Journal of the Applied Mathematics, Statistics and Informatics (JAMSI), **7** (2011), No. 2

SMALL SCALE DETAILS AND IMPROVED SURFACE RECONSTRUCTION FOR SPH

JURAJ ONDERIK, MICHAL CHLÁDEK AND ROMAN ĎURIKOVIČ

Abstract

We present a novel method for creating small scale details as splashes and foam for Smoothed Particle Hydrodynamics (SPH) simulations. In our technique, each fluid particle can become a source emitter of splash particles. The probability of emission is controlled by density decay and velocity of fluid particles. Splash particles are uncoupled, collide only with obstacles and follow only ballistic motion.

We have improved fluid surface reconstruction techniques [Zhu and Bridson 2005; Solenthaler et al. 2007] to better handle uneven distributions of particles. Our method first computes density distribution of particles which is then used for weighting the particle average similarly to previous methods. We also propose a different approach to reduce artifacts within isolated particles, without the need of eigen analysis, giving us a clean analytic expression of surface normals.

Mathematics Subject Classification 2000: I.3.5, I.3.7
Additional Key Words and Phrases: Small Scale Details, Splash, Neighbor Search, Coherency, Surface Reconstruction, Uneven Distribution

1. INTRODUCTION AND RELATED WORK

Modeling small scale details can greatly improve visual realism of large fluid simulations. Although splashes and foam can be successfully achieved with both Eulerian and Lagrangian techniques, the simulation domain must be enormously large. This is prohibitive for artists when prototyping such phenomena. Simplified techniques as shallow water simulation are still superior when modeling large bodies of water as oceans and rivers [Chentanez and Müller 2010]. Since they are unable to directly model small scale details, a number of works has focused on extending these techniques with uncoupled spray, bubble and foam particles[O'Brien and Hodgins 1995; Takahashi et al. 2003; Cleary et al. 2007]. When large, highly turbulent, splashing scenes (e.g. dam breaks, floods) are desired, these simplified methods fail. As a solution we provide in section 2 a combination of full 3D SPH fluid simulation with uncoupled splash particles to generate small scale details.

Many applications including fluid animation require realistic visualization of smooth surfaces covering some point cloud. Usually an iso-surface is defined with respect to the

©ACM, 2011. This is a minor revision of the work published in Proceedings of the Spring Conference on Computer Graphics - SCCG2011, ISBN: 978-80-223-3018-3, Vinin, Slovakia

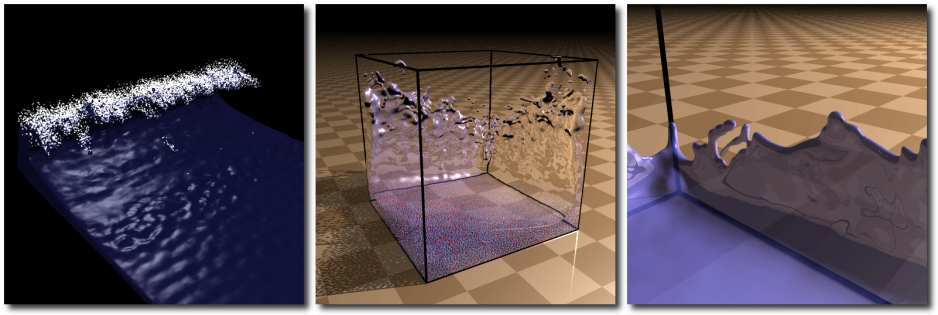

Fig. 1. Left: *Breaking wave simulation (only 20k particles) with 15k splash particles emitted using our technique.* Middle: *Dam break test in water tank (50k particles). Only red particles are sorted during coherent neighbor search.* Right: *Our improved surface reconstruction creates better thin regions (14k particles).*

particle data. This can be either converted to a polygonal mesh using Marching Cubes algorithm [Rosenberg and Birdwell 2008] or directly visualized using recent high quality volume rendering [Fraedrich et al. 2010]. Due to modern GPUs, point splatting is also a comparable alternative, mainly after a screen space post-processing [van der Laan et al. 2009]. Accurate iso-function definition is crucial when naturally looking surfaces are desired. Within the SPH context several methods based on density [Müller et al. 2003], center of mass [Zhu and Bridson 2005; Solenthaler et al. 2007], particle redistancing [Adams et al. 2007] and kernel anisotropy [Yu and Turk 2010] were presented. In the Section 3 we provide an improved surface reconstruction method which removes known artifacts and handles non-uniform particle distributions.

2. SMALL SCALE DETAILS FOR SPH

When modeling turbulent fluids with Eulerian methods, small scale details as splashes are usually considered as problematic since excessively large grid size is required. Although adaptive subdivision has been successfully employed[Losasso et al. 2004] a majority of previous works focused on exploiting simplified models of splashes and foam. Particle systems are a natural choice for modeling such phenomena in combination with almost any fluid simulation technique.

A simple extension to shallow water simulation using particle based splashes has been proposed in [O'Brien and Hodgins 1995]. In [Takahashi et al. 2003] spray particles improved the visual impact of breaking waves. Advanced bubbling and frothing effects have been achieved in [Cleary et al. 2007]. Recently [Chentanez and Müller 2010] presented a complex real-time simulation with several small scale details.

Particles have been used in numerous works (e.g. [Müller et al. 2005; Clavet et al. 2005]) to simulate full dynamics of fluids. Since particle based fluids naturally model splashing effects, less attention has been taken to extend even these methods with small scale details. Indeed, by combining an uncoupled particle system with a coupled SPH simulation one can achieve better results for large scale animations using only moderate number of fluid particles, see Figure (2).

Fig. 2. Left: *SPH fluid only.* Middle: *Splash Particles.* Right: *SPH with splash particles. The scene consists of 20k fluid particles and 14k splash particles.*

2.1 Emitting Splash Particles

Our simulation technique consist of bulk water modeled with regular SPH *fluid particles* governed either by WCSPH [Müller et al. 2003] or PCISPH [Solenthaler and Pajarola 2009] and uncoupled spray particles which represent splashes or foam. During the SPH simulation each fluid particle can become a source for emitting spray particles based on the following observations.

A compact fluid volume is a subject for splashing mainly due to a collision with some obstacle or when it is free-falling and the air friction causes it to split into drops. Within the SPH framework this can be loosely expressed with density decay near isolated particles or the free surface. During compression, when particles approach each other, their estimated density increases causing pressure forces to push them away. During splashing or collisions with obstacles, these forces can be strong enough to remove some surface particles from the bulk mass. While they get isolated, density strongly decreases giving us a good criterion for marking them source emitters for spray particles. However, measuring derivative of density is not a good idea here since relative change of density can be large due to pressure oscillations inside fluid volume as well.

We propose a simple mechanism based on the concept of *splashing probability* (from 0 to 1). Suppose some fluid particle has a full (100%) probability of splashing. We define the *maximal emit rate* as the total number of splash particles the fluid particle can emit per second. This allows us to calculate how many particles can be maximaly emitted within one simulation step. Instead of emitting all of them, we first proceed a simple stochastic test. For each splash particle we generate a uniform random number (from 0 to 1) and emit the particle only if the number is greater than the current splashing probability of the fluid particle. This gives us a consistent way to control the splash generation using only splashing probability.

We compute splashing probability using a criterion based on density decay. For each particle we first calculate its density ratio $\frac{\rho_i}{\rho_0}$. If it is under a *critical density ratio* α_d we say the fluid particle has a nonzero splashing probability. Formally the splashing probability

of i-th particle based on density decay is

$$P_i^d = \beta_d \left(\alpha_d - \frac{\rho_i}{\rho_0} \right), \tag{1}$$

where β_d is *splash density factor* which controls how strong the change in density affects the splashing probability.

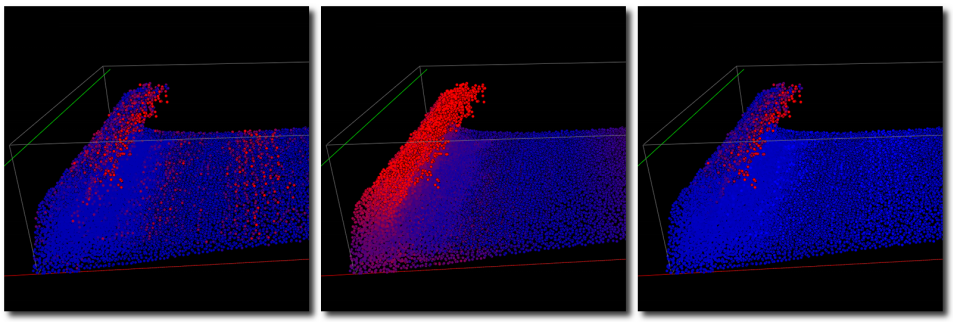

Fig. 3. *Parameters used to control splashing probability. Red particles represent larger probabilities.* Left: *Density based splashing probability.* Middle: *Velocity of particles.* Right: *Final splashing probability composed from density decay and velocity.*

As shown in the Figure (3) (left) this approach greatly captures particles on the tip of the wave. Some particles on the resting surface have low density as well, giving them higher splashing probability as well. This is, however, not desired since such particles are slowly floating on the free surface without making any splashes. We control this by constraining the splashing probability with the velocity of the particle, see Figure (3) (middle). Simple modulation with velocity length did not provide plausible results since we want to completely cut off particles with slow motion. We define splashing probability based purely on velocity as

$$P_i^v = \beta_v \left(|\mathbf{v}_i| - \alpha_v \right), \tag{2}$$

where again α_v is a *minimal splash velocity* which serves as a threshold and β_v is the *splash velocity factor* which controls how much the velocity changes the splashing probability.

Simple modulation of P_i^d and P_i^v can give positive probability also when both P_i^d and P_i^v are negative. This is the case of slow particles inside bulk fluid, which should definitely not make any splashes. Therefore we define the final splashing probability P_i of i-th particle as

$$P_i = \begin{cases} P_i^d \cdot P_i^v & P_i^d > 0 \ \wedge \ P_i^v > 0 \\ 0 & \text{otherwise.} \end{cases}. \tag{3}$$

Each emitted splash particle is first initialized with the position and velocity of its source particle adding small jitter to make it more realistic. Its life time is set to the *initial life time*. In every simulation step the its life time is decremented by the delta time until it becomes negative, making the particle dead, i.e. ready for another emission.

2.2 Negative Pressure Correction

It is important to note here, that we have modified the SPH pressure calculation to solve compression only and almost freely allow expansion. In standard SPH pressure is usually related to density using state equation

$$p(\mathbf{p}) = \frac{k\rho_0}{\gamma}\left(\left(\frac{\rho(\mathbf{p})}{\rho_0}\right)^{\gamma} - 1\right), \tag{4}$$

which can create negative pressures on isolated particles and near the boundary. Negative pressures push particles to each other creating artificial compressions. This is usually an unwanted effect which causes numerical instabilities. In case we want to model surface tension a small force pushing particles in the direction of negative pressure can be useful. Therefore we use the following modified pressure expression

$$p^*(\mathbf{p}) = \begin{cases} \lambda p(\mathbf{p}) & p(\mathbf{p}) < 0 \\ p(\mathbf{p}) & p(\mathbf{p}) \geq 0 \end{cases}, \tag{5}$$

where λ is a *negative pressure scaling factor*, which should by zero if we want to completely diminish artificial compression or a very small, positive number when modeling surface tension.

3. IMPROVED SURFACE RECONSTRUCTION

Generating a smooth surface from particles is crucial for high quality rendering of fluids. Due to major improvements possible with recent graphics hardware, real-time, screen-space based techniques become popular [van der Laan et al. 2009]. However, for high-end results, ray tracing implicit surfaces still plays an important role.

Various approaches has been already proposed throughout the literature. In [Müller et al. 2003] a color function is interpolated using SPH summation over neighbor particles. This generates a considerably bloby surface even for resting scenes, where a flat fluid surface is desired. Great improvement has been achieved [Zhu and Bridson 2005] by measuring the distance to a weighted average of neighbor particles. This approach suffers from various artifacts in concave regions and between isolated particles. This has been successfully corrected by [Solenthaler et al. 2007] using a distance decay function based on measuring the rate of change of the calculated center of mass. Based on weighted average of close particles, [Adams et al. 2007] introduced additional redistancing to achieve even flatter surfaces. Recently [Yu and Turk 2010] proposed different approach based on anisotropic scaling of kernel functions. They use PCA to estimate variance of neighboring particles and thus determine principal dimensions of scaled kernel function.

During our experiments we found that all three methods ([Müller et al. 2003], [Zhu and Bridson 2005], [Solenthaler et al. 2007]) generate an unnatural looking surface when particle distribution gets uneven. As shown in Figure (4), surface gets either bloby (row 1) or has artifacts within isolated particles (row 2) or provides unnatural shapes (row 3).

Similar to [Solenthaler et al. 2007] we define fluid surface as the zero level of the following *point-to-center* distance based implicit function

$$\phi(\mathbf{p}) = \|\mathbf{p} - \mathbf{C}(\mathbf{p})\| - Rf(\mathbf{p}). \tag{6}$$

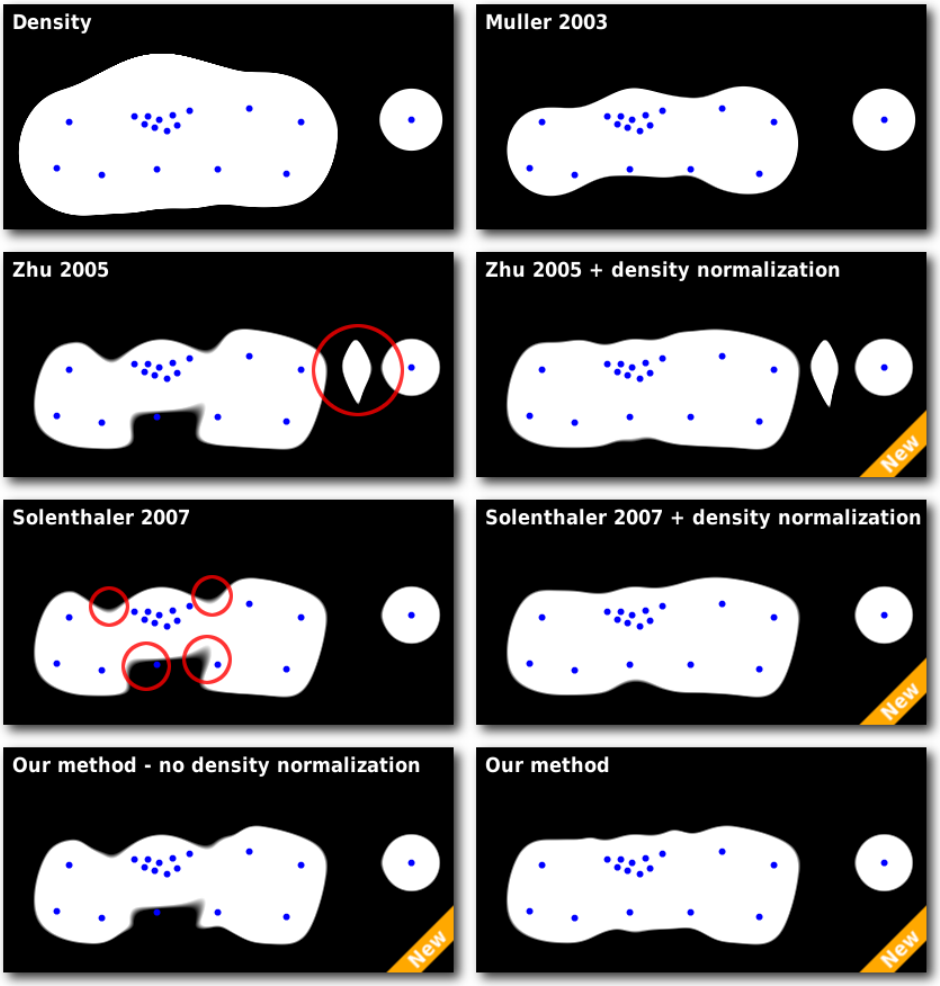

Fig. 4. *Comparison of various iso-surface construction methods. First row shows SPH density distribution (left) and color function of [Müller et al. 2003] (right). Second row depicts distance based approach of [Zhu and Bridson 2005] (left) and our density normalized modification (right). Artifacts are shown in red circles. Third row contains corrected surface by [Solenthaler et al. 2007] (left) and our density normalization for uneven distributions (right). Finally fourth row provides our new approach (left) and its density normalized version (right).*

Here $f(\mathbf{p})$ is a distance decay function (described later), R is the user defined distance of the surface from boundary particles and $C(\mathbf{p})$ is the *center* function which calculates for a given point $\mathbf{p}$ the weighted position average of neighboring particles.

As shown in figure (4) (first column) this approach can lead to unnatural surface deformations or even "ghost" regions lying outside of the fluid. As a solution we modify the center function by normalizing the weighted particle average using iso-density distribution

$(1/w_j)$ (second column)

$$\mathbf{C}(\mathbf{p}) = \frac{\mathbf{C}_1(\mathbf{p})}{C_2(\mathbf{p})} = \frac{\sum_j \frac{1}{w_j} W(\|\mathbf{p}-\mathbf{p}_j\|,h)\mathbf{p}_j}{\sum_j \frac{1}{w_j} W(\|\mathbf{p}-\mathbf{p}_j\|,h)}. \tag{7}$$

The *iso-density* w_j of j-th particle is calculated using standard SPH interpolation of unit mass of neighboring particles

$$w_i = \sum_j W(\|\mathbf{p}_i - \mathbf{p}_j\|,h), \tag{8}$$

where $W(\|\mathbf{r}\|,h)$ is a smooth kernel function. We use a simple polynomial kernel $W(r,h) = (1-\frac{r^2}{h^2})^3$ for this application. Notice the kernel support h used in surface reconstruction can be different to the physical interaction distance used in SPH. In our application we set the surface kernel support h to be 1.5x to 2x the length of the physical interaction radius.

This approach nicely overcomes problems with clustered particles, however artifacts near isolated particles still remain. As proposed in [Solenthaler et al. 2007] we can reduce these artifacts by modulating the surface distance R in equation (6) with a decay function $f(\mathbf{p})$, which is actually a factor that controls how close/far is the iso-surface from particles. Assuming two isolated particles, function $f(\mathbf{p})$ should be 1 for points close to the particles and should decay to zero in the middle between particles, scaling down artifact. This can be achieved by calculating the largest eigenvalue of the 3x3 Jacobian $\nabla C(\mathbf{p})$ and using it as a parameter that controls the amount of decay. Apart from the complexity of solving cubic equations to get eigenvalue of a 3x3 matrix, deriving an analytical expression of the surface normal (gradient of the surface) is even more complex.

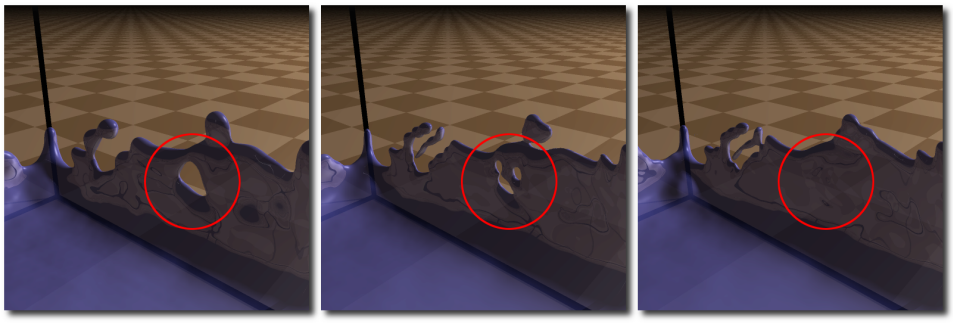

Fig. 5. Left: *Thick and bloby approach by [Müller et al. 2003].* Middle: *Distance based method of [Zhu and Bridson 2005; Solenthaler et al. 2007]. Notice some holes in the thin regions.* Right: *Our density normalized method making thin regions more compact.*

Therefore we provide an alternative definition of the distance decay function based solely on position. Formally $f(p)$ is a composition of center function $\mathbf{C}(\mathbf{c})$, normalized iso-

density $w(\mathbf{c})$ and transfer function $g(w)$

$$\begin{aligned} f(\mathbf{p}) &= g(w(\mathbf{C}(\mathbf{p}))) \\ w(\mathbf{c}) &= \sum_j \frac{1}{w_j} W(\|\mathbf{c}-\mathbf{p}_j\|, h) \\ g(w) &= \left(1 - \frac{(w-w_{\max})^2}{(w_{\max}-w_{\min})^2}\right)^2 \end{aligned} \tag{9}$$

For an arbitrary point $\mathbf{p}$ we compute $f(\mathbf{p})$ in three steps. First, we find the weighted center $\mathbf{c} = \mathbf{C}(\mathbf{c})$ using the center function $C(\mathbf{p})$. Next, we evaluate the normalized iso-density $w = w(\mathbf{c})$ at the center point $\mathbf{c}$. Finally, we smoothly transfer the normalized density into (0,1) interval using the transfer function $g(w)$ (see Figure (6) for more details).

The proposed decay function might seem complicated, however an analytical derivative $\nabla f(\mathbf{p})$ can be expressed using standard calculus. See Appendix (0**??**) for more details about surface normal (gradient) derivation $\nabla\phi(\mathbf{p})$.

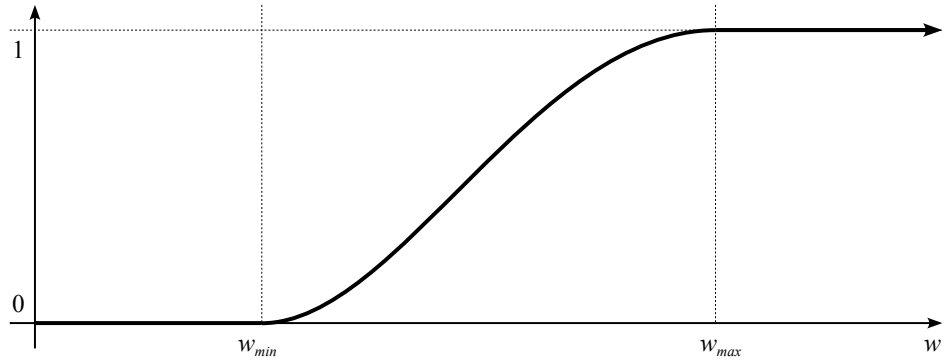

Fig. 6. *The given density input w is smoothly modulated from zero to one based on user editable minimal $w_{\min}$ and maximal $w_{\max}$ density parameters, which control the final shape of the constructed iso-surface.*

4. RESULTS

We have verified proposed methods with several testing simulations within our WCSPH, PCISPH framework and rendered using our direct implicit surface raytracer. Both applications were implemented in C++ targeting Intel x86 platform. Experiments have been performed several times with various parameters. Best results were achieved by setting particle support radius to $0.01m$, mass to $0.0002kg$, rest density to $1000kg/m^3$, viscosity to $0.4Ns/m^2$ and stiffness to $100Nm/kg$. Clamping negative pressures almost to zero together with a leap-frog integration highly increased the stability allowing us to perform even large scenes with a time-step up to $2ms$.

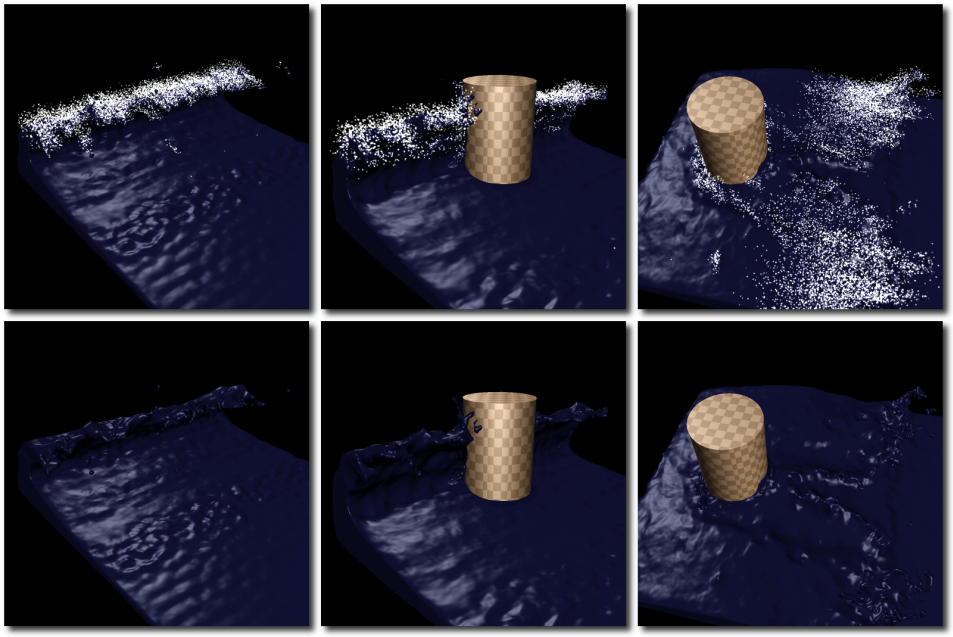

Fig. 7. *Side by side comparison of a wave breaking and a splashing simulation with (top row) or without (bottom row) slash particles. Only 20k fluid particles together with 14k uncoupled splash particles were used. Desired surface blobyness was achieved by setting iso interaction radius to only 1.3x of SPH interaction radius.*

All simulations and rendering were performed on a mobile Core2 Duo T9600 2.8GHz with 4GB RAM. In the largest scenes (512k particles) physics calculations took around 1.5 second per frame, and raytracing (1024x1024) took about 20 seconds per frame. Smaller scenes (7.5k particles) together with raytracing (256x256) were processed at interactive rates (about 10 frames per second), see Table (I)

Scene	**Simulation**	**Raytracing**	**Total**
WCSPH 27k	0.09s	1.1s	1.19s
WCSPH 64k	0.22s	2.5s	2.72s
WCSPH 125k	0.47s	4.3s	4.77s
WCSPH 512k	1.88s	19.3s	21.18s

Table I. Execution times of several dam break simulations.

4.1 Splash Generation

We have successfully demonstrated a large scale phenomena as wave breaking of ocean using SPH with our splashing technique. Only 20k fluid particles and about 15k splash particles were necessary to get plausible results, see Figure (7). This is an order of magnitude less than using only SPH with comparable results. Since splash particles are uncou-

pled (searching for neighbors in not necessary), they can be simulated in parallel gaining additional speedup.

In real units the whole scene is only about $0.6m$ wide and $0.7m$ long. Simulation duration until the first wave crash is $0.7s$. Maximal emit rate of every fluid particle was set to 700 splash particles per second, each having life time about $0.05s$. The probability of emission was calculated with Equation (3), where splashing density ratio α_d was set to 0.95 (95% of rest density), splashing density factor β_d to 2, minimal splashing velocity α_v to $0.7m/s$ and splashing velocity factor α_v to 2.

We rendered splash particles as additive semitransparent spherical point sprites. Contribution C_i of i-th particle with position p_i is accumulated along the ray based on the ray-particle distance d_i and particle life time L_i as

$$C_i = \left(1 - \frac{d_i}{d_0}\right)\left(1 - \left(2\frac{L_i}{L_0} - 1\right)^2\right), \tag{10}$$

where L_0 is the initial life time of particle and d_0 is the radius of spray particle. We set d_0 to $(0.001m)$ 10% of the fluid particle radius.

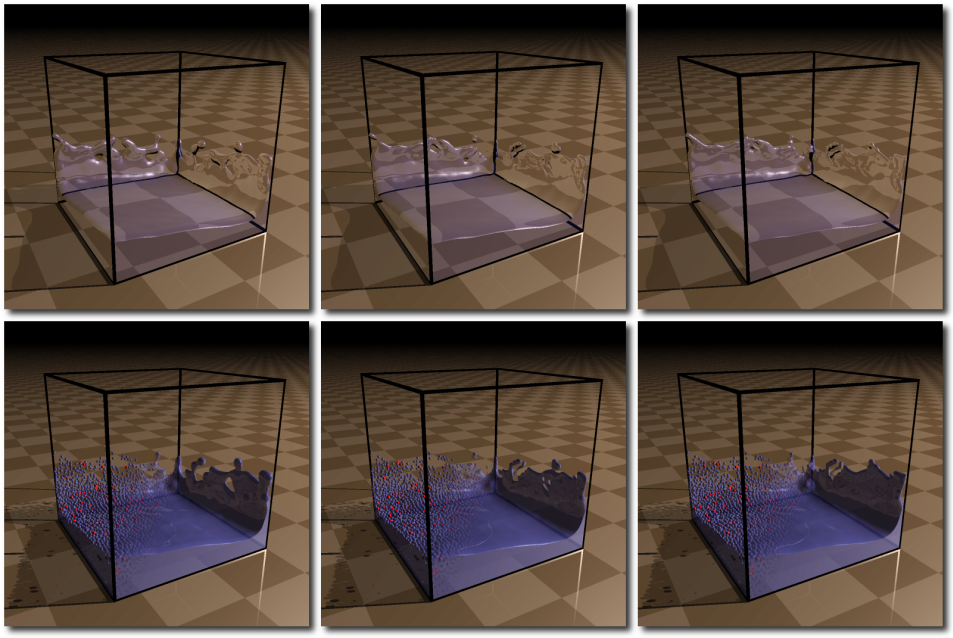

Fig. 8. *Comparison of surface generated by* Left: *[Müller et al. 2003]*, Middle: *[Zhu and Bridson 2005; Solenthaler et al. 2007] and* Right: *our density normalized method. Red particles in the bottom row have new indices used in the coherent neighbor search. Only 14k particles were used.*

4.2 Surface Reconstruction

Surface of small or moderate scenes usually looks too bloby, lacking thin regions of splashing fluid. We have thus setup an aquarium scene (0.4m x 0.4m x 0.4m) with only 14k particles for dam break test. As shown in Figure (8) and (5) the [Müller et al. 2003] method a nice smooth surface, but splashes are too bloby. Approach of [Zhu and Bridson 2005; Solenthaler et al. 2007] is less bloby, but shows several holes (no fluid) in thin regions. Due to density normalization our approach fills these holes making thin regions more compact.

The implicit surface is reconstructed with the iso level set to 0.8 (80% of iso particle radius). For the modulation function we set w_{min} to 0 and w_{max} to 1.05. To get smooth and flat surface, we use the density kernel function with support radius 2 times larger as used in SPH computation. Instead of using the physical density, we had to compute the new *iso* density with the larger kernel for each particle. This double interaction radius also has a significant impact on the performance of implicit surface raytracing since during iso funcion evaluation we had to visit 125 neighbor cells (including center cell) instead of only 27 as in SPH.

5. CONCLUSION AND FUTURE WORK

Small scale details has been added to SPH simulation by emitting spray and foam particles during splashes. Each fluid particle serves as a potential source of spray particles. We propose simple conditions based on the concept of splashing probability to control particle emission. Visual enhancements are clearly beneficial mainly when prototyping large scale simulations with moderate number of SPH particles.

Commonly adopted surface generation method [Zhu and Bridson 2005] has been improved for uneven particle distributions. We normalize the center of mass by weighting particles with their estimated iso-density. Moreover, a new approach has been proposed to prevent artifact near isolated particles. We modulate surface distance based on the decay of normalized iso-density on center of mass. In contrast to [Solenthaler et al. 2007] we do not need to perform eigenvalue analysis giving us a simple analytic expression of surface normals.

6. APPENDIX

Assuming $\mathbf{r}_j = \mathbf{p} - \mathbf{p}_j \in \mathbb{R}^{3\times 1}$ and $\nabla\|\mathbf{r}_j\| = \frac{\mathbf{r}_j^{\mathbf{T}}}{\|\mathbf{r}_j\|} \in \mathbb{R}^{1\times 3}$ surface normal (iso-function gradient) can be expressed as

$$\begin{aligned}
\nabla\phi(\mathbf{p}) &= \tfrac{(\mathbf{p}-\mathbf{C}(\mathbf{p}))^{\mathbf{T}}}{\|\mathbf{p}-\mathbf{C}(\mathbf{p})\|}\left(\mathbf{1}-\nabla\mathbf{C}(\mathbf{p})\right) - R\nabla f(\mathbf{p}) && \in \mathbb{R}^{1\times 3}\\
\nabla f(\mathbf{p}) &= g'\left(w\left(\mathbf{C}(\mathbf{p})\right)\right)\nabla w(\mathbf{C}(\mathbf{p}))\nabla\mathbf{C}(\mathbf{p}) && \in \mathbb{R}^{1\times 3}\\
\nabla w(\mathbf{c}) &= \textstyle\sum_j \frac{1}{w_j\|\mathbf{c}-\mathbf{p}_j\|} W'(\|\mathbf{c}-\mathbf{p}_j\|,h)(\mathbf{c}-\mathbf{p}_j)^{\mathbf{T}} && \in \mathbb{R}^{1\times 3}\\
g'(w) &= -4\left(1-\tfrac{(w-w_{\max})^2}{(w_{\max}-w_{\min})^2}\right)\tfrac{(w-w_{\max})}{(w_{\max}-w_{\min})^2} && \in \mathbb{R}\\
\nabla\mathbf{C}(\mathbf{p}) &= \tfrac{\nabla\mathbf{C}_1(\mathbf{p})C_2(\mathbf{p})-\mathbf{C}_1(\mathbf{p})\nabla C_2(\mathbf{p})}{C_2(\mathbf{p})C_2(\mathbf{p})} && \in \mathbb{R}^{3\times 3}\\
\nabla\mathbf{C}_1(\mathbf{p}) &= \textstyle\sum_j \frac{1}{\rho_j\|\mathbf{r}_j\|} W'(\|\mathbf{r}_j\|,h)\mathbf{p}_j\mathbf{r}_j^{\mathbf{T}} && \in \mathbb{R}^{3\times 3}\\
\nabla C_2(\mathbf{p}) &= \textstyle\sum_j \frac{1}{\rho_j\|\mathbf{r}_j\|} W'(\|\mathbf{r}_j\|,h)\mathbf{r}_j^{\mathbf{T}} && \in \mathbb{R}^{1\times 3}
\end{aligned} \tag{11}$$

Acknowledgment

This research was partially supported by a VEGA 1/0662/09 2009-2011 project a Scientific grant from Ministry of Education of Slovak Republic and Slovak Academy of Science.

REFERENCES

ADAMS, B., PAULY, M., KEISER, R., AND GUIBAS, L. J. 2007. Adaptively sampled particle fluids. In *ACM Transactions on Graphics (SIGGRAPH '07 papers)*. Vol. 26. ACM Press, New York, NY, USA, 48–48*.

CHENTANEZ, N. AND MÜLLER, M. 2010. Real-time simulation of large bodies of water with small scale details. In *Proceedings of the 2010 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. SCA '10. Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, 197–206.

CLAVET, S., BEAUDOIN, P., AND POULIN, P. 2005. Particle-based viscoelastic fluid simulation. In *Symposium on Computer Animation 2005*. 219–228.

CLEARY, P. W., PYO, S. H., PRAKASH, M., AND KOO, B. K. 2007. Bubbling and frothing liquids. *ACM Trans. Graph. 26,* 3, 97.

FRAEDRICH, R., AUER, S., AND WESTERMANN, R. 2010. Efficient high-quality volume rendering of sph data. *IEEE Transactions on Visualization and Computer Graphics (Proceedings Visualization / Information Visualization 2010) 16,* 6 (November-December), to appear.

LOSASSO, F., GIBOU, F., AND FEDKIW, R. 2004. Simulating water and smoke with an octree data structure. In *SIGGRAPH '04: ACM SIGGRAPH 2004 Papers*. ACM Press, New York, NY, USA, 457–462.

MÜLLER, M., CHARYPAR, D., AND GROSS, M. 2003. Particle-based fluid simulation for interactive applications. In *SCA '03: Proceedings of the 2003 ACM SIGGRAPH/Eurographics symposium on Computer animation*. Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, 154–159.

MÜLLER, M., SOLENTHALER, B., KEISER, R., AND GROSS, M. 2005. Particle-based fluid-fluid interaction. In *SCA '05: Proceedings of the 2005 ACM SIGGRAPH/Eurographics symposium on Computer animation*. ACM Press, New York, NY, USA, 237–244.

O'BRIEN, J. F. AND HODGINS, J. K. 1995. Dynamic simulation of splashing fluids. In *CA '95: Proceedings of the Computer Animation*. IEEE Computer Society, Washington, DC, USA, 198.

ROSENBERG, I. D. AND BIRDWELL, K. 2008. Real-time particle isosurface extraction. In *Proceedings of the 2008 symposium on Interactive 3D graphics and games*. I3D '08. ACM, New York, NY, USA, 35–43.

SOLENTHALER, B. AND PAJAROLA, R. 2009. Predictive-corrective incompressible sph. *ACM Trans. Graph. 28*, 40:1–40:6.

SOLENTHALER, B., SCHLÄFLI, J., AND PAJAROLA, R. 2007. A unified particle model for fluid-solid interactions. *Computer Animation and Virtual Worlds 18*, 69–82.

TAKAHASHI, T., FUJII, H., KUNIMATSU, A., HIWADA, K., SAITO, T., TANAKA, K., AND UEKI, H. 2003. Realistic animation of fluid with splash and foam. *Computer Graphics Forum 22,* 3, 391–400.

VAN DER LAAN, W. J., GREEN, S., AND SAINZ, M. 2009. Screen space fluid rendering with curvature flow. In *Proceedings of the 2009 symposium on Interactive 3D graphics and games*. I3D '09. ACM, New York, NY, USA, 91–98.

YU, J. AND TURK, G. 2010. Reconstructing surfaces of particle-based fluids using anisotropic kernels. In *Proceedings of the 2010 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. SCA '10. Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, 217–225.

ZHU, Y. AND BRIDSON, R. 2005. Animating sand as a fluid. *ACM Trans. Graph. 24,* 3, 965–972.

Juraj Onderik
Mathematics, Physics and Informatics,
Comenius University, 842 48 Bratislava, Slovak Republic
http://www.fmph.uniba.sk
email: juraj.onderik@fmph.uniba.sk

Michal Chládek
Mathematics, Physics and Informatics,
Comenius University, 842 48 Bratislava, Slovak Republic
http://www.fmph.uniba.sk
email: michal.chladek@fmph.uniba.sk

Roman Ďurikovič
Faculty of Mathematics, Physics and Informatics,
Comenius University,
842 48 Bratislava, Slovak Republic;
http://www.fmph.uniba.sk
email: roman.durikovic@fmph.uniba.sk

Received May 2011