Simulating SPH Fluid with Multi-Level Vorticity

Taekwon Jang, Roger Blanco i Ribera, Jinhyuk Bae and Junyong Noh

Visual Media Lab, Graduate School of Culture Technology, KAIST

***Abstract*— The vortical motion of fluid in various scales is one of the most important elements in capturing the vivid realism of turbulent water. However, it is still challenging to resolve multi-level vorticity effectively. This paper presents a novel hybrid method that combines a smoothed particle hydrodynamics (SPH) system with multiple Eulerian grids to reproduce the multi-level vorticity. In our hybrid method, the SPH system is responsible for resolving flow velocity while the multiple grids support the SPH system in efficiently detecting and computing the multi-level vorticity field.**

***Index Terms* — Fluid simulation; Multi-level approach; Vorticity; SPH;**

I. INTRODUCTION

The vortical motion of fluid is frequently observed both in real life and in computer animation. Examples include flow from a faucet, pouring water into a glass, and the natural stream of a river. With the vorticity, the flows often become turbulent and expose chaotic vortical motions. To capture this visually interesting effect, it is very important to resolve the vorticity in various scales. However, as the turbulence has a wide spectrum of velocity scales, it is generally difficult to capture the vivid reality through a numerical simulation. For example, high resolution is required for the spatial discretization which incurs a high computational cost.

The cost can be effectively reduced by using an adaptive approach to the simulation. However, a system using an adaptive method is still insufficient to simulate the multi-level vorticity [1] when it relies solely on the Eulerian approach or the Lagrangian approach. For example, Eulerian grid-based systems often employ more accurate numerical schemes or exploit an adaptive grid structure to overcome the difficulty in simulating the vortical motion. However, the required scales to resolve the small vortical motion are often smaller than the cell size of an adaptive grid, and the effects of the numerical diffusion are not negligible for the small scales.

Similarly, the Lagrangian particle system selectively increases the number of sampling particles at the regions of interest, such as near the free surface or around objects. However, the previous methods [2,3,4,5] emphasized only the spatial adaptiveness and paid little attention to the effects that last for a relatively long period of time, such as large scale eddies typically observed in water streams. It is inherently difficult to define time clustering operators for spatio-temporal adaptiveness due to the Lagrangian property of particles. The limitation makes the previous adaptive methods unsuitable for simulating vortical motions of fluid that vary both in spatial and temporal scales. Moreover, in spite of using an adaptive sampling method, the effects of artificial viscosity that occur during interpolation operations still exist in a smoothed particle hydrodynamics (SPH) based system.

In addition, Lagrangian particle methods suffer from a difficulty in determining accurately where and how much the vorticity should be enhanced or preserved [6]. For instance, in the typical vortex particle method [7], particles with a random initial vorticity are seeded at user specified regions in a certain frame. Without a spatial constraint, the vortex particles can freely move around in the whole domain, which often results in deformation at the unwanted location. As movement of the vortex particles are determined by Lagrangian advection and cannot be predicted before running the simulation, the deformation results are sometimes unpredictable and can look unnatural in some regions.

As an alternative, a hybrid approach can be employed to resolve the multi-level vorticity. The advantages from the Eulerian and Lagrangian methods can be combined to complement each other. For example, the vorticity in a scale that is smaller than the Eulerian grid size can be resolved by Lagrangian particles, which can diminish the effect of numerical diffusion from the grid system. The location and magnitude of the vorticity can easily be determined by taking the curl to the velocity field on the grid.

Based on the hybrid idea, this paper introduces a novel method that efficiently captures the multi-scale vortical motion in SPH fluid. A SPH system combined with multi-level Eulerian grids is used to compute the vorticity in various scales. Compared to a typical hybrid system that exploits an Eulerian grid system to handle large bodies of water but uses a Lagrangian particle system to generate small scale details, such as underwater bubbles [8] or water droplets in the air, our approach utilizes the particle system to resolve fluid velocities in every scale and exploits multiple Eulerian grids to support the particle

Manuscript Received on February 11, 2011.
E-mail: taekwon@kaist.ac.kr

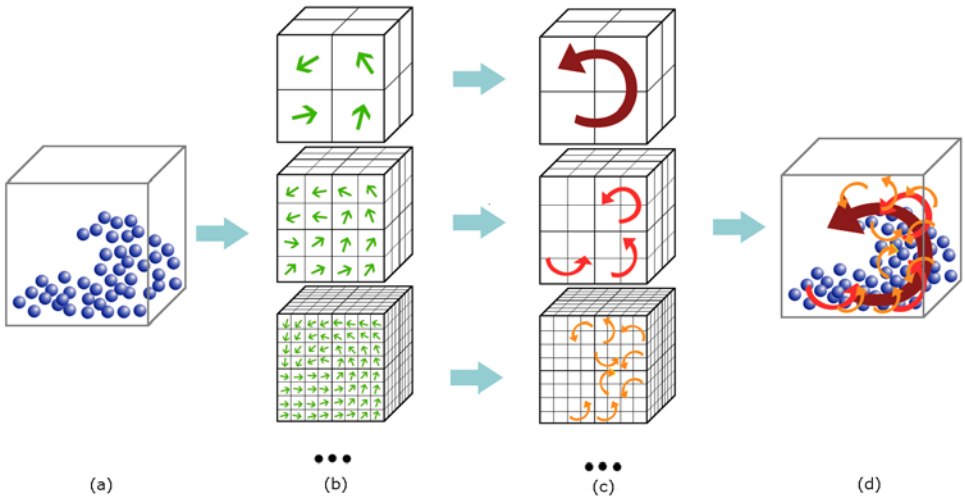


Fig. 1. The system overview. a) The particle velocities from a SPH system are transferred to the cells in multiple grids. b) Using the cell velocity, vorticity is computed at each cell. c) The vorticity fields from each level are combined and transferred to the particles in the SPH system. d) The multi-level particle vorticity is enhanced by applying the vorticity confinement to each particle.

system to efficiently resolve multi-level vorticity. In our setting, the multi-level vorticity is computed at each level by exploiting the regularity of an Eulerian grid as illustrated in Fig. 1. The detailed vortical motion is then reproduced by enhancing the vorticity around particles using a particle-based vorticity confinement method similar to [8]. We show examples with vortical motion in highly deformable regions to demonstrate that our system effectively reproduces multi-level vorticity for SPH fluid.

II. RELATED WORK

2.1 SPH

The original SPH [9] is an interpolation method that solves an astrophysics problem. The method was introduced to computer graphics by Desbrun and Gascuel [10] and employed in various applications including water simulation [11], multiple fluids [12, 13], viscoelastic fluid [14] and frothing liquid [15]. Despite its versatility, SPH suffers from the difficulty of enforcing incompressibility due to the inherent property of compressibility. Several methods [16, 17, 18] have been proposed to solve the problem. Recently, parallel and real-time simulations of SPH using GPU [4, 5, 19, 20] have gained popularity by exploiting the locality and the simplicity of the algorithm. Our particle system is also implemented with CUDA [21].

2.2 Vortex particle

The vortex particle method was introduced to the graphics community by Gamito et al. [22] and has received much attention for simulating turbulent motion. For example, a grid-particle hybrid method for water and smoke turbulence was proposed in [7]. Similarly, a Lagrangian-based method [23] was proposed to solve the vortex formulation including boundary conditions. To reproduce realistic turbulence effects around obstacles, vortex particles with energy dynamics equations were exploited at the precomputed boundary regions [24]. Recently, hybrid methods [25, 26] were proposed that exploit the vortex particles to synthesize small scale turbulence details on a low-resolution grid.

2.3 Adaptive approach

The basic idea of the adaptive approach is to change the particle resolution for the region of interest by spatial sampling. In computer graphics, Adams et al. [2] used the approach in water simulation by changing the particle resolution according to the geometric feature size. For example, a high resolution creates small splashes or thin sheet effects while a low resolution creates internal water volumes. Techniques with elaborate sampling methods, such as using a high pressure gradient [3] or a local pressure ratio and the number of neighbors [4, 5] were recently proposed.

2.4 Hybrid approach

A hybrid approach combines an Eulerian grid system and a Lagrangian particle system together to complement each other. Several variants have been proposed differing the degree of importance of each component to achieve a specific purpose. Early methods utilize auxiliary particles to assist the grid system with surface tracking or with correction for interface capturing [27]. In an effort to reproduce small scale details, use of particles started to replace the dissipative grid in the advection process [28]. Escaped particles are incorporated into the SPH system to simulate underwater bubbles [8]. A two-way coupling method [29] was proposed to simulate both

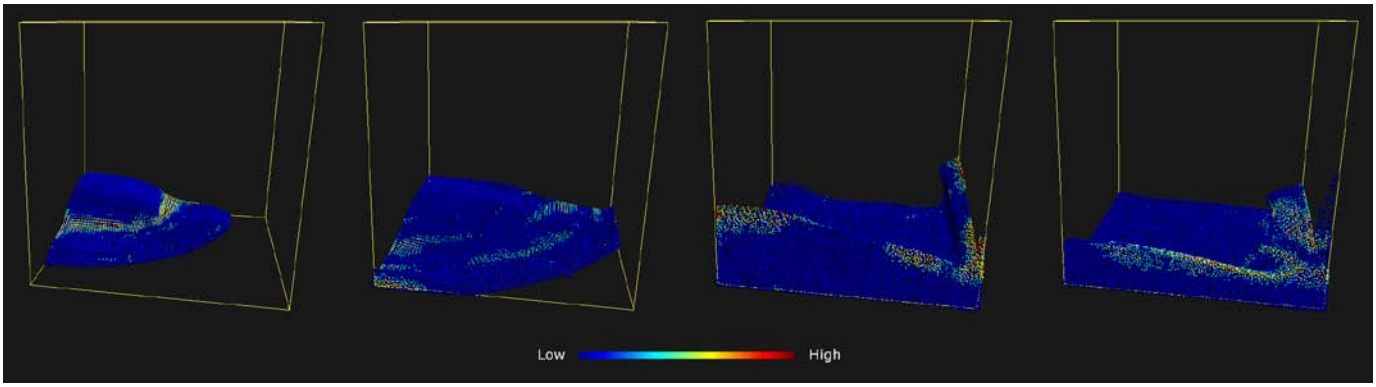


Fig. 2. Sequence of captured images from the simulation. The order start from left to right and the frame numbers are (90, 130, 216, 278). The magnitudes of the vorticity for each particles are represented with color spectrum, from low(blue) to high(red).

dense and diffuse fluid regions using the grid and SPH system, respectively.

Our system is a variant of the particle-grid approach, which is the opposite of previous grid-particle approaches in a hybrid domain. The particle-grid approach is mainly based on a particle system, which has the advantage of preserving sub-grid scale features from numerical diffusion. In contrast, the grid-particle methods require an expensive grid refinement process or an additional reseeding process to capture the sub-grid scale details. Another advantage of the particle-grid approach is that most of the simulation process can be parallelized easily, and can gain speedup through hardware acceleration. We focus on building a simple but effective solution that takes full advantage of the modern particle system while maintaining simplicity in the hybrid process. The steps for coupling are simplified and the parallel processing is maximally exploited.

III. PARTICLE-GRID HYBRID APPROACH

3.1 Incompressible SPH

The behavior of the SPH fluid is described by the approximation of the momentum equation

$$\langle \rho_i \rangle \frac{\partial v_i}{\partial t} = -\langle \nabla p \rangle (x_i) + \mu \langle \Delta v \rangle (x_i) + f_i^{ext}$$

where x_i is the particle position, v_i is the particle velocity, p is the pressure, μ is the viscosity constant, and f_i^{ext} is an external force term for gravity, user defined forces, or vorticity confinement forces. $\langle \rho_i \rangle$, $\langle \nabla p \rangle$ and $\langle \Delta v \rangle$ symbolize the interpolation kernel-based approximation of the density field, pressure force field, and viscous force field, respectively. In our implementation, we employ a poly kernel for field interpolation, a spiky kernel to approximate the gradient, and a viscosity kernel for the computation of the Laplacian, respectively.

In order to minimize the effects of spurious bouncing motion during the particle simulation, we apply the predictive-corrective incompressible SPH (PCISPH) solver [18]. In PCISPH, the pressure value for each particle is updated by solving the following prediction-correction scheme iteratively.

$$p_i += \delta(\rho_i^* - \rho_0)$$

where ρ_i^* is the predicted density, δ is the scaling variable, and ρ_0 is the rest density. In our test scenes, we precompute the δ value using a randomly picked internal particle. With the fixed scaling factor δ, we repeat the prediction-correction step at each frame with 3 iterations to enforce the incompressibility of the SPH fluid as in [18].

3.2 Multi-level grids

To resolve the multi-level vortical motion stably and efficiently, we exploit the spatially uniform Eulerian MAC grid. We generate multiple grids with different resolutions. The number of levels n and the ratio r between each level are provided by the user. The generated grids correspond to multiple spatial sub-samplings of the domain. In the highest resolution grid, the size of a cell is the same as the SPH particle's smoothing radius and the time interval is the same as the time-step of the SPH simulation. A cell size d_i and a time interval t_i at each grid are determined by applying the Courant-Friedrichs-Lewy (CFL) condition to each level

$$\frac{u \cdot \Delta t_i}{\Delta d_i} \le k_{cfl}$$

$$\Delta d_i = \left(\frac{1}{r}\right)^{j-i} \Delta d_j$$

$$\Delta t_i = \left(\frac{1}{r}\right)^{j-i} \Delta t_j$$

where $i < j \le n$ and u represents the velocity, and k_{cfl} is a system parameter. The multiple cell sizes and time intervals are

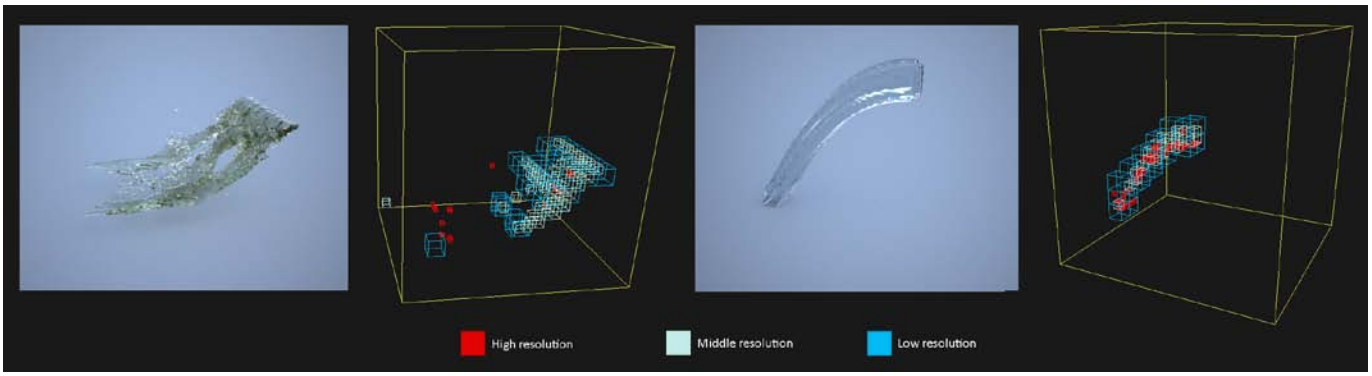


Fig. 3. The hybrid detection of deformable regions at multiple levels. Detected cells from three different levels are represented with different sizes and colors.

required to capture the vortical motion with different scales and speeds. For example, the motion of a large eddy that occupies many cells can be captured in the low resolution grid, which has larger cells and a longer time interval.

3.3 Hybrid deformation detection

The goal of our hybrid system is to exploit the regularity of the Eulerian grid to complement the SPH particle system in resolving the multi-level vorticity. For the detection of the deformation, we propose a hybrid technique. We formulate the particle deformation detection process as a local eigen-analysis on grid cells. The Principle Component Analysis (PCA) applied to the particles in each cell computes the variation of particles with respect to the X,Y, and Z axis of the cell. Assuming that a cell with coordinate (i, j, k) has the center position c^l in level l and number n of particles p, the covariance matrix Cov^l for the cell is

$$Cov^l(i,j,k) = \begin{bmatrix} p_1 - c^l \\ p_2 - c^l \\ \dots \\ p_n - c^l \end{bmatrix}^T \begin{bmatrix} p_1 - c^l \\ p_2 - c^l \\ \dots \\ p_n - c^l \end{bmatrix}$$

It is possible to encode the deformation for each particle. For example, we can change the c^l to a particle centric criterion like a centroid $\overline{p} = 1/|N|(\sum_{i \in N} p_i)$. However, the particle-based deformation encoding often requires attention in handling an exceptional case, such as a lump of particles or outliers. While the cell based particle deformation encoding is an approximation based on lower resolution, it handles the irregular situations including deficiency of neighboring particles better than the particle-based deformation encoding. In addition, other particle centric criteria, such as the local pressure ratio, the number of neighboring particles [4], or the combination of both [5] are dependent on the accuracy of kernel interpolation, which is vulnerable to the deformation. Moreover, for different scene settings, a time consuming trial and error process is required to determine a reasonable threshold for a single criterion or optimal weights for the criteria combination.

To detect the deformable regions, we examine the eigenvalues of the Cov^l. As the matrix is 3x3 positive semi-definite, all of its eigenvalues $\lambda_m (m \in \{0,1,2\})$ are real. The equations to quantify the degree of deformation D^l for each cell c^l are

$$\frac{\Delta V^l}{\Delta t^l} = \sqrt{\sum_{m=0}^{2} (\lambda_m^t - \lambda_m^{t-1})^2}$$

$$D^l(i,j,k) = \left| \frac{\Delta V^l}{\Delta t^l} \right| \geq k_{deform}$$

where V^l is the total variance of particles at cell c^l at level l. We measure the change of variations along each axis during time interval Δt^l. These equations effectively detect the cells with high deformation. For example, if the particles are in rapid divergent or convergent motion, the change of variance in each direction will be high enough to mark the container cell as deformable.

3.3 Hybrid vorticity enhancement

Once deformable cells are detected, we can compute the multi-level vorticity using multiple grids. The process begins by constructing a velocity field for each grid. A weighted average of the particle velocities produces the cell velocity u.

$$u(i,j,k) = \sum_i \left(\frac{d_i v_i}{\sum d_i} \right)$$

where v_i is the particle velocity and d_i is the distance between the particle and the cell center. A simple curl operation based on the finite difference method (FDM) applied to the given velocity field u^l estimates the vorticity field ω^l at each level.

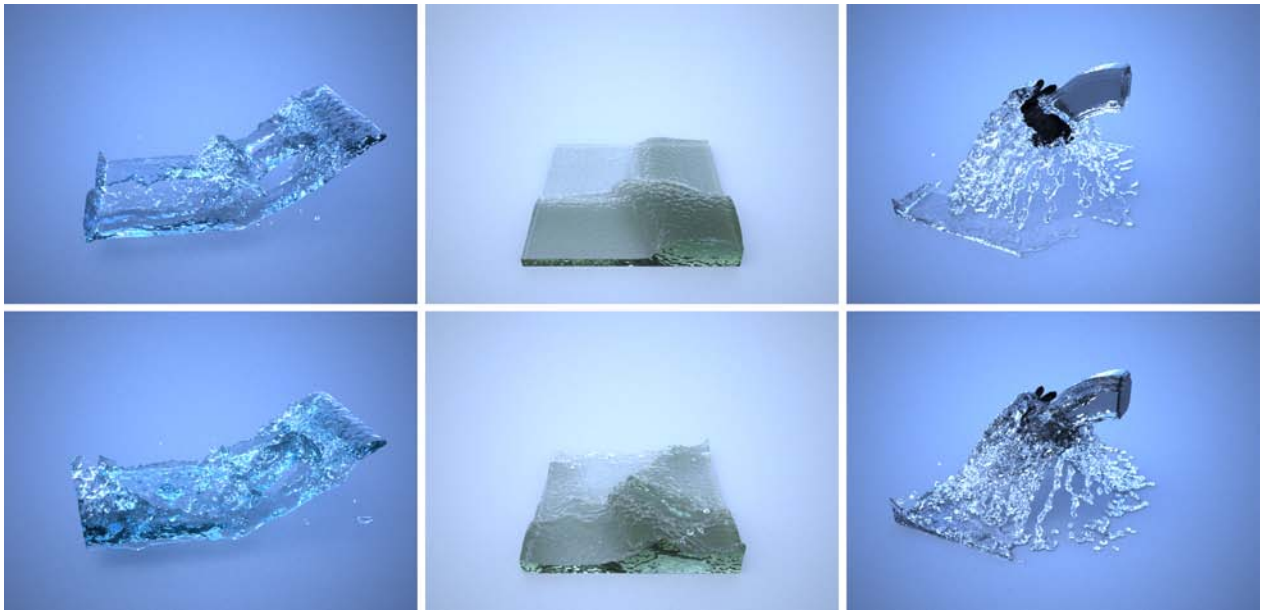


Fig. 4. Simulation results with SPH (top row) and with our method (bottom row). (a) water stream simulation with object collision (b) one dam-breaking (c) pouring water on a bunny model.(Color Plate 4)

$$\omega = \sum_l \omega^l = \sum_l (\nabla \times u^l)$$

By accumulating the multiple vorticity fields, we get a single vorticity field ω. The particle vorticity ω^l is computed by tri-linear interpolation on the field. For each particle, the vorticity confinement force is computed only using neighboring particles.

$$f^{vorticity} = \varepsilon \left(N \times \frac{\omega_i}{|\omega_i|} \right) \rho_i$$

where ε is a user parameter, the vorticity location $N = (p_\oplus - p_i) / |p_\oplus - p_i|$ with $p_\oplus$ being the mass center of two SPH particles.

IV. RESULTS AND DISCUSSION

All results are obtained on a 2.27 GHz Intel Xeon CPU with 8 GB of memory. We used uniform regular grids for computing the vorticity fields. For all of our experiments, we used three grids to represent the different levels. The resolutions are 42x42x42 for the high resolution grid, 21x21x21 for the mid resolution grid, and 11x11x11 for the low resolution grid, respectively. We set a fixed integration time step of 0.002s in all the examples.

Our system was implemented using CUDA to maximize the usage of parallel computation. The most frequent and time-consuming process in the particle system is to search for the nearest neighbors. A particle hashing is applied to the auxiliary grids followed by the radix sort algorithm for efficiency. We can achieve a real-time performance on the simulation for our test examples. We also use a CUDA-based marching cube with a grid (128x128x128) to gain a speed-up in the mesh generation process, which now takes only several seconds in our system.

We reuse the grids to calculate the cell-based velocity and the vorticity. The multi-level grids are extended auxiliary grids with different resolutions. They are stored in the GPU memory as 1D array. An individual CUDA thread is activated and run in parallel for each cell's coupling process. For example, the vorticity confinement in SPH evokes as many CUDA threads as the number of cells to utilize the velocity and the vorticity defined in each cell. The association of the cell and the particles inside is registered in the GPU memory and updated at every frame. The usage of the GPU accelerates the particle-grid coupling process. In addition, we further improve the performance when solving for the eigenvalue of a 3-by-3 matrix by implementing a simple kernel function in the GPU.

The second column in Fig. 4 shows the result of a simulation of a single dam-breaking scene. The simulation was carried out using 44,649 particles. We can notice that the surface has more deformation with our method. Although, the vortical motions are somewhat exaggerated by adjusting the user parameter, the effects of the multi-level vorticity make the flow look like a natural turbulence. A user can adjust the epsilon values to emphasize the vorticity at a specific level or to modify the effective ranges of the vorticity. We use 1.0, 0.15 and 0.01 as the epsilon values for the scene. While a tedious trial and error procedure is often required to determine the epsilon values, they can be reused for several scenes once they are set.

In the third column, we demonstrate a scene of pouring water on a bunny object. In contrast to the previous example, we can see the effects of the multi-level vorticity from the particles at the thin surface. After hitting the bunny, the motions of the particles become a little bit more scattered with the generated multi-level vorticity.

V. CONCLUSION

We present a new particle-grid hybrid system to efficiently reproduce multi-level vortical motion in SPH fluid. By combining a SPH system with multiple Eulerian grids, we can detect the deformable region robustly while efficiently computing the vorticity for particles. With the flexibility and the simplicity of our multiple grid system, one can easily extend our system to a wider spectrum in aspects of space in time. In a future study, we plan to extend our framework to resolve the sub-particle scale vortical motions as well..

ACKNOWLEDGEMENT

This work was supported by culture contents industry research and development program of KOCCA/MCST (210-7602-003-10743-01-007, Software Development for 2D to 3D Stereoscopic Image Conversion).

REFERENCES

[1] Taekwon Jang, Heeyoung Kim, Jinhyuk Bae, Jaewoo Seo, and Junyong Noh. *Multilevel vorticity confinement for water turbulence simulation. Vis. Comput.*, 26:873–881, June 2010.

[2] Bart Adams, Mark Pauly, Richard Keiser, and Leonidas J. Guibas. Adaptively sampled particle fluids. In ACM SIGGRAPH 2007 papers, *SIGGRAPH '07*, New York, NY, USA, 2007. ACM.

[3] Woosuck Hong, Donald H. House, and John Keyser. *Adaptive particles for incompressible fluid simulation. Vis. Comput.*, 24:535–543, July 2008.

[4] He Yan, ZhangyeWang, Jian He, Xi Chen, Changbo Wang, and Qunsheng Peng. Real-time fluid simulation with adaptive sph. *Comput. Animat. Virtual Worlds*, 20:417–426, June 2009.

[5] Jian He, Xi Chen, Zhangye Wang, Chen Cao, He Yan, and Qunsheng Peng. *Realtime adaptive fluid simulation with complex boundaries. Vis. Comput.*, 26:243–252, April 2010.

[6] Swets Zeitlinger, Jason Frank, and Sebastian Reich. C*onservation properties of smoothed particle hydrodynamics applied to the shallow water equations*, 2003.

[7] Andrew Selle, Nick Rasmussen, and Ronald Fedkiw. A vortex particle method for smoke, water and explosions. In ACM SIGGRAPH 2005 Papers, *SIGGRAPH '05*, pages 910–914, New York, NY, USA, 2005. ACM.

[8] Jeong-Mo Hong, Ho-Young Lee, Jong-Chul Yoon, and Chang-Hun Kim. Bubbles alive. In ACM SIGGRAPH 2008 papers, *SIGGRAPH '08*, pages 48:1–48:4, New York, NY, USA, 2008. ACM.

[9] J. J. Monaghan R.A. Gingold. Smoothed particle hydrodynamics: theory and application to nonspherical stars. *Monthly Notices of the Royal Astronomical Society*, pages 375–389, 1977.

[10] Mathieu Desbrun and Marie paule Gascuel. Smoothed particles: A new paradigm for animating highly deformable bodies. pages 61–76. Springer-Verlag, 1996..

[11] Matthias Muller, David Charypar, and Markus Gross. Particle-based fluid simulation for interactive applications. *SCA '03*, pages 154–159, Aire-la-Ville, Switzerland, 2003. Eurographics Association..

[12] Matthias Muller, Barbara Solenthaler, Richard Keiser, and Markus Gross. Particle-based fluid-fluid interaction. *SCA '05*, pages 237–244, New York, NY, USA, 2005. ACM.

[13] B. Solenthaler and R. Pajarola. Density contrast sph interfaces. *SCA '08*, pages 211–218, Aire-la-Ville, Switzerland, 2008. Eurographics Association.

[14] Simon Clavet, Philippe Beaudoin, and Pierre Poulin. Particle-based viscoelastic fluid simulation. *SCA '05*, pages 219–228, New York, NY, USA, 2005. ACM.

[15] Paul W. Cleary, Soon Hyoung Pyo, Mahesh Prakash, and Bon Ki Koo. Bubbling and frothing liquids. In ACM SIGGRAPH 2007 papers, *SIGGRAPH '07*, New York, NY, USA, 2007. ACM.

[16] Markus Becker and Matthias Teschner. Weakly compressible sph for free surface flows. *SCA '07*, pages 209–217, Aire-la-Ville, Switzerland, 2007. Eurographics Association.

[17] Kai Bao, Hui Zhang, Lili Zheng, and Enhua Wu. Pressure corrected sph for fluid animation. Comput. Animat. *Virtual Worlds*, 20:311–320, June 2009.

[18] B. Solenthaler and R. Pajarola. Predictivecorrective incompressible sph. In ACM SIGGRAPH 2009 papers, *SIGGRAPH '09*, pages 40:1–40:6, New York, NY, USA, 2009. ACM.

[19] Takahiro Harada, Seiichi Koshizuka, and Yoichiro Kawaguchi. Smoothed Particle Hydrodynamics on *GPUs*. In Proc. of *Computer Graphics International*, pages 63–70, 2007.

[20] Prashant Goswami, Philipp Schlegel, Barbara Solenthaler, and Renato Pajarola. Interactive SPH Simulation and Rendering on the GPU. *SCA '10*, pages 1–10, 2010.

[21] NVIDIA. NVIDIA CUDA Programming Guide 2.0. 2008.

[22] Manuel Noronha Gamito, Pedro Faria Lopes, and Mario Rui Gomes. Twodimensional simulation of gaseous phenomena using vortex particles. 1995.

[23] Sang I. Park and Myoung J. Kim. Vortex fluid for gaseous phenomena. In *SCA '05*, pages 261–270, New York, NY, USA, 2005. ACM.

[24] Tobias Pfaff, Nils Thuerey, Andrew Selle, and Markus Gross. Synthetic turbulence using artificial boundary layers. *SIGGRAPH Asia '09*. ACM, 2009.

[25] Ho-Young Lee, Jeong-Mo Hong, and Chang-Hun Kim. Interchangeable sph and level set method in multiphase fluids. *Vis. Comput*., 25:713–718, April 2009.

[26] Tobias Pfaff, Nils Thuerey, Jonathan Cohen, Sarah Tariq, and Markus Gross. Scalable fluid simulation using anisotropic turbulence particles. *ACM SIGGRAPH Asia 2010 papers*, 2010.

[27] Douglas Enright, Stephen Marschner, and Ronald Fedkiw. Animation and rendering of complex water surfaces. In Proceedings of the 29th annual conference on Computer graphics and interactive techniques, *SIGGRAPH '02*, pages 736–744, New York, NY, USA, 2002. ACM.

[28] Yongning Zhu and Robert Bridson. Animating sand as a fluid. In ACM SIGGRAPH 2005 Papers, *SIGGRAPH '05*, pages 965–972, New York, NY, USA, 2005. ACM.

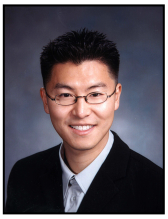
[29] Frank Losasso, Jerry Talton, Nipun Kwatra, and Ronald Fedkiw. Two-way coupled sph and particle level set fluid simulation. *IEEE Transactions on Visualization and Computer Graphics*, 14:797–804, 2008.

[30] Ye Fan Bo Zhu, Xubo Yang. Creating and preserving vortical details in sph fluid. *Computer Graphics Forum*, 29:2207–2214, 2010.

Taekwon Jang was born in S.Korea, October 7,1979, and received M.S. degree in culture technology from Korea Advanced Institute of Science and Technology (KAIST) in 2007. He is currently working towards his Ph.D. degree in the Visual Media Lab in the graduate school of culture technology at KAIST. Before starting his academic career, he worked at NTREEV soft for 4-years instead of serving in the military (a.k.a. tute service) and participated in several commercially shipped games including *Pang-Ya online*, *Trickster online*, and *whiteday* as a game programmer. As a graduate student, he participated in several research projects that are funded by government while experiencing various production -level developments including plug-in programming for Maya, Motion Builder, Real-flow and Houdini using C++. He also has more than 1 year of experience on CUDA programming. Current research interest is an efficient simulation of multi-level vorticity. As in his previous research, "Multi-level vorticity confinement for water turbulence simulation", he pursues to find practical technique to be used in real-world production pipeline with cost efficiency.

Roger Blanco i Ribera received the French engineering diploma in electrical engineering from the Institut National des Sciences Appliquees de Lyon (INSA) in 2006 and M.S. degree in electrical Engineering from the Korean Advance Institute of Science and Technology (KAIST) in 2008. He is currently working towards his Ph.D. degree in the Visual Media Lab in the graduate school of Culture Technology at KAIST.

Jinhyuk Bae received his B.S. degree in computer science in 2009 from Sungkyunkwan University, Suwon, Korea. He is currently working towards his M.S. degree at Korea Advanced Institute of Science and Technology. His research interests include fluid simulation.



Junyong Noh is an Associate Professor in the Graduate School of Culture Technology at the Korea Advanced Institute of Science and Technology (KAIST). He earned his computer science Ph.D. from the University of Southern California (USC) in 2002 where his focus was on facial modeling and animation. His research relates to human facial modeling/animation, character animation, fluid simulation, and stereo movie generation. Prior to his academic career, he was a graphics scientist at a Hollywood visual effects company, Rhythm and Hues Studios. He worked on movie post productions including Superman Returns, Happy Feet, The Chronicles of Narnia, Garfield, Around the world in 80 days, and The Chronicles of Riddick.