment, equations in [HHK08] may not strictly guarantee the conservation of volume. It is only theoretically based. Therefore, we modify the equations in [HHK08] a little.

When merging a group of particles, the new particle is placed at the center of gravity of the original particles. We can compute the mass, kernel length, position and velocity of the newly created particle i from all neighbor particles j by:

$$m_i = \sum_j m_j \qquad r_i = \sqrt[3]{\sum_j r_j^3} \qquad h_i = \varepsilon \sqrt[3]{\sum_j h_j^3}$$

$$x_i = \frac{\sum_j x_j V_j}{\sum_j V_j} \qquad u_i = \frac{\sum_j u_j V_j}{\sum_j V_j}$$

where r is the particle radius and h is the particle kernel length. We use user defined ε to reconcile the sub-particle kernel length to avoid the obvious change of volume.

Unlike merging, the splitting of a large particle is performed by creating a new set of n particles. Usually, n is 2, 4 or 8. In our demo, we use $n = 4$. With a larger n we could generate better visual effect. The mass, kernel length, radius length and velocity of each new particle are computed by the following equations:

$$m_j = \frac{m_i}{n} \qquad \varepsilon h_j = \sqrt[3]{\frac{h_i^3}{n}} \qquad r_j = \sqrt[3]{\frac{r_i^3}{n}} \qquad u_j = u_i$$

There are multiple ways to determine the position of the new generated sub particles. We choose to place them randomly at the position within the radius of the original large particle. For more details on particle splitting and merging, please check reference [HHK08].

We summarize the algorithm in Algorithm 1.

5. Implementation and Result

We have implemented our SPH simulator with C++ and CUDA in GPU. All examples in this paper are implemented on a machine with two 2.67 GHZ Core i5 CPU, 2.92 GB memory and NVIDIA NVS 3100m graphics card with 16 cores. After extrapolating the surface triangular mesh with marching cube, we render the fluid with POV-Ray.

We show three examples in Figure 5, Figure 6 and Figure 7. In each figure, the pictures on the left are examples rendered directly in particles. Red particles are in large scale and yellow particles are in small scale. One large particle

Algorithm 1 Wavelet based Adaptive SPH

1: Compute the density for each particle with equation (4).
2: Compute the pressure for each particle based on constant gas equation (15).
3: Compute the smoothed pressure for each particle based on equation (16).
4: Compute the smoothed viscosity force for each particle based on equation (17).
5: Compute the surface value and surface tension for each particle [MCG03].
6: Compute velocity based on the total force applied to particle.
7: Compute the wavelet-transformed energy for each particle in the frequency domain.
8: Adaptively split and merge particles. For each particle we get D based on (14). If $D > k_1$, we split the particle. If $D < k_2$, we merge the particles.
9: Advect the particle to the new position.

is equivalent to four small particles. For high-turbulence region, we split the large particle into four small particles. In non-turbulence region, we merge four small particles into one large particle. In Figure 5, we simulate the fluid with 8000 large particles according to the original SPH algorithm [MCG03]. In Figure6, we adaptively split and merge particles according to our algorithm in this paper. We initially have about 32k small particles and the number of particles keeps changing during the simulation. We set n as 4. For particles that fulfill the criteria of splitting, we may split each particle into four small particles. For particles fulfill the requirement of merging, we may merge four small particles into one large particle. And in Figure 7, we simulate the fluid with original SPH method directly with about 32k small size particles. Merging and splitting will not happen during simulation.

In Figure 5, since we use only 8000 large particles, features like splash are not obviously. We could only simulate the gross shape and behaviour of fluid. After we increase the number of particles to 32k in Figure 7, we could have features like wave and splash. But simulation with large number of particles is time consuming, compared to example in Figure 5. In Figure 6, we adaptively merge and split particles. Compared to Figure 5, the visual effect in Figure 6 is effectly improved. And there is no obvious visual difference between Figure 6 and Figure 7. High-fidelity visual details are preserved. We could see the large red particles under yellow particles in Figure 6. Even without taking geometry into consideration, we are still able to effectively detect high-turbulence area.

Table 1 lists the performance statistics of our simulations. In Figure 5 the FPS is about 20. In Figure 7, after we increase the number of particles to 32k, the FPS becomes 0.7. In Figure 6 the FPS is about 2.3. Therefore, our method is able to improve the performance of simulation while preserving details. Table 2 lists the parameter values in our simulation.

We also show the simulation of falling water ball with our algorithm in Figure 8 and Figure 9. In Figure 8, fluid is directly rendered as particles. In Figure 9, fluid is rendered with POV-Ray. For the case of falling water ball, our algorithm could also effectively and accurately detect where we need to split particles and where we need to merge particles.

Simulation	Particles	FPS(s)
Figure5	8000	20
Figure6	32k	2.3
Figure7	32k	0.7

Table 1. Simulation Statistics

6. Conclusion and Future Work

We have presented a novel technique for particle based fluid simulation that uses refinement and simplification based on wavelet-transformed energy. We use both the wavelet-transformed energy and surface particles to guide the particle splitting and merging. We first attempt to guide adaptive particle fluid simulation in an wavelet-transformed energy driven approach. Since we use wavelet to analyze energy distribution, our method is strictly physically based. Even without involving the distance from particles to fluid surface, our method could accurately detect where to split particles and where to merge particles. With demonstrations in different cases, we show that our method could improve the performance of simulation while preserving high quality visual details. However, the analysis in our method is only isotropic. We hope to extend our method in the future with anisotropic analysis.

Simulation Parameters	
Large kernel length	1.2
Large particle mass	1.0
Large particle radius	1.2
ε (Adaptive coefficient)	1.5
Gravity value	-4.0
Rest density	0.5
Gas constant coefficient	1.0
Viscosity	1.0
Time step	0.1
α (Energy coefficient)	1.0
β (Surface coefficient)	1.5
k_1 (Splitting threshold)	1.0
k_2 (Merging threshold)	0.6
Simulation World	$32 \times 16 \times 16$
n (Adaptive number)	4

Table 2. Simulation Parameters

References

[APKG07] ADAMS B., PAULY M., KEISER R., GUIBAS L. J.: Adaptively sampled particle fluids. *ACM Trans. Graph. 26*, 3 (July 2007). doi:10.1145/1276377.1276437. 1, 2, 3, 5

[CBP05] CLAVET S., BEAUDOIN P., POULIN P.: Particle-based viscoelastic fluid simulation. In *Proceedings of the 2005 ACM SIGGRAPH/Eurographics symposium on Computer animation* (2005), pp. 219–228. doi:10.1145/1073368.1073400. 2

[CM11] CHENTANEZ N., MÜLLER M.: Real-time eulerian water simulation using a restricted tall cell grid. *ACM Trans. Graph. 30*, 4 (July 2011), 82:1–82:10. doi:10.1145/2010324.1964977. 2

[CTG10] COHEN J. M., TARIQ S., GREEN S.: Interactive fluid-particle simulation using translating eulerian grids. In *Proceedings of the 2010 ACM SIGGRAPH symposium on Interactive 3D Graphics and Games* (New York, NY, USA, 2010), I3D '10, ACM, pp. 15–22. doi:10.1145/1730804.1730807. 2

[DG96] DESBRUN M., GASCUEL M.-P.: Smoothed particles: a new paradigm for animating highly deformable bodies. In *Proceedings of the Eurographics workshop on Computer animation and simulation '96* (New York, NY, USA, 1996), Springer-Verlag New York, Inc., pp. 61–76. 3

[FM96] FOSTER N., METAXAS D.: Realistic animation of liquids. *Graph. Models Image Process. 58*, 5 (Sept. 1996), 471–483. doi:10.1006/gmip.1996.0039. 2

[GLHB09] GAO Y., LI C.-F., HU S.-M., BARSKY B. A.: Simulating gaseous fluids with low and high speeds. *Computer Graphics Forum 28*, 7 (2009), 1845–1852. doi:10.1111/j.1467-8659.2009.01562.x. 2

[HHK08] HONG W., HOUSE D. H., KEYSER J.: Adaptive particles for incompressible fluid simulation. *Vis. Comput. 24*, 7 (July 2008), 535–543. doi:10.1007/s00371-008-0234-z. 1, 2, 3, 5

[HKK07] HARADA T., KOSHIZUKA S., KAWAGUCHI Y.: Smoothed particle hydrodynamics on gpus. *Structure* (2007), 1–8. 2

[KTJG08] KIM T., THÜREY N., JAMES D., GROSS M.: Wavelet turbulence for fluid simulation. *ACM Trans. Graph. 27*, 3 (Aug. 2008), 50:1–50:6. doi:10.1145/1360612.1360649. 2, 4

[LGF04] LOSASSO F., GIBOU F., FEDKIW R.: Simulating water and smoke with an octree data structure. *ACM Trans. Graph. 23*, 3 (Aug. 2004), 457–462. doi:10.1145/1015706.1015745. 1

[MCG03] MÜLLER M., CHARYPAR D., GROSS M.: Particle-based fluid simulation for interactive applications. In *Proceedings of the 2003 ACM SIGGRAPH/Eurographics symposium on Computer animation* (Aire-la-Ville, Switzerland, Switzerland, 2003), SCA '03, Eurographics Association, pp. 154–159. 2, 3, 4, 5, 6, 8, 9

[MGT*11] MAKOTO F., GO M., TOSHIYUKI A., JUN M., HIROKAZU K.: A Fast Simulation Method Using SPH and Wavelet for Sub-Particle-Scale Turbulent Flow. pp. 67–72. doi:10.2312/PE/PG/PG2011short/067-072. 2, 4

[MSKG05] MÜLLER M., SOLENTHALER B., KEISER R., GROSS M.: Particle-based fluid-fluid interaction. In *Proceedings of the 2005 ACM SIGGRAPH/Eurographics symposium on Computer animation* (New York, NY, USA, 2005), SCA '05, ACM, pp. 237–244. doi:10.1145/1073368.1073402. 2

[PTC*10] PFAFF T., THUEREY N., COHEN J., TARIQ S., GROSS M.: Scalable fluid simulation using anisotropic turbulence particles. *ACM Trans. Graph. 29*, 6 (Dec. 2010), 174:1–174:8. doi:10.1145/1882261.1866196. 2

[SG11] SOLENTHALER B., GROSS M.: Two-scale particle simulation. *ACM Trans. Graph. 30*, 4 (July 2011), 81:1–81:8. doi:10.1145/2010324.1964976. 3

[SRF05] SELLE A., RASMUSSEN N., FEDKIW R.: A vortex particle method for smoke, water and explosions. *ACM Trans. Graph. 24*, 3 (July 2005), 910–914. doi:10.1145/1073204.1073282. 2

[Sta99] STAM J.: Stable fluids. In *Proceedings of the 26th annual conference on Computer graphics and interactive techniques* (New York, NY, USA, 1999), SIGGRAPH 99, Addison-Wesley Publishing Co., pp. 121–128. doi:10.1145/311535.311548. 2

[YWH*09] YAN H., WANG Z., HE J., CHEN X., WANG C., PENG Q.: Real-time fluid simulation with adaptive sph. *Comput. Animat. Virtual Worlds 20* (June 2009), 417–426. doi:10.1002/cav.v20:2/3. 1, 2, 3, 4

[ZB05] ZHU Y., BRIDSON R.: Animating sand as a fluid. *ACM Trans. Graph. 24*, 3 (July 2005), 965–972. doi:10.1145/1073204.1073298. 2, 3

[ZSP08] ZHANG Y., SOLENTHALER B., PAJAROLA R.: Adaptive sampling and rendering of fluids on the gpu. In *Volume Graphics* (2008), Eurographics Association, pp. 137–146. 3

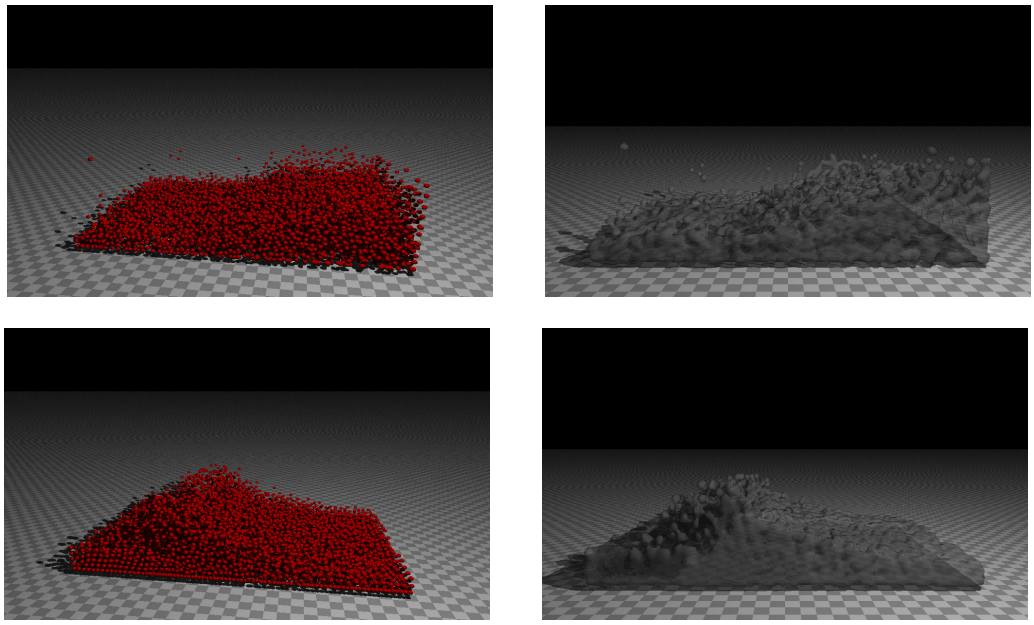

Figure 5: *The example implemented in original SPH [MCG03] method with 8000 large particles.*

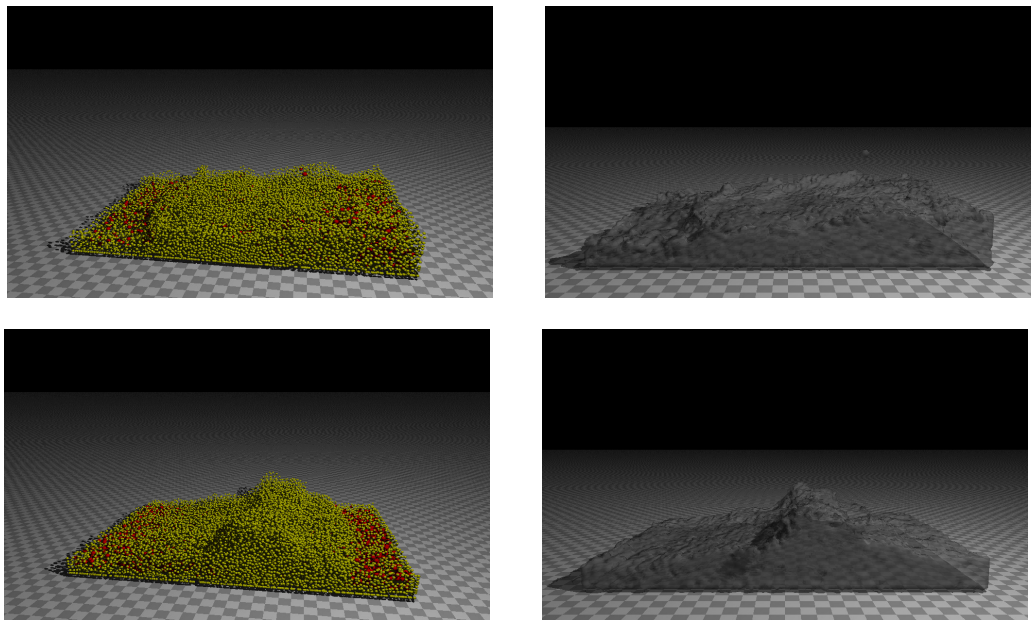

Figure 6: *The example implemented in our algorithm. There are initially 32k small particles. Splitting and merging are enabled.*

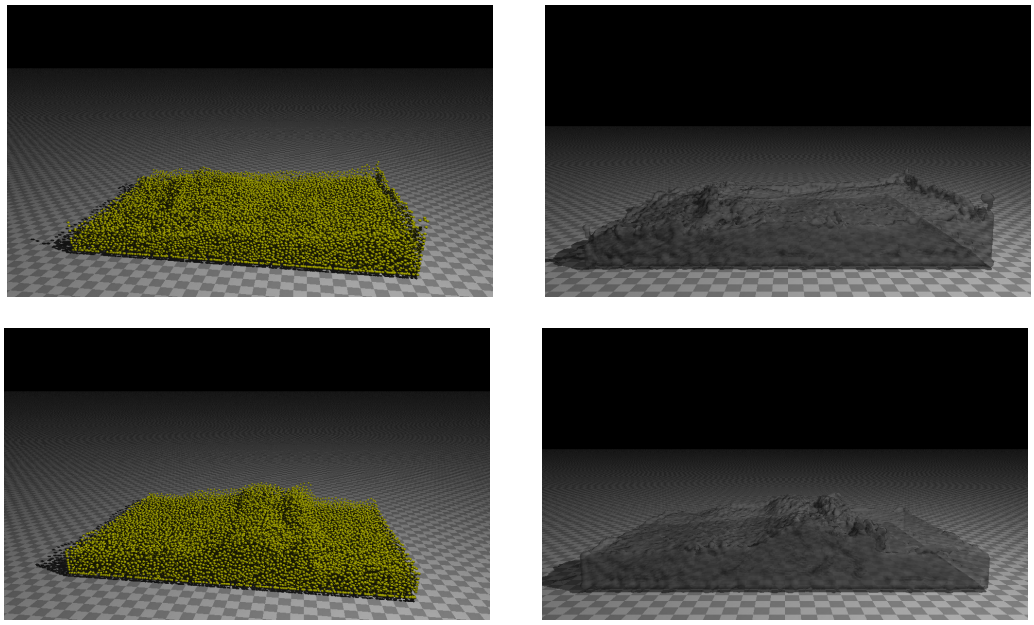

Figure 7: *The example implemented in original SPH [MCG03] method with 32k small particles.*

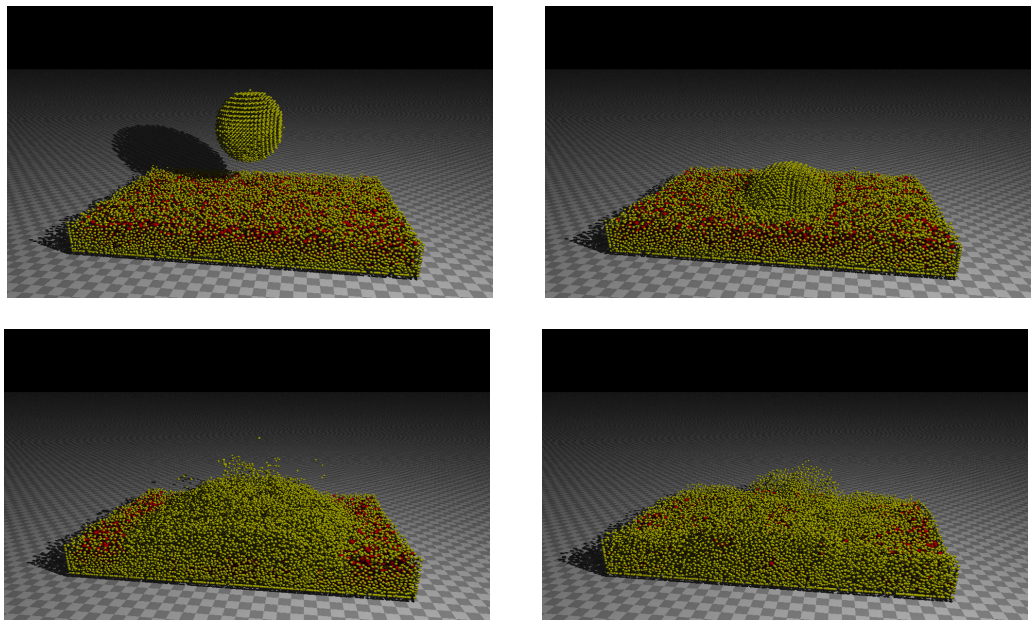

Figure 8: *Simulation of falling water ball in our algorithm with 32k initial small particles*

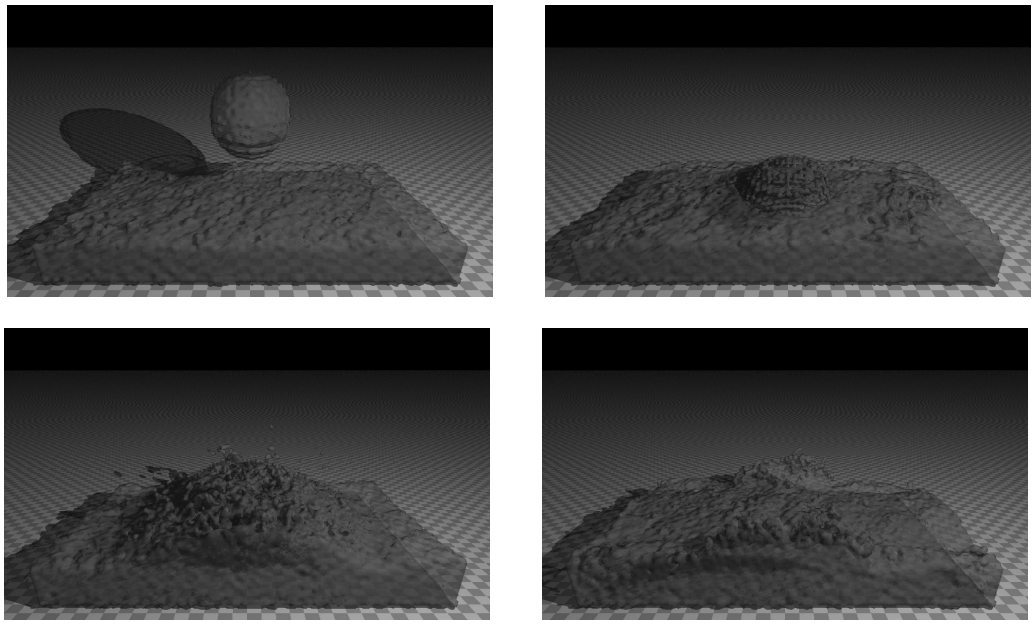

Figure 9: *Rendered simulation of falling water ball in our algorithm with 32k initial small particles*